PyGithub Documentation

Release 1.43.7

Vincent Jacques

Jun 24, 2019

Contents

CHAPTER 1

Introduction

PyGithub is a Python (2 and 3) library to use the Github API v3. With it, you can manage your Github resources (repositories, user profiles, organizations, etc.) from Python scripts.

Should you have any question, any remark, or if you find a bug, or if there is something you can do with the API but not with PyGithub, please open an issue.

1.1 (Very short) tutorial

First create a Github instance:

```
from github import Github

# using username and password
g = Github("user", "password")

# or using an access token
g = Github("access_token")

# Github Enterprise with custom hostname
g = Github(base_url="https://{hostname}/api/v3", login_or_token="access_token")
```

Then play with your Github objects:

```
for repo in g.get_user().get_repos():
    print(repo.name)
    repo.edit(has_wiki=False)
    # to see all the available attributes and methods
    print(dir(repo))
```

1.2 Download and install

This package is in the Python Package Index, so `pip install PyGithub` should be enough. You can also clone it on Github.

If you wish to use GitHub Integrations, you'll want to be sure to install the 'integrations' option: `pip install PyGithub['integrations']`

1.3 Licensing

PyGithub is distributed under the GNU Lesser General Public Licence. See files COPYING and COPYING.LESSER, as requested by GNU.

1.4 What next?

You need to use a Github API and wonder which class implements it? Reference of APIs.

You want all the details about PyGithub classes? Reference of Classes.

1.5 Projects using PyGithub

(Open an issue if you want to be listed here, I'll be glad to add your project)

- Github-iCalendar returns all of your Github issues and pull requests as a list of tasks / VTODO items in iCalendar format.
- DevAssistant
- Upverter is a web-based schematic capture and PCB layout tool for people who design electronics. Designers can attach a Github project to an Upverter project.
- Notifico receives messages (such as commits and issues) from services and scripts and delivers them to IRC channels. It can import/sync from Github.
- Tratihubis converts Trac tickets to Github issues
- https://github.com/fga-gpp-mds/2018.1-Cardinals - website that shows metrics for any public repository (issues, commits, pull requests etc)
- https://github.com/CMB/cligh
- https://github.com/natduca/quickopen uses PyGithub to automaticaly create issues
- https://gist.github.com/3433798
- https://github.com/zsiciarz/aquila-dsp.org
- https://github.com/robcowie/virtualenvwrapper.github
- https://github.com/kokosing/git-gifi - Git and github enhancements to git.
- https://github.com/csurfer/gitsuggest - A tool to suggest github repositories based on the repositories you have shown interest in
- https://github.com/gomesfernanda/some-github-metrics - Python functions for relevant metrics on GitHub repositories

- https://github.com/SOM-Research/Gitana - a SQL-based Project Activity Inspector
- https://github.com/plus3it/satsuki - Automate GitHub releases and uploading binary release assets
- check-in — Python CLI distribution that allows user to use GitHub Checks API as a bot.

1.6 They talk about PyGithub

- http://stackoverflow.com/questions/10625190/most-suitable-python-library-for-github-api-v3
- http://stackoverflow.com/questions/12379637/django-social-auth-github-authentication
- http://www.freebsd.org/cgi/cvsweb.cgi/ports/devel/py-pygithub/
- https://bugzilla.redhat.com/show_bug.cgi?id=910565

CHAPTER 2

Examples

2.1 Main Class

This is the main class.

2.1.1 Get current user

```
>>> user = g.get_user()
>>> user.login
u'sfdye'
```

2.1.2 Get user by name

```
>>> user = g.get_user("sfdye")
>>> user.name
u'Wan Liuyang'
```

2.1.3 Get repository by name

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.name
u'PyGithub'
```

2.1.4 Get organization by name

```
>>> org = g.get_organization("PyGithub")
>>> org.login
u'PyGithub'
```

2.1.5 Search repositories by language

```
>>> repositories = g.search_repositories(query='language:python')
>>> for repo in repositories:
...     print(repo)
...
Repository(full_name="vinta/awesome-python")
Repository(full_name="donnemartin/system-design-primer")
Repository(full_name="toddmotto/public-apis")
Repository(full_name="rg3/youtube-dl")
Repository(full_name="tensorflow/models")
Repository(full_name="django/django")
```

2.1.6 Search repositories based on number of issues with good-first-issue

```
>>> repositories = g.search_repositories(query='good-first-issues:>3')
>>> for repo in repositories:
...     print(repo)
...
Repository(full_name="vuejs/vue")
Repository(full_name="facebook/react")
Repository(full_name="facebook/react-native")
Repository(full_name="electron/electron")
Repository(full_name="Microsoft/vscode")
```

2.2 Repository

2.2.1 Get repository topics

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.get_topics()
[u'pygithub', u'python', u'github', u'github-api']
```

2.2.2 Get count of stars

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.stargazers_count
2086
```

2.2.3 Get list of open issues

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> open_issues = repo.get_issues(state='open')
>>> for issue in open_issues:
...     print(issue)
...
Issue(title="How to get public events?", number=913)
Issue(title="Prevent .netrc from overwriting Auth header", number=910)
Issue(title="Cache fetch responses", number=901)
Issue(title="Is suspended_users for github enterprise implemented in NamedUser?",␣
↪number=900)
Issue(title="Adding migration api wrapper", number=899)
```

2.2.4 Get all the labels of the repository

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> labels = repo.get_labels()
>>> for label in labels:
...     print(label)
...
Label(name="Hacktoberfest")
Label(name="WIP")
Label(name="bug")
Label(name="documentation")
```

2.2.5 Get all of the contents of the root directory of the repository

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_contents("")
>>> for content_file in contents:
...     print(content_file)
...
ContentFile(path=".github")
ContentFile(path=".gitignore")
ContentFile(path=".travis.yml")
ContentFile(path="CONTRIBUTING.md")
ContentFile(path="COPYING")
ContentFile(path="COPYING.LESSER")
ContentFile(path="MAINTAINERS")
ContentFile(path="MANIFEST.in")
ContentFile(path="README.md")
ContentFile(path="doc")
ContentFile(path="github")
ContentFile(path="manage.sh")
ContentFile(path="requirements.txt")
ContentFile(path="scripts")
ContentFile(path="setup.py")
```

2.2.6 Get all of the contents of the repository recursively

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_contents("")
```

(continues on next page)

(continued from previous page)

```
>>> while len(contents) > 1:
...     file_content = contents.pop(0)
...     if file_content.type == "dir":
...         contents.extend(repo.get_contents(file_content.path))
...     else:
...         print(file_content)
...
ContentFile(path=".gitignore")
ContentFile(path=".travis.yml")
ContentFile(path="CONTRIBUTING.md")
...
ContentFile(path="github/tests/ReplayData/Team.testRepoPermission.txt")
ContentFile(path="github/tests/ReplayData/Team.testRepos.txt")
ContentFile(path="github/tests/ReplayData/UserKey.setUp.txt")
```

2.2.7 Get a specific content file

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_contents("README.md")
>>> print(contents)
...
ContentFile(path="README.md")
```

2.2.8 Create a new file in the repository

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.create_file("test.txt", "test", "test", branch="test")
{'content': ContentFile(path="test.txt"), 'commit': Commit(sha=
↪"5b584cf6d32d960bb7bee8ce94f161d939aec377")}
```

2.2.9 Update a file in the repository

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_contents("test.txt", ref="test")
>>> repo.update_file(contents.path, "more tests", "more tests", contents.sha, branch=
↪"test")
{'commit': Commit(sha="b06e05400afd6baee13fff74e38553d135dca7dc"), 'content':␣
↪ContentFile(path="test.txt")}
```

2.2.10 Delete a file in the repository

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_contents("test.txt", ref="test")
>>> repo.delete_file(contents.path, "remove test", contents.sha, branch="test")
{'commit': Commit(sha="0f40b0b4f31f62454f1758d7e6b384795e48fd96"), 'content': NotSet}
```

2.2.11 Get the top 10 referrers over the last 14 days

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_top_referrers()
>>> print(contents)
[
  Referrer(referrer="Google", count=4, uniques=3),
  Referrer(referrer="stackoverflow.com", count=2, uniques=2),
  Referrer(referrer="eggsonbread.com", count=1, uniques=1),
  Referrer(referrer="yandex.ru", count=1, uniques=1)
]
```

2.2.12 Get the top 10 popular contents over the last 14 days

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_top_paths()
>>> print(contents)
[
  Path(path="/github/hubot", title="github/hubot: A customizable life embetterment␣
↪robot.", count=3542, uniques=2225),
  Path(path="/github/hubot/blob/master/docs/scripting.md", title="hubot/scripting.md␣
↪at master · github/hubot · GitHub", count=1707, uniques=804),
  Path(path="/github/hubot/tree/master/docs", title="hubot/docs at master · github/
↪hubot · GitHub", count=685, uniques=435),
  Path(path="/github/hubot/tree/master/src", title="hubot/src at master · github/
↪hubot · GitHub", count=577, uniques=347),
  Path(path="/github/hubot/blob/master/docs/index.md", title="hubot/index.md at␣
↪master · github/hubot · GitHub", count=379, uniques=259),
  Path(path="/github/hubot/blob/master/docs/adapters.md", title="hubot/adapters.md at␣
↪master · github/hubot · GitHub", count=354, uniques=201),
  Path(path="/github/hubot/tree/master/examples", title="hubot/examples at master ·␣
↪github/hubot · GitHub", count=340, uniques=260),
  Path(path="/github/hubot/blob/master/docs/deploying/heroku.md", title="hubot/heroku.
↪md at master · github/hubot · GitHub", count=324, uniques=217),
  Path(path="/github/hubot/blob/master/src/robot.coffee", title="hubot/robot.coffee␣
↪at master · github/hubot · GitHub", count=293, uniques=191),
  Path(path="/github/hubot/blob/master/LICENSE.md", title="hubot/LICENSE.md at master␣
↪· github/hubot · GitHub", count=281, uniques=222)
]
```

2.2.13 Get number of clones and breakdown for the last 14 days

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_clones_traffic()
>>> contents = repo.get_clones_traffic(per="week")
>>> print(contents)
{
  "count": 173,
  "uniques": 128,
  "clones": [
    Clones(timestamp=2016-10-10 00:00:00, count=2, uniques=1),
    Clones(timestamp=2016-10-11 00:00:00, count=17, uniques=16),
    Clones(timestamp=2016-10-12 00:00:00, count=21, uniques=15),
```

(continues on next page)

(continued from previous page)

```
    Clones(timestamp=2016-10-13 00:00:00, count=8, uniques=7),
    Clones(timestamp=2016-10-14 00:00:00, count=5, uniques=5),
    Clones(timestamp=2016-10-15 00:00:00, count=2, uniques=2),
    Clones(timestamp=2016-10-16 00:00:00, count=8, uniques=7),
    Clones(timestamp=2016-10-17 00:00:00, count=26, uniques=15),
    Clones(timestamp=2016-10-18 00:00:00, count=19, uniques=17),
    Clones(timestamp=2016-10-19 00:00:00, count=19, uniques=14),
    Clones(timestamp=2016-10-20 00:00:00, count=19, uniques=15),
    Clones(timestamp=2016-10-21 00:00:00, count=9, uniques=7),
    Clones(timestamp=2016-10-22 00:00:00, count=5, uniques=5),
    Clones(timestamp=2016-10-23 00:00:00, count=6, uniques=5),
    Clones(timestamp=2016-10-24 00:00:00, count=7, uniques=5)
  ]
}
```

2.2.14 Get number of views and breakdown for the last 14 days

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> contents = repo.get_views_traffic()
>>> contents = repo.get_views_traffic(per="week")
>>> print(contents)
{
  "count": 14850,
  "uniques": 3782,
  "views": [
    View(timestamp=2016-10-10 00:00:00, count=440, uniques=143),
    View(timestamp=2016-10-11 00:00:00, count=1308, uniques=414),
    View(timestamp=2016-10-12 00:00:00, count=1486, uniques=452),
    View(timestamp=2016-10-13 00:00:00, count=1170, uniques=401),
    View(timestamp=2016-10-14 00:00:00, count=868, uniques=266),
    View(timestamp=2016-10-15 00:00:00, count=495, uniques=157),
    View(timestamp=2016-10-16 00:00:00, count=524, uniques=175),
    View(timestamp=2016-10-17 00:00:00, count=1263, uniques=412),
    View(timestamp=2016-10-18 00:00:00, count=1402, uniques=417),
    View(timestamp=2016-10-19 00:00:00, count=1394, uniques=424),
    View(timestamp=2016-10-20 00:00:00, count=1492, uniques=448),
    View(timestamp=2016-10-21 00:00:00, count=1153, uniques=332),
    View(timestamp=2016-10-22 00:00:00, count=566, uniques=168),
    View(timestamp=2016-10-23 00:00:00, count=675, uniques=184),
    View(timestamp=2016-10-24 00:00:00, count=614, uniques=237)
  ]
}
```

2.2.15 Mark the notifications of the repository as read

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.mark_notifications_as_read()
```

2.3 Branch

2.3.1 Get list of branches

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> list(repo.get_branches())
[Branch(name="master")]
```

Note that the Branch object returned by get_branches() is not fully populated, and you can not query everything. Use get_branch(branch="master") once you have the branch name.

2.3.2 Get a branch

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.get_branch(branch="master")
Branch(name="master")
```

2.3.3 Get HEAD commit of a branch

```
>>> branch = g.get_repo("PyGithub/PyGithub").get_branch("master")
>>> branch.commit
Commit(sha="5e69ff00a3be0a76b13356c6ff42af79ff469ef3")
```

2.3.4 Get protection status of a branch

```
>>> branch = g.get_repo("PyGithub/PyGithub").get_branch("master")
>>> branch.protection
True
```

2.3.5 See required status checks of a branch

```
>>> branch = g.get_repo("PyGithub/PyGithub").get_branch("master")
>>> branch.get_required_status_checks()
RequiredStatusChecks(url="https://api.github.com/repos/PyGithub/PyGithub/branches/
↪master/protection/required_status_checks", strict=True)
```

2.4 Commit

2.4.1 Create commit status check

```
# sha -> commit on which the status check will be created
# For example, for a webhook payload
# sha = data["pull_request"]["head"]["sha"]
repo.get_commit(sha=sha).create_status(
    state="pending",
```

(continues on next page)

(continued from previous page)

```
    target_url="https://FooCI.com",
    description="FooCI is building",
    context="ci/FooCI"
)
```

2.4.2 Get commit date

```
>>> commit = repo.get_commit(sha=sha)
>>> print(commit.commit.author.date)
2018-10-11 03:04:52
>>> print(commit.commit.committer.date)
2018-10-11 03:04:52
```

2.5 PullRequest

2.5.1 Get Pull Request by Number

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> pr = repo.get_pull(664)
>>> pr
PullRequest(title="Use 'requests' instead of 'httplib'", number=664)
```

2.5.2 Get Pull Requests by Query

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> pulls = repo.get_pulls(state='open', sort='created', base='master')
>>> for pr in pulls:
...     print(pr.number)
...
400
861
875
876
```

2.6 Issues

2.6.1 Get issue

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.get_issue(number=874)
    Issue(title="PyGithub example usage", number=874)
```

2.6.2 Create issue

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.create_issue(title="This is a new issue")
    Issue(title="This is a new issue", number=XXX)
```

2.6.3 Create issue with body

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.create_issue(title="This is a new issue", body="This is the issue body")
    Issue(title="This is a new issue", number=XXX)
```

2.6.4 Create issue with labels

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> label = repo.get_label("My Label")
>>> repo.create_issue(title="This is a new issue", labels=[label])
    Issue(title="This is a new issue", number=XXX)
```

2.6.5 Create issue with assignee

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> repo.create_issue(title="This is a new issue", assignee="github-username")
    Issue(title="This is a new issue", number=XXX)
```

2.6.6 Create issue with milestone

```
>>> repo = g.get_repo("PyGithub/PyGithub")
>>> milestone = repo.create_milestone("New Issue Milestone")
>>> repo.create_issue(title="This is a new issue", milestone=milestone)
    Issue(title="This is a new issue", number=XXX)
```

2.7 Milestone

2.7.1 Get Milestone list

```
>>> repo = g.get_repo('PyGithub/PyGithub')
>>> open_milestones = repo.get_milestones(state='open')
>>> for milestone in open_milestones:
...     print(milestone)
...
Milestone(number=1)
Milestone(number=2)
```

2.7.2 Get Milestone

```
>>> repo = g.get_repo('PyGithub/PyGithub')
>>> repo.get_milestone(number=1)
Milestone(number=1)
```

2.7.3 Create Milestone

```
>>> repo = g.get_repo('PyGithub/PyGithub')
>>> repo.create_milestone(title='New Milestone')
Milestone(number=1)
```

2.7.4 Create Milestone with State and Description

```
>>> repo = g.get_repo('PyGithub/PyGithub')
>>> repo.create_milestone(title='New Milestone', state='open', description='Milestone␣
↪description')
Milestone(number=1)
```

2.8 Webhook

2.8.1 Creating and Listening to Webhooks with PyGithub and Pyramid

To receive a continuous stream of events, one can set up a wsgiref app using Pyramid to handle incoming POST requests.

The below code sets up a listener which creates and utilizes a webhook. Using 'pull_request' and 'push' for the EVENT attributes, any time a PR is opened, closed, merged, or synced, or a commit is pushed, Github sends a POST containing a payload with information about the PR/push and its state.

The below example was drawn largely from Github's Examples on working with Webhooks. A list of all applicable event types for Webhooks can be found in Github's documentation

```
from __future__ import print_function

from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.view import view_config, view_defaults
from pyramid.response import Response
from github import Github

ENDPOINT = "webhook"

@view_defaults(
    route_name=ENDPOINT, renderer="json", request_method="POST"
)
class PayloadView(object):
    """
    View receiving of Github payload. By default, this view it's fired only if
    the request is json and method POST.
```

(continues on next page)

(continued from previous page)

```
    """

    def __init__(self, request):
        self.request = request
        # Payload from Github, it's a dict
        self.payload = self.request.json

    @view_config(header="X-Github-Event:push")
    def payload_push(self):
        """This method is a continuation of PayloadView process, triggered if
        header HTTP-X-Github-Event type is Push"""
        print("No. commits in push:", len(self.payload['commits']))
        return Response("success")

    @view_config(header="X-Github-Event:pull_request")
    def payload_pull_request(self):
        """This method is a continuation of PayloadView process, triggered if
        header HTTP-X-Github-Event type is Pull Request"""
        print("PR", self.payload['action'])
        print("No. Commits in PR:", self.payload['pull_request']['commits'])

        return Response("success")

    @view_config(header="X-Github-Event:ping")
    def payload_else(self):
        print("Pinged! Webhook created with id {}!".format(self.payload["hook"]["id
↪"]))
        return {"status": 200}


def create_webhook():
    """ Creates a webhook for the specified repository.

    This is a programmatic approach to creating webhooks with PyGithub's API. If you␣
↪wish, this can be done
    manually at your repository's page on Github in the "Settings" section. There is␣
↪a option there to work with
    and configure Webhooks.
    """

    USERNAME = ""
    PASSWORD = ""
    OWNER = ""
    REPO_NAME = ""
    EVENTS = ["push", "pull_request"]
    HOST = ""

    config = {
        "url": "http://{host}/{endpoint}".format(host=HOST, endpoint=ENDPOINT),
        "content_type": "json"
    }

    g = Github(USERNAME, PASSWORD)
    repo = g.get_repo("{owner}/{repo_name}".format(owner=OWNER, repo_name=REPO_NAME))
    repo.create_hook("web", config, EVENTS, active=True)
```

(continues on next page)

(continued from previous page)

```
if __name__ == "__main__":
    config = Configurator()

    create_webhook()

    config.add_route(ENDPOINT, "/{}".format(ENDPOINT))
    config.scan()

    app = config.make_wsgi_app()
    server = make_server("0.0.0.0", 80, app)
    server.serve_forever()
```

Outputs from a server configured as above:

```
x.y.w.z - - [15/Oct/2018 23:49:19] "POST /webhook HTTP/1.1" 200 15
Pinged! Webhook created with id <redacted id>!
No. commits in push: 1
x.y.w.z - - [15/Oct/2018 23:49:32] "POST /webhook HTTP/1.1" 200 7
PR synchronize
x.y.w.z - - [15/Oct/2018 23:49:33] "POST /webhook HTTP/1.1" 200 7
No. Commits in PR: 10
PR closed
x.y.w.z - - [15/Oct/2018 23:49:56] "POST /webhook HTTP/1.1" 200 7
No. Commits in PR: 10
x.y.w.z - - [15/Oct/2018 23:50:00] "POST /webhook HTTP/1.1" 200 7
PR reopened
No. Commits in PR: 10
```

CHAPTER 3

Reference

The primary class you will instanciate is *github.MainClass.Github*. From its `get_`, `create_` methods, you will obtain instances of all Github objects like *github.NamedUser.NamedUser* or *github.Repository.Repository*.

All classes inherit from *github.GithubObject.GithubObject*.

3.1 Main class: Github

class github.MainClass.**Github**(*login_or_token=None*, *password=None*, *jwt=None*, *base_url='https://api.github.com'*, *timeout=15*, *client_id=None*, *client_secret=None*, *user_agent='PyGithub/Python'*, *per_page=30*, *api_preview=False*, *verify=True*, *retry=None*)

This is the main class you instantiate to access the Github API v3. Optional parameters allow different authentication methods.

Parameters

- **login_or_token** – string
- **password** – string
- **base_url** – string
- **timeout** – integer
- **client_id** – string
- **client_secret** – string
- **user_agent** – string
- **per_page** – int
- **verify** – boolean or string
- **retry** – int or urllib3.util.retry.Retry object

FIX_REPO_GET_GIT_REF

Type bool

per_page

Type int

rate_limiting
First value is requests remaining, second value is request limit.

Type (int, int)

rate_limiting_resettime
Unix timestamp indicating when rate limiting will reset.

Type int

get_rate_limit()
Rate limit status for different resources (core/search/graphql).

Calls GET /rate_limit

Return type *github.RateLimit.RateLimit*

oauth_scopes

Type list of string

get_license(*key=NotSet*)

Calls GET /license/:license

Parameters **key** – string

Return type *github.License.License*

get_licenses()

Calls GET /licenses

Return type *github.PaginatedList.PaginatedList* of *github.License.License*

get_user(*login=NotSet*)

Calls GET /users/:user or GET /user

Parameters **login** – string

Return type *github.NamedUser.NamedUser*

get_users(*since=NotSet*)

Calls GET /users

Parameters **since** – integer

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_organization(*login*)

Calls GET /orgs/:org

Parameters **login** – string

Return type *github.Organization.Organization*

get_organizations(*since=NotSet*)

Calls GET /organizations

Parameters **since** – integer

Return type *github.PaginatedList.PaginatedList* of *github.Organization.Organization*

get_repo(*full_name_or_id*, *lazy=False*)

Calls GET /repos/:owner/:repo or GET /repositories/:id

Return type *github.Repository.Repository*

get_repos(*since=NotSet*, *visibility=NotSet*)

Calls GET /repositories

Parameters

- **since** – integer
- **visibility** – string ('all','public')

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_project(*id*)

Calls GET /projects/:project_id

Return type *github.Project.Project*

Parameters **id** – integer

get_gist(*id*)

Calls GET /gists/:id

Parameters **id** – string

Return type *github.Gist.Gist*

get_gists(*since=NotSet*)

Calls GET /gists/public

Parameters **since** – datetime.datetime format YYYY-MM-DDTHH:MM:SSZ

Return type *github.PaginatedList.PaginatedList* of *github.Gist.Gist*

search_repositories(*query*, *sort=NotSet*, *order=NotSet*, ***qualifiers*)

Calls GET /search/repositories

Parameters

- **query** – string
- **sort** – string ('stars', 'forks', 'updated')
- **order** – string ('asc', 'desc')
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

search_users(*query*, *sort=NotSet*, *order=NotSet*, ***qualifiers*)

Calls GET /search/users

Parameters

- **query** – string
- **sort** – string ('followers', 'repositories', 'joined')
- **order** – string ('asc', 'desc')
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

search_issues(*query*, *sort=NotSet*, *order=NotSet*, ***qualifiers*)

Calls GET /search/issues

Parameters

- **query** – string
- **sort** – string ('comments', 'created', 'updated')
- **order** – string ('asc', 'desc')
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

search_code(*query*, *sort=NotSet*, *order=NotSet*, *highlight=False*, ***qualifiers*)

Calls GET /search/code

Parameters

- **query** – string
- **sort** – string ('indexed')
- **order** – string ('asc', 'desc')
- **highlight** – boolean (True, False)
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.ContentFile.ContentFile*

search_commits(*query*, *sort=NotSet*, *order=NotSet*, ***qualifiers*)

Calls GET /search/commits

Parameters

- **query** – string
- **sort** – string ('author-date', 'committer-date')
- **order** – string ('asc', 'desc')
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.Commit.Commit*

search_topics(*query*, ***qualifiers*)

Calls GET /search/topics

Parameters

- **query** – string
- **qualifiers** – keyword dict query qualifiers

Return type *github.PaginatedList.PaginatedList* of *github.Topic.Topic*

render_markdown(*text*, *context=NotSet*)

Calls POST /markdown

Parameters

- **text** – string
- **context** – *github.Repository.Repository*

Return type string

get_hook(*name*)

Calls GET /hooks/:name

Parameters **name** – string

Return type *github.HookDescription.HookDescription*

get_hooks()

Calls GET /hooks

Return type list of *github.HookDescription.HookDescription*

get_gitignore_templates()

Calls GET /gitignore/templates

Return type list of string

get_gitignore_template(*name*)

Calls GET /gitignore/templates/:name

Return type *github.GitignoreTemplate.GitignoreTemplate*

get_emojis()

Calls GET /emojis

Return type dictionary of type => url for emoji‘

create_from_raw_data(*klass*, *raw_data*, *headers={}*)

Creates an object from raw_data previously obtained by *github.GithubObject.GithubObject.raw_data*, and optionaly headers previously obtained by *github.GithubObject.GithubObject.raw_headers*.

Parameters

- **klass** – the class of the object to create
- **raw_data** – dict
- **headers** – dict

Return type instance of class `klass`

dump(*obj*, *file*, *protocol=0*)

Dumps (pickles) a PyGithub object to a file-like object. Some effort is made to not pickle sensitive informations like the Github credentials used in the *Github* instance. But NO EFFORT is made to remove sensitive information from the object's attributes.

Parameters

- **obj** – the object to pickle
- **file** – the file-like object to pickle to
- **protocol** – the pickling protocol

load(*f*)

Loads (unpickles) a PyGithub object from a file-like object.

Parameters **f** – the file-like object to unpickle from

Returns the unpickled object

get_installation(*id*)

Parameters **id** –

Returns

3.2 APIs

- /authorizations
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_authorizations()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_authorization()*
- /authorizations/:id
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_authorization()*
  - PATCH: *github.Authorization.Authorization.edit()*
  - DELETE: *github.Authorization.Authorization.delete()*
- /emojis
  - GET: *github.MainClass.Github.get_emojis()*
- /events
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_events()*
- /gists
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_gists()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_gist()*
- /gists/:gist_id/comments
  - GET: *github.Gist.Gist.get_comments()*
  - POST: *github.Gist.Gist.create_comment()*
- /gists/:gist_id/comments/:id
  - GET: *github.Gist.Gist.get_comment()*
  - PATCH: *github.GistComment.GistComment.edit()*
  - DELETE: *github.GistComment.GistComment.delete()*
- /gists/:id
  - GET: *github.MainClass.Github.get_gist()*

  - PATCH: *github.Gist.Gist.edit()*
  - DELETE: *github.Gist.Gist.delete()*
- `/gists/:id/forks`
  - POST: *github.Gist.Gist.create_fork()*
- `/gists/:id/star`
  - GET: *github.Gist.Gist.is_starred()*
  - PUT: *github.Gist.Gist.set_starred()*
  - DELETE: *github.Gist.Gist.reset_starred()*
- `/gists/public`
  - GET: *github.MainClass.Github.get_gists()*
- `/gists/starred`
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_starred_gists()*
- `/gitignore/templates`
  - GET: *github.MainClass.Github.get_gitignore_templates()*
- `/gitignore/templates/:name`
  - GET: *github.MainClass.Github.get_gitignore_template()*
- `/hooks`
  - GET: *github.MainClass.Github.get_hooks()*
- `/hooks/:name`
  - GET: *github.MainClass.Github.get_hook()*
- `/hub`
  - POST: *github.Repository.Repository.subscribe_to_hub()* or *github.Repository.Repository.unsubscribe_from_hub()*
- `/installation/repositories`
  - GET: *github.Installation.Installation.get_repos()*
- `/issues`
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_issues()*
- `/legacy/issues/search/:owner/:repository/:state/:keyword`
  - GET: *github.Repository.Repository.legacy_search_issues()*
- `/license/:license`
  - GET: *github.MainClass.Github.get_license()*
- `/licenses`
  - GET: *github.MainClass.Github.get_licenses()*
- `/markdown`
  - POST: *github.MainClass.Github.render_markdown()*
- `/markdown/raw`

  - POST: Not implemented, see `/markdown`
- `/networks/:owner/:repo/events`
  - GET: *github.Repository.Repository.get_network_events()*
- `/notifications`
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_notifications()*
  - PUT: *github.AuthenticatedUser.AuthenticatedUser.mark_notifications_as_read()*
- `/notifications/threads/:id`
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_notification()*
  - PATCH: *github.Notification.Notification.mark_as_read()*
- `/organizations`
  - GET: *github.MainClass.Github.get_organizations()*
- `/orgs/:org`
  - GET: *github.MainClass.Github.get_organization()*
  - PATCH: *github.Organization.Organization.edit()*
- `/orgs/:org/events`
  - GET: *github.Organization.Organization.get_events()*
- `/orgs/:org/invitations`
  - POST: *github.Organization.Organization.invite_user()*
- `/orgs/:org/issues`
  - GET: *github.Organization.Organization.get_issues()*
- `/orgs/:org/members`
  - GET: *github.Organization.Organization.get_members()*
- `/orgs/:org/members/:user`
  - GET: *github.Organization.Organization.has_in_members()*
  - DELETE: *github.Organization.Organization.remove_from_members()*
- `/orgs/:org/memberships/:user`
  - PUT: *github.Organization.Organization.add_to_members()*
  - DELETE: *github.Organization.Organization.remove_from_membership()*
- `/orgs/:org/migrations`_
  - GET: *github.Organization.Organization.get_migrations()*
  - POST: *github.Organization.Organization.create_migration()*
- `/orgs/:org/outside_collaborators`
  - GET: *github.Organization.Organization.get_outside_collaborators()*
- `/orgs/:org/outside_collaborators/:username`
  - PUT: *github.Organization.Organization.convert_to_outside_collaborator()*
  - DELETE: *github.Organization.Organization.remove_outside_collaborator()*

- /orgs/:org/projects
  - GET: *github.Organization.Organization.get_projects()*
- /orgs/:org/public_members
  - GET: *github.Organization.Organization.get_public_members()*
- /orgs/:org/public_members/:user
  - GET: *github.Organization.Organization.has_in_public_members()*
  - PUT: *github.Organization.Organization.add_to_public_members()*
  - DELETE: *github.Organization.Organization.remove_from_public_members()*
- /orgs/:org/repos
  - GET: *github.Organization.Organization.get_repos()*
  - POST: *github.Organization.Organization.create_repo()*
- /orgs/:org/teams
  - GET: *github.Organization.Organization.get_teams()*
  - POST: *github.Organization.Organization.create_team()*
- /orgs/:org/teams/:team_slug
  - GET: *github.Organization.Organization.get_team_by_slug()*
- /orgs/:owner/hooks
  - GET: *github.Organization.Organization.get_hooks()*
  - POST: *github.Organization.Organization.create_hook()*
- /orgs/:owner/hooks/:id
  - GET: *github.Organization.Organization.get_hook()*
  - PATCH: *github.Organization.Organization.edit_hook()*
  - DELETE: *github.Organization.Organization.delete_hook()*
- /projects/:project_id
  - GET: *github.MainClass.Github.get_project()*
- /projects/:project_id/columns
  - GET: *github.Project.Project.get_columns()*
- /projects/columns/:column_id/cards
  - GET: *github.ProjectColumn.ProjectColumn.get_cards()*
  - POST: *github.ProjectColumn.ProjectColumn.create_card()*
- /rate_limit
  - GET: Not implemented, see *Github.rate_limiting*
- /reactions/:id
  - DELETE: *github.Reaction.Reaction.delete()*
- /repos/:owner/:repo

  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_repo()* or *github.MainClass.Github.get_repo()* or *github.NamedUser.NamedUser.get_repo()* or *github.Organization.Organization.get_repo()*
  - PATCH: *github.Repository.Repository.edit()*
  - DELETE: *github.Repository.Repository.delete()*
- /repos/:owner/:repo/:archive_format/:ref
  - GET: *github.Repository.Repository.get_archive_link()*
- /repos/:owner/:repo/assignees
  - GET: *github.Repository.Repository.get_assignees()*
- /repos/:owner/:repo/assignees/:assignee
  - GET: *github.Repository.Repository.has_in_assignees()*
- /repos/:owner/:repo/branches
  - GET: *github.Repository.Repository.get_branches()*
- /repos/:owner/:repo/branches/:branch
  - GET: *github.Repository.Repository.get_branch()*
- /repos/:owner/:repo/branches/:branch/protection
  - GET: *github.Branch.Branch.get_protection()*
  - PUT: *github.Branch.Branch.edit_protection()*
  - DELETE: *github.Branch.Branch.remove_protection()*
- /repos/:owner/:repo/branches/:branch/protection/enforce_admins
  - GET: *github.Branch.Branch.get_admin_enforcement()*
  - POST: *github.Branch.Branch.set_admin_enforcement()*
  - DELETE: *github.Branch.Branch.remove_admin_enforcement()*
- /repos/:owner/:repo/branches/:branch/protection/required_pull_request_reviews
  - GET: *github.Branch.Branch.get_required_pull_request_reviews()*
  - PATCH: *github.Branch.Branch.edit_required_pull_request_reviews()*
  - DELETE: *github.Branch.Branch.remove_required_pull_request_reviews()*
- /repos/:owner/:repo/branches/:branch/protection/required_signatures
  - GET: *github.Branch.Branch.get_required_signatures()*
  - POST: *github.Branch.Branch.add_required_signatures()*
  - DELETE: *github.Branch.Branch.remove_required_signatures()*
- /repos/:owner/:repo/branches/:branch/protection/required_status_checks
  - GET: *github.Branch.Branch.get_required_status_checks()*
  - PATCH: *github.Branch.Branch.edit_required_status_checks()*
  - DELETE: *github.Branch.Branch.remove_required_status_checks()*
- /repos/:owner/:repo/branches/:branch/protection/restrictions

  - POST: *github.Branch.Branch.edit_team_push_restrictions()* or *github.Branch.Branch.edit_user_push_restrictions()*
  - DELETE: *github.Branch.Branch.remove_push_restrictions()*
- /repos/:owner/:repo/branches/:branch/protection/restrictions/teams
  - GET: *github.Branch.Branch.get_team_push_restrictions()*
- /repos/:owner/:repo/branches/:branch/protection/restrictions/users
  - GET: *github.Branch.Branch.get_user_push_restrictions()*
- /repos/:owner/:repo/collaborators
  - GET: *github.Repository.Repository.get_collaborators()*
- /repos/:owner/:repo/collaborators/:user
  - GET: *github.Repository.Repository.has_in_collaborators()*
  - PUT: *github.Repository.Repository.add_to_collaborators()*
  - DELETE: *github.Repository.Repository.remove_from_collaborators()*
- /repos/:owner/:repo/collaborators/:username/permission
  - GET: *github.Repository.Repository.get_collaborator_permission()*
- /repos/:owner/:repo/comments
  - GET: *github.Repository.Repository.get_comments()*
- /repos/:owner/:repo/comments/:id
  - GET: *github.Repository.Repository.get_comment()*
  - PATCH: *github.CommitComment.CommitComment.edit()*
  - DELETE: *github.CommitComment.CommitComment.delete()*
- /repos/:owner/:repo/comments/:id/reactions
  - GET: *github.CommitComment.CommitComment.get_reactions()*
  - POST: *github.CommitComment.CommitComment.create_reaction()*
- /repos/:owner/:repo/commits
  - GET: *github.Repository.Repository.get_commits()*
- /repos/:owner/:repo/commits/:ref/status/
  - GET: *github.Commit.Commit.get_combined_status()*
- /repos/:owner/:repo/commits/:ref/status`
  - GET: *github.GitRef.GitRef.get_status()*
- /repos/:owner/:repo/commits/:ref/statuses`
  - GET: *github.GitRef.GitRef.get_statuses()*
- /repos/:owner/:repo/commits/:sha
  - GET: *github.Repository.Repository.get_commit()*
- /repos/:owner/:repo/commits/:sha/comments
  - GET: *github.Commit.Commit.get_comments()*

  - POST: *github.Commit.Commit.create_comment()*

- /repos/:owner/:repo/compare/:base...:head
  - GET: *github.Repository.Repository.compare()*
- /repos/:owner/:repo/contents/:path
  - GET: *github.Repository.Repository.get_contents()* or *github.Repository.Repository.get_dir_contents()*
  - PUT: *github.Repository.Repository.create_file()* or *github.Repository.Repository.update_file()*
  - DELETE: *github.Repository.Repository.delete_file()*
- /repos/:owner/:repo/contributors
  - GET: *github.Repository.Repository.get_contributors()*
- /repos/:owner/:repo/downloads
  - GET: *github.Repository.Repository.get_downloads()*
- /repos/:owner/:repo/downloads/:id
  - GET: *github.Repository.Repository.get_download()*
  - DELETE: *github.Download.Download.delete()*
- /repos/:owner/:repo/events
  - GET: *github.Repository.Repository.get_events()*
- /repos/:owner/:repo/forks
  - GET: *github.Repository.Repository.get_forks()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_fork()* or *github.Organization.Organization.create_fork()*
- /repos/:owner/:repo/git/blobs
  - POST: *github.Repository.Repository.create_git_blob()*
- /repos/:owner/:repo/git/blobs/:sha
  - GET: *github.Repository.Repository.get_git_blob()*
- /repos/:owner/:repo/git/commits
  - POST: *github.Repository.Repository.create_git_commit()*
- /repos/:owner/:repo/git/commits/:sha
  - GET: *github.Repository.Repository.get_git_commit()*
- /repos/:owner/:repo/git/refs
  - GET: *github.Repository.Repository.get_git_refs()*
  - POST: *github.Repository.Repository.create_git_ref()*
- /repos/:owner/:repo/git/refs/:ref
  - GET: *github.Repository.Repository.get_git_ref()*
  - PATCH: *github.GitRef.GitRef.edit()*
  - DELETE: *github.GitRef.GitRef.delete()*

- /repos/:owner/:repo/git/tags
  - POST: *github.Repository.Repository.create_git_tag()*
- /repos/:owner/:repo/git/tags/:sha
  - GET: *github.Repository.Repository.get_git_tag()*
- /repos/:owner/:repo/git/trees
  - POST: *github.Repository.Repository.create_git_tree()*
- /repos/:owner/:repo/git/trees/:sha
  - GET: *github.Repository.Repository.get_git_tree()*
- /repos/:owner/:repo/hooks
  - GET: *github.Repository.Repository.get_hooks()*
  - POST: *github.Repository.Repository.create_hook()*
- /repos/:owner/:repo/hooks/:id
  - GET: *github.Repository.Repository.get_hook()*
  - PATCH: *github.Hook.Hook.edit()*
  - DELETE: *github.Hook.Hook.delete()*
- /repos/:owner/:repo/hooks/:id/pings
  - POST: *github.Hook.Hook.ping()*
- /repos/:owner/:repo/hooks/:id/tests
  - POST: *github.Hook.Hook.test()*
- /repos/:owner/:repo/import
  - GET: *github.Repository.Repository.get_source_import()*
  - PUT: *github.Repository.Repository.create_source_import()*
- /repos/:owner/:repo/installation
  - GET: *github.MainClass.Github.get_installation()*
- /repos/:owner/:repo/issues
  - GET: *github.Repository.Repository.get_issues()*
  - POST: *github.Repository.Repository.create_issue()*
- /repos/:owner/:repo/issues/:issue_number/events
  - GET: *github.Issue.Issue.get_events()*
- /repos/:owner/:repo/issues/:issue_number/lock
  - PUT: *github.Issue.Issue.lock()*
  - DELETE: *github.Issue.Issue.unlock()*
- /repos/:owner/:repo/issues/:number
  - GET: *github.PullRequest.PullRequest.as_issue()* or *github.Repository.Repository.get_issue()*
  - PATCH: *github.Issue.Issue.edit()*

- /repos/:owner/:repo/issues/:number/assignees
  - POST: *github.Issue.Issue.add_to_assignees()*
  - DELETE: *github.Issue.Issue.remove_from_assignees()*
- /repos/:owner/:repo/issues/:number/comments
  - GET: *github.Issue.Issue.get_comments()* or *github.PullRequest.PullRequest.get_issue_comments()*
  - POST: *github.Issue.Issue.create_comment()* or *github.PullRequest.PullRequest.create_issue_comment()*
- /repos/:owner/:repo/issues/:number/labels
  - GET: *github.Issue.Issue.get_labels()* or *github.PullRequest.PullRequest.get_labels()*
  - POST: *github.Issue.Issue.add_to_labels()* or *github.PullRequest.PullRequest.add_to_labels()*
  - PUT: *github.Issue.Issue.set_labels()* or *github.PullRequest.PullRequest.set_labels()*
  - DELETE: *github.Issue.Issue.delete_labels()* or *github.PullRequest.PullRequest.delete_labels()*
- /repos/:owner/:repo/issues/:number/labels/:name
  - DELETE: *github.Issue.Issue.remove_from_labels()* or *github.PullRequest.PullRequest.remove_from_labels()*
- /repos/:owner/:repo/issues/:number/reactions
  - GET: *github.Issue.Issue.get_reactions()*
  - POST: *github.Issue.Issue.create_reaction()*
- /repos/:owner/:repo/issues/comments
  - GET: *github.Repository.Repository.get_issues_comments()*
- /repos/:owner/:repo/issues/comments/:id
  - GET: *github.Issue.Issue.get_comment()* or *github.PullRequest.PullRequest.get_issue_comment()*
  - PATCH: *github.IssueComment.IssueComment.edit()*
  - DELETE: *github.IssueComment.IssueComment.delete()*
- /repos/:owner/:repo/issues/comments/:id/reactions
  - GET: *github.IssueComment.IssueComment.get_reactions()*
  - POST: *github.IssueComment.IssueComment.create_reaction()*
- /repos/:owner/:repo/issues/events
  - GET: *github.Repository.Repository.get_issues_events()*
- /repos/:owner/:repo/issues/events/:id
  - GET: *github.Repository.Repository.get_issues_event()*
- /repos/:owner/:repo/keys
  - GET: *github.Repository.Repository.get_keys()*

  - POST: *github.Repository.Repository.create_key()*
- /repos/:owner/:repo/keys/:id
  - GET: *github.Repository.Repository.get_key()*
  - DELETE: *github.RepositoryKey.RepositoryKey.delete()*
- /repos/:owner/:repo/labels
  - GET: *github.Repository.Repository.get_labels()*
  - POST: *github.Repository.Repository.create_label()*
- /repos/:owner/:repo/labels/:name
  - GET: *github.Repository.Repository.get_label()*
  - PATCH: *github.Label.Label.edit()*
  - DELETE: *github.Label.Label.delete()*
- /repos/:owner/:repo/languages
  - GET: *github.Repository.Repository.get_languages()*
- /repos/:owner/:repo/license
  - GET: *github.Repository.Repository.get_license()*
- /repos/:owner/:repo/merges
  - POST: *github.Repository.Repository.merge()*
- /repos/:owner/:repo/milestones
  - GET: *github.Repository.Repository.get_milestones()*
  - POST: *github.Repository.Repository.create_milestone()*
- /repos/:owner/:repo/milestones/:number
  - GET: *github.Repository.Repository.get_milestone()*
  - PATCH: *github.Milestone.Milestone.edit()*
  - DELETE: *github.Milestone.Milestone.delete()*
- /repos/:owner/:repo/milestones/:number/labels
  - GET: *github.Milestone.Milestone.get_labels()*
- /repos/:owner/:repo/notifications
  - PUT: *github.Repository.Repository.mark_notifications_as_read()*
- /repos/:owner/:repo/projects
  - GET: *github.Repository.Repository.get_projects()*
- /repos/:owner/:repo/pulls
  - GET: *github.Repository.Repository.get_pulls()*
  - POST: *github.Repository.Repository.create_pull()*
- /repos/:owner/:repo/pulls/:number
  - GET: *github.Issue.Issue.as_pull_request()* or *github.ProjectCard.ProjectCard.get_content()* or *github.Repository.Repository.get_pull()*

  - PATCH: *github.PullRequest.PullRequest.edit()*

- /repos/:owner/:repo/pulls/:number/comments
  - GET: *github.PullRequest.PullRequest.get_comments()* or *github.PullRequest.PullRequest.get_review_comments()*
  - POST: *github.PullRequest.PullRequest.create_comment()* or *github.PullRequest.PullRequest.create_review_comment()*
- /repos/:owner/:repo/pulls/:number/commits
  - GET: *github.PullRequest.PullRequest.get_commits()*
- /repos/:owner/:repo/pulls/:number/files
  - GET: *github.PullRequest.PullRequest.get_files()*
- /repos/:owner/:repo/pulls/:number/merge
  - GET: *github.PullRequest.PullRequest.is_merged()*
  - PUT: *github.PullRequest.PullRequest.merge()*
- /repos/:owner/:repo/pulls/:number/requested_reviewers
  - GET: *github.PullRequest.PullRequest.get_review_requests()*
  - POST: *github.PullRequest.PullRequest.create_review_request()*
  - DELETE: *github.PullRequest.PullRequest.delete_review_request()*
- /repos/:owner/:repo/pulls/:number/review/:id/comments
  - GET: *github.PullRequest.PullRequest.get_single_review_comments()*
- /repos/:owner/:repo/pulls/:number/reviews
  - GET: *github.PullRequest.PullRequest.get_reviews()*
  - POST: *github.PullRequest.PullRequest.create_review()*
- /repos/:owner/:repo/pulls/:number/reviews/:id
  - GET: *github.PullRequest.PullRequest.get_review()*
- /repos/:owner/:repo/pulls/:number/reviews/:review_id/dismissals
  - PUT: *github.PullRequestReview.PullRequestReview.dismiss()*
- /repos/:owner/:repo/pulls/comments
  - GET: *github.Repository.Repository.get_pulls_comments()* or *github.Repository.Repository.get_pulls_review_comments()*
- /repos/:owner/:repo/pulls/comments/:number
  - GET: *github.PullRequest.PullRequest.get_comment()* or *github.PullRequest.PullRequest.get_review_comment()*
  - PATCH: *github.PullRequestComment.PullRequestComment.edit()*
  - DELETE: *github.PullRequestComment.PullRequestComment.delete()*
- /repos/:owner/:repo/pulls/comments/:number/reactions
  - GET: *github.PullRequestComment.PullRequestComment.get_reactions()*
  - POST: *github.PullRequestComment.PullRequestComment.create_reaction()*

- /repos/:owner/:repo/readme
  - GET: *github.Repository.Repository.get_readme()*
- /repos/:owner/:repo/releases
  - GET: *github.Repository.Repository.get_releases()*
  - POST: *github.Repository.Repository.create_git_release()*
- /repos/:owner/:repo/releases/:id
  - GET: *github.Repository.Repository.get_release()*
- /repos/:owner/:repo/releases/:release_id
  - PATCH: *github.GitRelease.GitRelease.update_release()*
  - DELETE: *github.GitRelease.GitRelease.delete_release()*
- /repos/:owner/:repo/releases/:release_id/assets
  - GET: *github.GitRelease.GitRelease.get_assets()*
- /repos/:owner/:repo/releases/latest
  - GET: *github.Repository.Repository.get_latest_release()*
- /repos/:owner/:repo/stargazers
  - GET: *github.Repository.Repository.get_stargazers_with_dates()* or *github.Repository.Repository.get_stargazers()*
- /repos/:owner/:repo/stats/code_frequency
  - GET: *github.Repository.Repository.get_stats_code_frequency()*
- /repos/:owner/:repo/stats/commit_activity
  - GET: *github.Repository.Repository.get_stats_commit_activity()*
- /repos/:owner/:repo/stats/contributors
  - GET: *github.Repository.Repository.get_stats_contributors()*
- /repos/:owner/:repo/stats/participation
  - GET: *github.Repository.Repository.get_stats_participation()*
- /repos/:owner/:repo/stats/punch_card
  - GET: *github.Repository.Repository.get_stats_punch_card()*
- /repos/:owner/:repo/statuses/:ref
  - GET: *github.Commit.Commit.get_statuses()*
- /repos/:owner/:repo/statuses/:sha
  - POST: *github.Commit.Commit.create_status()*
- /repos/:owner/:repo/subscribers
  - GET: *github.Repository.Repository.get_subscribers()*
- /repos/:owner/:repo/subscription
  - GET: *github.AuthenticatedUser.AuthenticatedUser.has_in_watched()*
  - PUT: *github.AuthenticatedUser.AuthenticatedUser.add_to_watched()*

  - DELETE: *github.AuthenticatedUser.AuthenticatedUser.remove_from_watched()*

- /repos/:owner/:repo/tags
  - GET: *github.Repository.Repository.get_tags()*
- /repos/:owner/:repo/teams
  - GET: *github.Repository.Repository.get_teams()*
- /repos/:owner/:repo/topics
  - GET: *github.Repository.Repository.get_topics()*
  - PUT: *github.Repository.Repository.replace_topics()*
- /repos/:owner/:repo/traffic/clones
  - GET: *github.Repository.Repository.get_clones_traffic()*
- /repos/:owner/:repo/traffic/popular/paths
  - GET: *github.Repository.Repository.get_top_paths()*
- /repos/:owner/:repo/traffic/popular/referrers
  - GET: *github.Repository.Repository.get_top_referrers()*
- /repos/:owner/:repo/traffic/views
  - GET: *github.Repository.Repository.get_views_traffic()*
- /repos/:owner/:repo/watchers
  - GET: *github.Repository.Repository.get_watchers()*
- /repositories
  - GET: *github.MainClass.Github.get_repos()*
- /repositories/:id
  - GET: *github.MainClass.Github.get_repo()*
- /search/code
  - GET: *github.MainClass.Github.search_code()*
- /search/commits
  - GET: *github.MainClass.Github.search_commits()*
- /search/issues
  - GET: *github.MainClass.Github.search_issues()*
- /search/repositories
  - GET: *github.MainClass.Github.search_repositories()*
- /search/topics
  - GET: *github.MainClass.Github.search_topics()*
- /search/users
  - GET: *github.MainClass.Github.search_users()*
- /teams/:id

  - GET: *github.Organization.Organization.get_team()*
  - PATCH: *github.Team.Team.edit()*
  - DELETE: *github.Team.Team.delete()*
- /teams/:id/members
  - GET: *github.Team.Team.get_members()*
- /teams/:id/members/:user
  - GET: *github.Team.Team.has_in_members()*
  - PUT: *github.Team.Team.add_to_members()*
  - DELETE: *github.Team.Team.remove_from_members()*
- /teams/:id/memberships/:user
  - PUT: *github.Team.Team.add_membership()*
- /teams/:id/repos
  - GET: *github.Team.Team.get_repos()*
- /teams/:id/repos/:org/:repo
  - PUT: *github.Team.Team.add_to_repos()* or *github.Team.Team.set_repo_permission()*
- /teams/:id/repos/:owner/:repo
  - GET: *github.Team.Team.has_in_repos()*
  - DELETE: *github.Team.Team.remove_from_repos()*
- /teams/:team_id/memberships/:username
  - DELETE: *github.Team.Team.remove_membership()*
- /user
  - GET: *github.MainClass.Github.get_user()*
  - PATCH: *github.AuthenticatedUser.AuthenticatedUser.edit()*
- /user/emails
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_emails()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.add_to_emails()*
  - DELETE: *github.AuthenticatedUser.AuthenticatedUser.remove_from_emails()*
- /user/followers
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_followers()*
- /user/following
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_following()*
- /user/following/:user
  - GET: *github.AuthenticatedUser.AuthenticatedUser.has_in_following()*
  - PUT: *github.AuthenticatedUser.AuthenticatedUser.add_to_following()*
  - DELETE: *github.AuthenticatedUser.AuthenticatedUser.remove_from_following()*

- /user/issues
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_user_issues()*
- /user/keys
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_keys()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_key()*
- /user/keys/:id
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_key()*
  - DELETE: *github.UserKey.UserKey.delete()*
- /user/migrations/:migration_id/archive`_
  - GET: *github.Migration.Migration.get_archive_url()*
  - DELETE: *github.Migration.Migration.delete()*
- /user/migrations/:migration_id/repos/:repo_name/lock`_
  - DELETE: *github.Migration.Migration.unlock_repo()*
- /user/migrations/:migration_id`_
  - GET: *github.Migration.Migration.get_status()*
- /user/migrations`_
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_migrations()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_migration()*
- /user/orgs
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_orgs()*
- /user/repos
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_repos()*
  - POST: *github.AuthenticatedUser.AuthenticatedUser.create_repo()*
- /user/repository_invitations/:invitation_id
  - PATCH: *github.AuthenticatedUser.AuthenticatedUser.accept_invitation()*
- /user/starred
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_starred()*
- /user/starred/:owner/:repo
  - GET: *github.AuthenticatedUser.AuthenticatedUser.has_in_starred()*
  - PUT: *github.AuthenticatedUser.AuthenticatedUser.add_to_starred()*
  - DELETE: *github.AuthenticatedUser.AuthenticatedUser.remove_from_starred()*
- /user/subscriptions
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_subscriptions()* or *github.AuthenticatedUser.AuthenticatedUser.get_watched()*
- /user/subscriptions/:owner/:repo
  - GET: *github.AuthenticatedUser.AuthenticatedUser.has_in_subscriptions()*

  - PUT: *github.AuthenticatedUser.AuthenticatedUser.add_to_subscriptions()*
  - DELETE: *github.AuthenticatedUser.AuthenticatedUser.remove_from_subscriptions()*
- /user/teams
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_teams()*
- /users
  - GET: *github.MainClass.Github.get_users()*
- /users/:user
  - GET: *github.MainClass.Github.get_user()*
- /users/:user/events
  - GET: *github.NamedUser.NamedUser.get_events()*
- /users/:user/events/orgs/:org
  - GET: *github.AuthenticatedUser.AuthenticatedUser.get_organization_events()*
- /users/:user/events/public
  - GET: *github.NamedUser.NamedUser.get_public_events()*
- /users/:user/followers
  - GET: *github.NamedUser.NamedUser.get_followers()*
- /users/:user/following
  - GET: *github.NamedUser.NamedUser.get_following()*
- /users/:user/following/:target_user
  - GET: *github.NamedUser.NamedUser.has_in_following()*
- /users/:user/gists
  - GET: *github.NamedUser.NamedUser.get_gists()*
- /users/:user/keys
  - GET: *github.NamedUser.NamedUser.get_keys()*
- /users/:user/orgs
  - GET: *github.NamedUser.NamedUser.get_orgs()*
- /users/:user/received_events
  - GET: *github.NamedUser.NamedUser.get_received_events()*
- /users/:user/received_events/public
  - GET: *github.NamedUser.NamedUser.get_public_received_events()*
- /users/:user/repos
  - GET: *github.NamedUser.NamedUser.get_repos()*
- /users/:user/starred
  - GET: *github.NamedUser.NamedUser.get_starred()*
- /users/:user/subscriptions

  - GET: *github.NamedUser.NamedUser.get_subscriptions()*

- `/users/:user/watched`
  - GET: *github.NamedUser.NamedUser.get_watched()*
- `https://<upload_url>/repos/:owner/:repo/releases/:release_id/assets`
  - POST: *github.GitRelease.GitRelease.upload_asset()*

3.3 Utilities

3.3.1 Logging

`github.`**`enable_console_debug_logging`**`()`
This function sets up a very simple logging configuration (log everything on standard output) that is useful for troubleshooting.

3.3.2 Error Handling

exception `github.GithubException.`**`GithubException`**(*status*, *data*)
Error handling in PyGithub is done with exceptions. This class is the base of all exceptions raised by PyGithub (but *github.GithubException.BadAttributeException*).

Some other types of exceptions might be raised by underlying libraries, for example for network-related issues.

`status`
The status returned by the Github API

`data`
The (decoded) data returned by the Github API

exception `github.GithubException.`**`BadCredentialsException`**(*status*, *data*)
Exception raised in case of bad credentials (when Github API replies with a 401 or 403 HTML status)

exception `github.GithubException.`**`UnknownObjectException`**(*status*, *data*)
Exception raised when a non-existing object is requested (when Github API replies with a 404 HTML status)

exception `github.GithubException.`**`BadUserAgentException`**(*status*, *data*)
Exception raised when request is sent with a bad user agent header (when Github API replies with a 403 bad user agent HTML status)

exception `github.GithubException.`**`RateLimitExceededException`**(*status*, *data*)
Exception raised when the rate limit is exceeded (when Github API replies with a 403 rate limit exceeded HTML status)

exception `github.GithubException.`**`BadAttributeException`**(*actualValue*, *expectedType*, *transformationException*)
Exception raised when Github returns an attribute with the wrong type.

`actual_value`
The value returned by Github

`expected_type`
The type PyGithub expected

`transformation_exception`
The exception raised when PyGithub tried to parse the value

exception github.GithubException.**TwoFactorException**(*status*, *data*)
Exception raised when Github requires a onetime password for two-factor authentication

3.3.3 Default argument

`github.NotSet` is a special value for arguments you don't want to provide. You should not have to manipulate it directly, because it's the default value of all parameters accepting it. Just note that it is different from `None`, which is an allowed value for some parameters.

3.3.4 Pagination

class github.PaginatedList.**PaginatedList**
This class abstracts the pagination of the API.

You can simply enumerate through instances of this class:

```
for repo in user.get_repos():
    print(repo.name)
```

If you want to know the total number of items in the list:

```
print(user.get_repos().totalCount)
print(len(user.get_repos()))
```

You can also index them or take slices:

```
second_repo = user.get_repos()[1]
first_repos = user.get_repos()[:10]
```

If you want to iterate in reversed order, just do:

```
for repo in user.get_repos().reversed:
    print(repo.name)
```

And if you really need it, you can explicitly access a specific page:

```
some_repos = user.get_repos().get_page(0)
some_other_repos = user.get_repos().get_page(3)
```

3.3.5 Input classes

class github.InputFileContent.**InputFileContent**(*content*, *new_name=NotSet*)
This class represents InputFileContents

Parameters

- **content** – string
- **new_name** – string

class github.InputGitAuthor.**InputGitAuthor**(*name*, *email*, *date=NotSet*)
This class represents InputGitAuthors

Parameters

- **name** – string

- **email** – string
- **date** – string

class github.InputGitTreeElement.**InputGitTreeElement**(*path*, *mode*, *type*, *content=NotSet*, *sha=NotSet*)

This class represents InputGitTreeElements

Parameters

- **path** – string
- **mode** – string
- **type** – string
- **content** – string
- **sha** – string

3.4 Github objects

class github.GithubObject.**GithubObject**

Base class for all classes representing objects returned by the API.

raw_data

Type dict

raw_headers

Type dict

etag

Type str

last_modified

Type str

get__repr__(*params*)

Converts the object to a nicely printable string.

3.4.1 AuthenticatedUser

class github.AuthenticatedUser.**AuthenticatedUser**

This class represents AuthenticatedUsers as returned by https://developer.github.com/v3/users/#get-the-authenticated-user

An AuthenticatedUser object can be created by calling `get_user()` on a Github object.

avatar_url

Type string

bio

Type string

blog

Type string

collaborators

Type integer

company

Type string

created_at

Type datetime.datetime

disk_usage

Type integer

email

Type string

events_url

Type string

followers

Type integer

followers_url

Type string

following

Type integer

following_url

Type string

gists_url

Type string

gravatar_id

Type string

hireable

Type bool

html_url

Type string

id

Type integer

location

Type string

login

Type string

name

Type string

node_id

Type string

organizations_url

Type string

owned_private_repos

Type integer

plan

Type *github.Plan.Plan*

private_gists

Type integer

public_gists

Type integer

public_repos

Type integer

received_events_url

Type string

repos_url

Type string

site_admin

Type bool

starred_url

Type string

subscriptions_url

Type string

total_private_repos

Type integer

type

Type string

updated_at

Type datetime.datetime

url

Type string

add_to_emails(**emails*)

Calls POST /user/emails

Parameters **email** – string

Return type None

add_to_following(*following*)

Calls PUT /user/following/:user

Parameters **following** – *github.NamedUser.NamedUser*

Return type None

add_to_starred(*starred*)

Calls PUT /user/starred/:owner/:repo

Parameters **starred** – *github.Repository.Repository*

Return type None

add_to_subscriptions(*subscription*)

Calls PUT /user/subscriptions/:owner/:repo

Parameters **subscription** – *github.Repository.Repository*

Return type None

add_to_watched(*watched*)

Calls PUT /repos/:owner/:repo/subscription

Parameters **watched** – *github.Repository.Repository*

Return type None

create_authorization(*scopes=NotSet*, *note=NotSet*, *note_url=NotSet*, *client_id=NotSet*, *client_secret=NotSet*, *onetime_password=None*)

Calls POST /authorizations

Parameters

- **scopes** – list of string
- **note** – string
- **note_url** – string
- **client_id** – string
- **client_secret** – string
- **onetime_password** – string

Return type *github.Authorization.Authorization*

create_fork(*repo*)

Calls POST /repos/:owner/:repo/forks

Parameters **repo** – *github.Repository.Repository*

Return type *github.Repository.Repository*

create_gist(*public*, *files*, *description=NotSet*)

Calls POST /gists

Parameters

- **public** – bool
- **files** – dict of string to *github.InputFileContent.InputFileContent*
- **description** – string

Return type *github.Gist.Gist*

create_key(*title*, *key*)

Calls POST /user/keys

Parameters

- **title** – string
- **key** – string

Return type *github.UserKey.UserKey*

create_repo(*name*, *description=NotSet*, *homepage=NotSet*, *private=NotSet*, *has_issues=NotSet*, *has_wiki=NotSet*, *has_downloads=NotSet*, *has_projects=NotSet*, *auto_init=NotSet*, *license_template=NotSet*, *gitignore_template=NotSet*, *allow_squash_merge=NotSet*, *allow_merge_commit=NotSet*, *allow_rebase_merge=NotSet*)

Calls POST /user/repos

Parameters

- **name** – string
- **description** – string
- **homepage** – string
- **private** – bool
- **has_issues** – bool
- **has_wiki** – bool
- **has_downloads** – bool
- **has_projects** – bool
- **auto_init** – bool
- **license_template** – string
- **gitignore_template** – string
- **allow_squash_merge** – bool
- **allow_merge_commit** – bool
- **allow_rebase_merge** – bool

Return type *github.Repository.Repository*

edit(*name=NotSet*, *email=NotSet*, *blog=NotSet*, *company=NotSet*, *location=NotSet*, *hireable=NotSet*, *bio=NotSet*)

Calls PATCH /user

Parameters

- **name** – string
- **email** – string
- **blog** – string
- **company** – string
- **location** – string
- **hireable** – bool

- **bio** – string

Return type None

get_authorization(*id*)

Calls GET /authorizations/:id

Parameters **id** – integer

Return type *github.Authorization.Authorization*

get_authorizations()

Calls GET /authorizations

Return type *github.PaginatedList.PaginatedList* of *github.Authorization.Authorization*

get_emails()

Calls GET /user/emails

Return type list of string

get_events()

Calls GET /events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_followers()

Calls GET /user/followers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_following()

Calls GET /user/following

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_gists(*since=NotSet*)

Calls GET /gists

Parameters **since** – datetime.datetime format YYYY-MM-DDTHH:MM:SSZ

Return type *github.PaginatedList.PaginatedList* of *github.Gist.Gist*

get_issues(*filter=NotSet*, *state=NotSet*, *labels=NotSet*, *sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /issues

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

Parameters

- **filter** – string
- **state** – string
- **labels** – list of *github.Label.Label*
- **sort** – string
- **direction** – string

- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

get_user_issues(*filter=NotSet*, *state=NotSet*, *labels=NotSet*, *sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /user/issues

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

Parameters

- **filter** – string
- **state** – string
- **labels** – list of *github.Label.Label*
- **sort** – string
- **direction** – string
- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

get_key(*id*)

Calls GET /user/keys/:id

Parameters **id** – integer

Return type *github.UserKey.UserKey*

get_keys()

Calls GET /user/keys

Return type *github.PaginatedList.PaginatedList* of *github.UserKey.UserKey*

get_notification(*id*)

Calls GET /notifications/threads/:id

Return type *github.Notification.Notification*

get_notifications(*all=NotSet*, *participating=NotSet*, *since=NotSet*, *before=NotSet*)

Calls GET /notifications

Parameters

- **all** – bool
- **participating** – bool
- **since** – datetime.datetime
- **before** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.Notification.Notification*

get_organization_events(*org*)

Calls GET /users/:user/events/orgs/:org

Parameters **org** – *github.Organization.Organization*

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_orgs()

Calls GET /user/orgs

Return type *github.PaginatedList.PaginatedList* of *github.Organization.Organization*

get_repo(*name*)

Calls GET /repos/:owner/:repo

Parameters **name** – string

Return type *github.Repository.Repository*

get_repos(*visibility=NotSet*, *affiliation=NotSet*, *type=NotSet*, *sort=NotSet*, *direction=NotSet*)

Calls *GET /user/repos <http://developer.github.com/v3/repos>*

Parameters

- **visibility** – string
- **affiliation** – string
- **type** – string
- **sort** – string
- **direction** – string

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_starred()

Calls GET /user/starred

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_starred_gists()

Calls GET /gists/starred

Return type *github.PaginatedList.PaginatedList* of *github.Gist.Gist*

get_subscriptions()

Calls GET /user/subscriptions

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_teams()

Calls GET /user/teams

Return type *github.PaginatedList.PaginatedList* of *github.Team.Team*

get_watched()

Calls GET /user/subscriptions

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

has_in_following(*following*)

Calls GET /user/following/:user

Parameters **following** – *github.NamedUser.NamedUser*

Return type bool

has_in_starred(*starred*)

Calls GET /user/starred/:owner/:repo

Parameters **starred** – *github.Repository.Repository*

Return type bool

has_in_subscriptions(*subscription*)

Calls GET /user/subscriptions/:owner/:repo

Parameters **subscription** – *github.Repository.Repository*

Return type bool

has_in_watched(*watched*)

Calls GET /repos/:owner/:repo/subscription

Parameters **watched** – *github.Repository.Repository*

Return type bool

mark_notifications_as_read(*last_read_at=datetime.datetime(2019, 6, 24, 7, 15, 21, 309524)*)

Calls PUT /notifications

Parameters **last_read_at** – datetime

remove_from_emails(**emails*)

Calls DELETE /user/emails

Parameters **email** – string

Return type None

remove_from_following(*following*)

Calls DELETE /user/following/:user

Parameters **following** – *github.NamedUser.NamedUser*

Return type None

remove_from_starred(*starred*)

Calls DELETE /user/starred/:owner/:repo

Parameters **starred** – *github.Repository.Repository*

Return type None

remove_from_subscriptions(*subscription*)

Calls DELETE /user/subscriptions/:owner/:repo

Parameters **subscription** – *github.Repository.Repository*

Return type None

remove_from_watched(*watched*)

Calls DELETE /repos/:owner/:repo/subscription

Parameters **watched** – *github.Repository.Repository*

Return type None

accept_invitation(*invitation*)

Calls *PATCH /user/repository_invitations/:invitation_id <https://developer.github.com/v3/repos/invitations/>*

Parameters **invitation** – *github.Invitation.Invitation* or int

Return type None

create_migration(*repos*, *lock_repositories=NotSet*, *exclude_attachments=NotSet*)

Calls **'POST /user/migrations'_**

Parameters

- **repos** – list or tuple of str
- **lock_repositories** – bool
- **exclude_attachments** – bool

Return type *github.Migration.Migration*

get_migrations()

Calls **'GET /user/migrations'_**

Return type *github.PaginatedList.PaginatedList* of *github.Migration.Migration*

3.4.2 Authorization

class github.Authorization.**Authorization**

This class represents Authorizations. The reference can be found here https://developer.github.com/v3/oauth_authorizations/

app

Type *github.AuthorizationApplication.AuthorizationApplication*

created_at

Type datetime.datetime

id

Type integer

note

Type string

note_url

Type string

scopes

Type list of string

token

Type string

updated_at

Type datetime.datetime

url

Type string

delete()

Calls DELETE /authorizations/:id

Return type None

edit(*scopes=NotSet*, *add_scopes=NotSet*, *remove_scopes=NotSet*, *note=NotSet*, *note_url=NotSet*)

Calls PATCH /authorizations/:id

Parameters

- **scopes** – list of string
- **add_scopes** – list of string
- **remove_scopes** – list of string
- **note** – string
- **note_url** – string

Return type None

3.4.3 AuthorizationApplication

class github.AuthorizationApplication.**AuthorizationApplication**

This class represents AuthorizationApplications

name

Type string

url

Type string

3.4.4 Branch

class github.Branch.**Branch**

This class represents Branches. The reference can be found here https://developer.github.com/v3/repos/branches

commit

Type *github.Commit.Commit*

name

Type string

protected

Type bool

protection_url

Type string

get_protection()

Calls GET /repos/:owner/:repo/branches/:branch/protection

edit_protection(*strict=NotSet*, *contexts=NotSet*, *enforce_admins=NotSet*, *dismissal_users=NotSet*, *dismissal_teams=NotSet*, *dismiss_stale_reviews=NotSet*, *require_code_owner_reviews=NotSet*, *required_approving_review_count=NotSet*, *user_push_restrictions=NotSet*, *team_push_restrictions=NotSet*)

Calls PUT /repos/:owner/:repo/branches/:branch/protection

Strict bool

Contexts list of strings

Enforce_admins bool

Dismissal_users list of strings

Dismissal_teams list of strings

Dismiss_stale_reviews bool

Require_code_owner_reviews bool

Required_approving_review_count int

User_push_restrictions list of strings

Team_push_restrictions list of strings

NOTE: The GitHub API groups strict and contexts together, both must be submitted. Take care to pass both as arguments even if only one is changing. Use edit_required_status_checks() to avoid this.

remove_protection()

Calls DELETE /repos/:owner/:repo/branches/:branch/protection

get_required_status_checks()

Calls GET /repos/:owner/:repo/branches/:branch/protection/required_status_checks

Return type *github.RequiredStatusChecks.RequiredStatusChecks*

edit_required_status_checks(*strict=NotSet*, *contexts=NotSet*)

Calls PATCH /repos/:owner/:repo/branches/:branch/protection/required_status_checks

Strict bool

Contexts list of strings

remove_required_status_checks()

Calls DELETE /repos/:owner/:repo/branches/:branch/protection/required_status_checks

get_required_pull_request_reviews()

Calls GET /repos/:owner/:repo/branches/:branch/protection/required_pull_request_reviews

Return type *github.RequiredPullRequestReviews.RequiredPullRequestReviews*

edit_required_pull_request_reviews(*dismissal_users=NotSet*, *dismissal_teams=NotSet*, *dismiss_stale_reviews=NotSet*, *require_code_owner_reviews=NotSet*, *required_approving_review_count=NotSet*)

Calls PATCH /repos/:owner/:repo/branches/:branch/protection/required_pull_request_reviews

Dismissal_users list of strings

Dismissal_teams list of strings

Dismiss_stale_reviews bool

Require_code_owner_reviews bool

Required_approving_review_count int

remove_required_pull_request_reviews()

Calls DELETE /repos/:owner/:repo/branches/:branch/protection/required_pull_request_reviews

get_admin_enforcement()

Calls GET /repos/:owner/:repo/branches/:branch/protection/enforce_admins

Return type bool

set_admin_enforcement()

Calls POST /repos/:owner/:repo/branches/:branch/protection/enforce_admins

remove_admin_enforcement()

Calls DELETE /repos/:owner/:repo/branches/:branch/protection/enforce_admins

get_user_push_restrictions()

Calls GET /repos/:owner/:repo/branches/:branch/protection/restrictions/users

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_team_push_restrictions()

Calls GET /repos/:owner/:repo/branches/:branch/protection/restrictions/teams

Return type *github.PaginatedList.PaginatedList* of *github.Team.Team*

edit_user_push_restrictions(**users*)

Calls POST /repos/:owner/:repo/branches/:branch/protection/restrictions

Users list of strings

edit_team_push_restrictions(**teams*)

Calls POST /repos/:owner/:repo/branches/:branch/protection/restrictions

Teams list of strings

remove_push_restrictions()

Calls DELETE /repos/:owner/:repo/branches/:branch/protection/restrictions

get_required_signatures()

Calls *GET /repos/:owner/:repo/branches/:branch/protection/required_signatures <https://developer.github.com/v3/repos/branches>*

add_required_signatures()

Calls *POST /repos/:owner/:repo/branches/:branch/protection/required_signatures <https://developer.github.com/v3/repos/branches>*

remove_required_signatures()

Calls *DELETE /repos/:owner/:repo/branches/:branch/protection/required_signatures <https://developer.github.com/v3/repos/branches>*

3.4.5 BranchProtection

class github.BranchProtection.**BranchProtection**

This class represents Branch Protection. The reference can be found here https://developer.github.com/v3/repos/branches/#get-branch-protection

url

Type string

required_status_checks

Type *github.RequiredStatusChecks.RequiredStatusChecks*

enforce_admins

Type bool

required_pull_request_reviews

Type *github.RequiredPullRequestReviews.RequiredPullRequestReviews*

get_user_push_restrictions()

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_team_push_restrictions()

Return type *github.PaginatedList.PaginatedList* of *github.Team.Team*

3.4.6 Clones

class github.Clones.**Clones**

This class represents a popular Path for a GitHub repository. The reference can be found here https://developer.github.com/v3/repos/traffic/

timestamp

Type datetime.datetime

count

Type integer

uniques

Type integer

3.4.7 Commit

class github.Commit.**Commit**

This class represents Commits. The reference can be found here http://developer.github.com/v3/git/commits/

author

Type *github.NamedUser.NamedUser*

comments_url

Type string

commit

Type *github.GitCommit.GitCommit*

committer

Type *github.NamedUser.NamedUser*

files

Type list of *github.File.File*

html_url

Type string

parents

Type list of *github.Commit.Commit*

sha

Type string

stats

Type *github.CommitStats.CommitStats*

url

Type string

create_comment(*body*, *line=NotSet*, *path=NotSet*, *position=NotSet*)

Calls POST /repos/:owner/:repo/commits/:sha/comments

Parameters

- **body** – string
- **line** – integer
- **path** – string
- **position** – integer

Return type *github.CommitComment.CommitComment*

create_status(*state*, *target_url=NotSet*, *description=NotSet*, *context=NotSet*)

Calls POST /repos/:owner/:repo/statuses/:sha

Parameters

- **state** – string
- **target_url** – string
- **description** – string
- **context** – string

Return type *github.CommitStatus.CommitStatus*

get_comments()

Calls GET /repos/:owner/:repo/commits/:sha/comments

Return type *github.PaginatedList.PaginatedList* of *github.CommitComment.CommitComment*

get_statuses()

Calls GET /repos/:owner/:repo/statuses/:ref

Return type *github.PaginatedList.PaginatedList* of *github.CommitStatus.CommitStatus*

get_combined_status()

Calls GET /repos/:owner/:repo/commits/:ref/status/

Return type *github.CommitCombinedStatus.CommitCombinedStatus*

3.4.8 CommitCombinedStatus

class github.CommitCombinedStatus.**CommitCombinedStatus**

This class represents CommitCombinedStatuses. The reference can be found here https://developer.github.com/v3/repos/statuses/#get-the-combined-status-for-a-specific-ref

state

Type string

sha

Type string

total_count

Type integer

commit_url

Type string

url

Type string

repository

Type *github.Repository.Repository*

statuses

Type list of CommitStatus

3.4.9 CommitComment

class github.CommitComment.**CommitComment**

This class represents CommitComments. The reference can be found here https://developer.github.com/v3/repos/comments/

body

Type string

commit_id

Type string

created_at

Type datetime.datetime

html_url

Type string

id

Type integer

line

Type integer

path

Type string

position

Type integer

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

delete()

Calls DELETE /repos/:owner/:repo/comments/:id

Return type None

edit(*body*)

Calls PATCH /repos/:owner/:repo/comments/:id

Parameters **body** – string

Return type None

get_reactions()

Calls GET /repos/:owner/:repo/comments/:id/reactions

Returns

class *github.PaginatedList.PaginatedList* of *github.Reaction.Reaction*

create_reaction(*reaction_type*)

Calls POST /repos/:owner/:repo/comments/:id/reactions

Parameters **reaction_type** – string

Return type *github.Reaction.Reaction*

3.4.10 CommitStats

class github.CommitStats.**CommitStats**
This class represents CommitStatses.

additions

Type integer

deletions

Type integer

total

Type integer

3.4.11 CommitStatus

class github.CommitStatus.**CommitStatus**
This class represents CommitStatuses.The reference can be found here https://developer.github.com/v3/repos/statuses/

created_at

Type datetime.datetime

creator

Type *github.NamedUser.NamedUser*

description

Type string

id

Type integer

state

Type string

context

Type string

target_url

Type string

updated_at

Type datetime.datetime

url

Type string

3.4.12 Comparison

class github.Comparison.**Comparison**
This class represents Comparisons

ahead_by

Type integer

base_commit

Type *github.Commit.Commit*

behind_by

Type integer

commits

Type list of *github.Commit.Commit*

diff_url

Type string

files

Type list of *github.File.File*

html_url

Type string

merge_base_commit

Type *github.Commit.Commit*

patch_url

Type string

permalink_url

Type string

status

Type string

total_commits

Type integer

url

Type string

3.4.13 ContentFile

class github.ContentFile.**ContentFile**

This class represents ContentFiles. The reference can be found here https://developer.github.com/v3/repos/contents/#get-contents

content

Type string

download_url

Type string

encoding

Type string

git_url

Type string

html_url

Type string

license

Type *github.License.License*

name

Type string

path

Type string

repository

Type *github.Repository.Repository*

sha

Type string

size

Type integer

type

Type string

url

Type string

text_matches

Type string

3.4.14 Download

class github.Download.**Download**

This class represents Downloads. The reference can be found here https://developer.github.com/v3/repos/downloads/

accesskeyid

Type string

acl

Type string

bucket

Type string

content_type

Type string

created_at

Type datetime.datetime

description

Type string

download_count

Type integer

expirationdate

Type datetime.datetime

html_url

Type string

id

Type integer

mime_type

Type string

name

Type string

path

Type string

policy

Type string

prefix

Type string

redirect

Type bool

s3_url

Type string

signature

Type string

size

Type integer

url

Type string

delete()

Calls DELETE /repos/:owner/:repo/downloads/:id

Return type None

3.4.15 Event

class github.Event.**Event**

This class represents Events. The reference can be found here http://developer.github.com/v3/activity/events/

actor

Type *github.NamedUser.NamedUser*

created_at

Type datetime.datetime

id

Type string

org

Type *github.Organization.Organization*

payload

Type dict

public

Type bool

repo

Type *github.Repository.Repository*

type

Type string

3.4.16 File

class github.File.**File**

This class represents Files

additions

Type integer

blob_url

Type string

changes

Type integer

contents_url

Type string

deletions

Type integer

filename

Type string

patch

Type string

previous_filename

Type string

raw_url

Type string

sha

Type string

status

Type string

3.4.17 Gist

class github.Gist.**Gist**

This class represents Gists. The reference can be found here https://developer.github.com/v3/gists/

comments

Type integer

comments_url

Type string

commits_url

Type string

created_at

Type datetime.datetime

description

Type string

files

Type dict of string to *github.GistFile.GistFile*

fork_of

Type *github.Gist.Gist*

forks

Type list of *github.Gist.Gist*

forks_url

Type string

git_pull_url

Type string

git_push_url

Type string

history

Type list of *github.GistHistoryState.GistHistoryState*

html_url

Type string

id

Type string

owner

Type *github.NamedUser.NamedUser*

public

Type bool

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

create_comment(*body*)

Calls POST /gists/:gist_id/comments

Parameters **body** – string

Return type *github.GistComment.GistComment*

create_fork()

Calls POST /gists/:id/forks

Return type *github.Gist.Gist*

delete()

Calls DELETE /gists/:id

Return type None

edit(*description=NotSet*, *files=NotSet*)

Calls PATCH /gists/:id

Parameters

- **description** – string
- **files** – dict of string to *github.InputFileContent.InputFileContent*

Return type None

get_comment(*id*)

Calls GET /gists/:gist_id/comments/:id

Parameters **id** – integer

Return type *github.GistComment.GistComment*

get_comments()

Calls GET /gists/:gist_id/comments

Return type *github.PaginatedList.PaginatedList* of *github.GistComment.GistComment*

is_starred()

Calls GET /gists/:id/star

Return type bool

reset_starred()

Calls DELETE /gists/:id/star

Return type None

set_starred()

Calls PUT /gists/:id/star

Return type None

3.4.18 GistComment

class github.GistComment.**GistComment**

This class represents GistComments. The reference can be found here https://developer.github.com/v3/gists/comments/

body

Type string

created_at

Type datetime.datetime

id

Type integer

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

delete()

Calls DELETE /gists/:gist_id/comments/:id

Return type None

edit(*body*)

Calls PATCH /gists/:gist_id/comments/:id

Parameters **body** – string

Return type None

3.4.19 GistFile

class github.GistFile.**GistFile**
This class represents GistFiles

content

Type string

filename

Type string

language

Type string

raw_url

Type string

size

Type integer

type

Type string

3.4.20 GistHistoryState

class github.GistHistoryState.**GistHistoryState**
This class represents GistHistoryStates

change_status

Type *github.CommitStats.CommitStats*

comments

Type integer

comments_url

Type string

commits_url

Type string

committed_at

Type datetime.datetime

created_at

Type datetime.datetime

description

Type string

files

Type dict of string to *github.GistFile.GistFile*

forks

Type list of *github.Gist.Gist*

forks_url

Type string

git_pull_url

Type string

git_push_url

Type string

history

Type list of *GistHistoryState*

html_url

Type string

id

Type string

owner

Type *github.NamedUser.NamedUser*

public

Type bool

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

version

Type string

3.4.21 GitAuthor

class github.GitAuthor.**GitAuthor**

This class represents GitAuthors

date

Type datetime.datetime

email

Type string

name

Type string

3.4.22 GitBlob

class github.GitBlob.**GitBlob**

This class represents GitBlobs. The reference can be found here https://developer.github.com/v3/git/blobs/

content

Type string

encoding

Type string

sha

Type string

size

Type integer

url

Type string

3.4.23 GitCommit

class github.GitCommit.**GitCommit**

This class represents GitCommits. The reference can be found here https://developer.github.com/v3/git/commits/

author

Type *github.GitAuthor.GitAuthor*

committer

Type *github.GitAuthor.GitAuthor*

html_url

Type string

message

Type string

parents

Type list of *github.GitCommit.GitCommit*

sha

Type string

tree

Type *github.GitTree.GitTree*

url

Type string

3.4.24 GitObject

class github.GitObject.**GitObject**

This class represents GitObjects

sha

Type string

type

Type string

url

Type string

3.4.25 GitRef

class github.GitRef.**GitRef**

This class represents GitRefs. The reference can be found here https://developer.github.com/v3/git/refs/

object

Type *github.GitObject.GitObject*

ref

Type string

url

Type string

delete()

Calls DELETE /repos/:owner/:repo/git/refs/:ref

Return type None

edit(*sha*, *force=NotSet*)

Calls PATCH /repos/:owner/:repo/git/refs/:ref

Parameters

- **sha** – string
- **force** – bool

Return type None

get_statuses()

https://developer.github.com/v3/repos/statuses/#list-statuses-for-a-specific-ref :calls: *GET /repos/:owner/:repo/commits/:ref/statuses* :return:

get_status()

https://developer.github.com/v3/repos/statuses/#get-the-combined-status-for-a-specific-ref :calls: *GET /repos/:owner/:repo/commits/:ref/status* :return:

3.4.26 GitRelease

class github.GitRelease.**GitRelease**

This class represents GitReleases. The reference can be found here https://developer.github.com/v3/repos/releases

id

Type integer

body

Type string

title

Type string

tag_name

Type string

target_commitish

Type string

draft

Type bool

prerelease

Type bool

author

Type *github.NamedUser.NamedUser*

created_at

Type datetime.datetime

published_at

Type datetime.datetime

url

Type string

upload_url

Type string

html_url

Type string

tarball_url

Type string

zipball_url

Type string

delete_release()

Calls DELETE /repos/:owner/:repo/releases/:release_id

Return type None

update_release(*name*, *message*, *draft=False*, *prerelease=False*, *tag_name=NotSet*, *target_commitish=NotSet*)

Calls PATCH /repos/:owner/:repo/releases/:release_id

Return type *github.GitRelease.GitRelease*

upload_asset(*path*, *label=''*, *content_type=NotSet*, *name=NotSet*)

Calls POST https://<upload_url>/repos/:owner/:repo/releases/:release_id/assets

Return type *github.GitReleaseAsset.GitReleaseAsset*

get_assets()

Calls GET /repos/:owner/:repo/releases/:release_id/assets

Return type *github.PaginatedList.PaginatedList*

3.4.27 GitReleaseAsset

class github.GitReleaseAsset.**GitReleaseAsset**

This class represents GitReleaseAssets. The reference can be found here https://developer.github.com/v3/repos/releases/#get-a-single-release-asset

url

Type string

id

Type integer

name

Type string

label

Type string

content_type

Type string

state

Type string

size

Type integer

download_count

Type integer

created_at

Type datetime

updated_at

Type datetime

browser_download_url

Type string

uploader

Type github.NamedUser.NamedUser

delete_asset()
Delete asset from the release. :rtype: bool

update_asset(*name*, *label=''*)
Update asset metadata. :rtype: github.GitReleaseAsset.GitReleaseAsset

3.4.28 GitTag

class github.GitTag.**GitTag**
This class represents GitTags. The reference can be found here https://developer.github.com/v3/git/tags/

message

Type string

object

Type *github.GitObject.GitObject*

sha

Type string

tag

Type string

tagger

Type *github.GitAuthor.GitAuthor*

url

Type string

3.4.29 GitTree

class github.GitTree.**GitTree**
This class represents GitTrees. The reference can be found here https://developer.github.com/v3/git/trees/

sha

Type string

tree

Type list of *github.GitTreeElement.GitTreeElement*

url

Type string

3.4.30 GitTreeElement

class github.GitTreeElement.**GitTreeElement**

This class represents GitTreeElements

mode

Type string

path

Type string

sha

Type string

size

Type integer

type

Type string

url

Type string

3.4.31 GitignoreTemplate

class github.GitignoreTemplate.**GitignoreTemplate**

This class represents GitignoreTemplates. The reference can be found here https://developer.github.com/v3/gitignore/

source

Type string

name

Type string

3.4.32 Hook

class github.Hook.**Hook**

This class represents Hooks. The reference can be found here http://developer.github.com/v3/repos/hooks

active

Type bool

config

Type dict

created_at

Type datetime.datetime

events

Type list of string

id

Type integer

last_response

Type *github.HookResponse.HookResponse*

name

Type string

test_url

Type string

updated_at

Type datetime.datetime

url

Type string

ping_url

Type string

delete()

Calls DELETE /repos/:owner/:repo/hooks/:id

Return type None

edit(*name*, *config*, *events=NotSet*, *add_events=NotSet*, *remove_events=NotSet*, *active=NotSet*)

Calls PATCH /repos/:owner/:repo/hooks/:id

Parameters

- **name** – string
- **config** – dict
- **events** – list of string
- **add_events** – list of string
- **remove_events** – list of string
- **active** – bool

Return type None

test()

Calls POST /repos/:owner/:repo/hooks/:id/tests

Return type None

ping()

Calls POST /repos/:owner/:repo/hooks/:id/pings

Return type None

3.4.33 HookDescription

class github.HookDescription.**HookDescription**
This class represents HookDescriptions

events

Type list of string

name

Type string

schema

Type list of list of string

supported_events

Type list of string

3.4.34 HookResponse

class github.HookResponse.**HookResponse**
This class represents HookResponses

code

Type integer

message

Type string

status

Type string

3.4.35 Installation

class github.Installation.**Installation**
This class represents Installations. The reference can be found here https://developer.github.com/v3/apps/installations/

get_repos()

Calls GET /installation/repositories

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

3.4.36 InstallationAuthorization

class github.InstallationAuthorization.**InstallationAuthorization**
This class represents InstallationAuthorizations

token

Type string

expires_at

Type datetime

on_behalf_of

Type *github.NamedUser.NamedUser*

3.4.37 Invitation

class github.Invitation.**Invitation**

This class represents repository invitations. The reference can be found here https://developer.github.com/v3/repos/invitations/

id

Type integer

permissions

Type string

created_at

Type string

url

Type string

html_url

Type string

repository

Type Repository

3.4.38 Issue

class github.Issue.**Issue**

This class represents Issues. The reference can be found here https://developer.github.com/v3/issues/

assignee

Type *github.NamedUser.NamedUser*

assignees

Type list of *github.NamedUser.NamedUser*

body

Type string

closed_at

Type datetime.datetime

closed_by

Type *github.NamedUser.NamedUser*

comments

Type integer

comments_url

Type string

created_at

Type datetime.datetime

events_url

Type string

html_url

Type string

id

Type integer

labels

Type list of *github.Label.Label*

labels_url

Type string

milestone

Type *github.Milestone.Milestone*

number

Type integer

pull_request

Type *github.IssuePullRequest.IssuePullRequest*

repository

Type *github.Repository.Repository*

state

Type string

title

Type string

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

locked

Type bool

active_lock_reason

Type string

as_pull_request()

Calls GET /repos/:owner/:repo/pulls/:number

Return type *github.PullRequest.PullRequest*

add_to_assignees(**assignees*)

Calls POST /repos/:owner/:repo/issues/:number/assignees

Parameters **assignee** – *github.NamedUser.NamedUser* or string

Return type None

add_to_labels(**labels*)

Calls POST /repos/:owner/:repo/issues/:number/labels

Parameters **label** – *github.Label.Label* or string

Return type None

create_comment(*body*)

Calls POST /repos/:owner/:repo/issues/:number/comments

Parameters **body** – string

Return type *github.IssueComment.IssueComment*

delete_labels()

Calls DELETE /repos/:owner/:repo/issues/:number/labels

Return type None

edit(*title=NotSet*, *body=NotSet*, *assignee=NotSet*, *state=NotSet*, *milestone=NotSet*, *labels=NotSet*, *assignees=NotSet*)

Calls PATCH /repos/:owner/:repo/issues/:number

Parameters

- **title** – string
- **body** – string
- **assignee** – string or *github.NamedUser.NamedUser* or None
- **assignees** – list (of string or *github.NamedUser.NamedUser*)
- **state** – string
- **milestone** – *github.Milestone.Milestone* or None
- **labels** – list of string

Return type None

lock(*lock_reason*)

Calls PUT /repos/:owner/:repo/issues/:issue_number/lock

Parameters **lock_reason** – string

Return type None

unlock()

Calls DELETE /repos/:owner/:repo/issues/:issue_number/lock

Return type None

get_comment(*id*)

Calls GET /repos/:owner/:repo/issues/comments/:id

Parameters **id** – integer

Return type *github.IssueComment.IssueComment*

get_comments(*since=NotSet*)

Calls GET /repos/:owner/:repo/issues/:number/comments

Parameters **since** – datetime.datetime format YYYY-MM-DDTHH:MM:SSZ

Return type *github.PaginatedList.PaginatedList* of *github.IssueComment.IssueComment*

get_events()

Calls GET /repos/:owner/:repo/issues/:issue_number/events

Return type *github.PaginatedList.PaginatedList* of *github.IssueEvent.IssueEvent*

get_labels()

Calls GET /repos/:owner/:repo/issues/:number/labels

Return type *github.PaginatedList.PaginatedList* of *github.Label.Label*

remove_from_assignees(**assignees*)

Calls DELETE /repos/:owner/:repo/issues/:number/assignees

Parameters **assignee** – *github.NamedUser.NamedUser* or string

Return type None

remove_from_labels(*label*)

Calls DELETE /repos/:owner/:repo/issues/:number/labels/:name

Parameters **label** – *github.Label.Label* or string

Return type None

set_labels(**labels*)

Calls PUT /repos/:owner/:repo/issues/:number/labels

Parameters **labels** – list of *github.Label.Label* or strings

Return type None

get_reactions()

Calls GET /repos/:owner/:repo/issues/:number/reactions

Returns

class *github.PaginatedList.PaginatedList* of *github.Reaction.Reaction*

create_reaction(*reaction_type*)

Calls POST /repos/:owner/:repo/issues/:number/reactions

Parameters **reaction_type** – string

Return type *github.Reaction.Reaction*

3.4.39 IssueComment

class github.IssueComment.**IssueComment**

This class represents IssueComments. The reference can be found here https://developer.github.com/v3/issues/comments/

body

Type string

created_at

Type datetime.datetime

id

Type integer

issue_url

Type string

updated_at

Type datetime.datetime

url

Type string

html_url

Type string

user

Type *github.NamedUser.NamedUser*

delete()

Calls DELETE /repos/:owner/:repo/issues/comments/:id

Return type None

edit(*body*)

Calls PATCH /repos/:owner/:repo/issues/comments/:id

Parameters **body** – string

Return type None

get_reactions()

Calls GET /repos/:owner/:repo/issues/comments/:id/reactions

Returns

class *github.PaginatedList.PaginatedList* of *github.Reaction.Reaction*

create_reaction(*reaction_type*)

Calls POST /repos/:owner/:repo/issues/comments/:id/reactions

Parameters **reaction_type** – string

Return type *github.Reaction.Reaction*

3.4.40 IssueEvent

class github.IssueEvent.**IssueEvent**

This class represents IssueEvents. The reference can be found here https://developer.github.com/v3/issues/events/

actor

Type *github.NamedUser.NamedUser*

commit_id

Type string

created_at

Type datetime.datetime

event

Type string

id

Type integer

issue

Type *github.Issue.Issue*

url

Type string

node_id

Type string

commit_url

Type string

label

Type *github.Label.Label*

assignee

Type *github.NamedUser.NamedUser*

assigner

Type *github.NamedUser.NamedUser*

review_requester

Type *github.NamedUser.NamedUser*

requested_reviewer

Type *github.NamedUser.NamedUser*

milestone

Type *github.Milestone.Milestone*

rename

Type dict

dismissed_review

Type dict

lock_reason

Type string

3.4.41 IssuePullRequest

class github.IssuePullRequest.**IssuePullRequest**

This class represents IssuePullRequests

diff_url

Type string

html_url

Type string

patch_url

Type string

3.4.42 Label

class github.Label.**Label**

This class represents Labels. The reference can be found here http://developer.github.com/v3/issues/labels/

color

Type string

description

Type string

name

Type string

url

Type string

delete()

Calls DELETE /repos/:owner/:repo/labels/:name

Return type None

edit(*name*, *color*, *description=NotSet*)

Calls PATCH /repos/:owner/:repo/labels/:name

Parameters

- **name** – string
- **color** – string
- **description** – string

Return type None

3.4.43 License

class github.License.**License**

This class represents Licenses. The reference can be found here https://developer.github.com/v3/licenses/

key

Type string

name

Type string

spdx_id

Type string

url

Type string

html_url

Type string

description

Type string

implementation

Type string

body

Type string

permissions

Type list of string

conditions

Type list of string

limitations

Type list of string

3.4.44 Migration

class github.Migration.**Migration**

This class represents Migrations. The reference can be found here http://developer.github.com/v3/migrations/

id

Type int

owner

Type *github.NamedUser.NamedUser*

guid

Type str

state

Type str

lock_repositories

Type bool

exclude_attachments

Type bool

repositories

Type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

url

Type str

created_at

Type datetime.datetime

Return type None

updated_at

Type datetime.datetime

Return type None

get_status()

Calls **'GET /user/migrations/:migration_id'_**

Return type str

get_archive_url()

Calls **'GET /user/migrations/:migration_id/archive'_**

Return type str

delete()

Calls **'DELETE /user/migrations/:migration_id/archive'_**

unlock_repo(*repo_name*)

Calls **'DELETE /user/migrations/:migration_id/repos/:repo_name/lock'_**

Parameters **repo_name** – str

Return type None

3.4.45 Milestone

class github.Milestone.**Milestone**

This class represents Milestones. The reference can be found here http://developer.github.com/v3/issues/milestones/

closed_issues

Type integer

created_at

Type datetime.datetime

creator

Type *github.NamedUser.NamedUser*

description

Type string

due_on

Type datetime.datetime

id

Type integer

labels_url

Type string

number

Type integer

open_issues

Type integer

state

Type string

title

Type string

updated_at

Type datetime.datetime

url

Type string

delete()

Calls DELETE /repos/:owner/:repo/milestones/:number

Return type None

edit(*title*, *state=NotSet*, *description=NotSet*, *due_on=NotSet*)

Calls PATCH /repos/:owner/:repo/milestones/:number

Parameters

- **title** – string
- **state** – string
- **description** – string
- **due_on** – date

Return type None

get_labels()

Calls GET /repos/:owner/:repo/milestones/:number/labels

Return type *github.PaginatedList.PaginatedList* of *github.Label.Label*

3.4.46 NamedUser

class github.NamedUser.**NamedUser**

This class represents NamedUsers. The reference can be found here https://developer.github.com/v3/users/#get-a-single-user

node_id

Type string

avatar_url

Type string

bio

Type string

blog

Type string

collaborators

Type integer

company

Type string

contributions

Type integer

created_at

Type datetime.datetime

disk_usage

Type integer

email

Type string

events_url

Type string

followers

Type integer

followers_url

Type string

following

Type integer

following_url

Type string

gists_url

Type string

gravatar_id

Type string

hireable

Type bool

html_url

Type string

id

Type integer

location

Type string

login

Type string

name

Type string

organizations_url

Type string

owned_private_repos

Type integer

permissions

Type *github.Permissions.Permissions*

plan

Type *github.Plan.Plan*

private_gists

Type integer

public_gists

Type integer

public_repos

Type integer

received_events_url

Type string

repos_url

Type string

site_admin

Type bool

starred_url

Type string

subscriptions_url

Type string

suspended_at

Type datetime.datetime

total_private_repos

Type integer

type

Type string

updated_at

Type datetime.datetime

url

Type string

get_events()

Calls GET /users/:user/events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_followers()

Calls GET /users/:user/followers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_following()

Calls GET /users/:user/following

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_gists(*since=NotSet*)

Calls GET /users/:user/gists

Parameters **since** – datetime.datetime format YYYY-MM-DDTHH:MM:SSZ

Return type *github.PaginatedList.PaginatedList* of *github.Gist.Gist*

get_keys()

Calls GET /users/:user/keys

Return type *github.PaginatedList.PaginatedList* of *github.UserKey.UserKey*

get_orgs()

Calls GET /users/:user/orgs

Return type *github.PaginatedList.PaginatedList* of *github.Organization.Organization*

get_public_events()

Calls GET /users/:user/events/public

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_public_received_events()

Calls GET /users/:user/received_events/public

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_received_events()

Calls GET /users/:user/received_events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_repo(*name*)

Calls GET /repos/:owner/:repo

Parameters **name** – string

Return type *github.Repository.Repository*

get_repos(*type=NotSet*, *sort=NotSet*, *direction=NotSet*)

Calls GET /users/:user/repos

Parameters

- **type** – string
- **sort** – string
- **direction** – string

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_starred()

Calls GET /users/:user/starred

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_subscriptions()

Calls GET /users/:user/subscriptions

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_watched()

Calls GET /users/:user/watched

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

has_in_following(*following*)

Calls GET /users/:user/following/:target_user

Parameters **following** – *github.NamedUser.NamedUser*

Return type bool

3.4.47 Notification

class github.Notification.**Notification**

This class represents Notifications. The reference can be found here http://developer.github.com/v3/activity/notifications/

id

Type string

last_read_at

Type datetime.datetime

repository

Type *github.Repository.Repository*

subject

Type *github.NotificationSubject.NotificationSubject*

reason

Type string

subscription_url

Type string

unread

Type bool

updated_at

Type datetime.datetime

url

Type string

mark_as_read()

Calls PATCH /notifications/threads/:id

3.4.48 NotificationSubject

class github.NotificationSubject.**NotificationSubject**

This class represents Subjects of Notifications. The reference can be found here http://developer.github.com/v3/activity/notifications/#list-your-notifications

title

Type string

url

Type string

latest_comment_url

Type string

type

Type string

3.4.49 Organization

class github.Organization.**Organization**

This class represents Organizations. The reference can be found here http://developer.github.com/v3/orgs/

avatar_url

Type string

billing_email

Type string

blog

Type string

collaborators

Type integer

company

Type string

created_at

Type datetime.datetime

description

Type string

disk_usage

Type integer

email

Type string

events_url

Type string

followers

Type integer

following

Type integer

gravatar_id

Type string

html_url

Type string

id

Type integer

location

Type string

login

Type string

members_url

Type string

name

Type string

owned_private_repos

Type integer

plan

Type *github.Plan.Plan*

private_gists

Type integer

public_gists

Type integer

public_members_url

Type string

public_repos

Type integer

repos_url

Type string

total_private_repos

Type integer

type

Type string

updated_at

Type datetime.datetime

url

Type string

add_to_members(*member*, *role=NotSet*)

Calls PUT /orgs/:org/memberships/:user

Parameters

- **member** – *github.NamedUser.NamedUser*
- **role** – string

Return type None

add_to_public_members(*public_member*)

Calls PUT /orgs/:org/public_members/:user

Parameters **public_member** – *github.NamedUser.NamedUser*

Return type None

create_fork(*repo*)

Calls POST /repos/:owner/:repo/forks

Parameters **repo** – *github.Repository.Repository*

Return type *github.Repository.Repository*

create_hook(*name*, *config*, *events=NotSet*, *active=NotSet*)

Calls POST /orgs/:owner/hooks

Parameters

- **name** – string
- **config** – dict
- **events** – list of string
- **active** – bool

Return type *github.Hook.Hook*

create_repo(*name*, *description=NotSet*, *homepage=NotSet*, *private=NotSet*, *has_issues=NotSet*, *has_wiki=NotSet*, *has_downloads=NotSet*, *has_projects=NotSet*, *team_id=NotSet*, *auto_init=NotSet*, *license_template=NotSet*, *gitignore_template=NotSet*, *allow_squash_merge=NotSet*, *allow_merge_commit=NotSet*, *allow_rebase_merge=NotSet*)

Calls POST /orgs/:org/repos

Parameters

- **name** – string
- **description** – string
- **homepage** – string
- **private** – bool
- **has_issues** – bool
- **has_wiki** – bool
- **has_downloads** – bool
- **has_projects** – bool
- **team_id** – : int
- **auto_init** – bool
- **license_template** – string
- **gitignore_template** – string
- **allow_squash_merge** – bool
- **allow_merge_commit** – bool
- **allow_rebase_merge** – bool

Return type *github.Repository.Repository*

create_team(*name*, *repo_names=NotSet*, *permission=NotSet*, *privacy=NotSet*, *description=NotSet*)

Calls POST /orgs/:org/teams

Parameters

- **name** – string
- **repo_names** – list of *github.Repository.Repository*
- **permission** – string
- **privacy** – string
- **description** – string

Return type *github.Team.Team*

delete_hook(*id*)

Calls DELETE /orgs/:owner/hooks/:id

Parameters **id** – integer

Return type None‘

edit(*billing_email=NotSet*, *blog=NotSet*, *company=NotSet*, *description=NotSet*, *email=NotSet*, *location=NotSet*, *name=NotSet*)

Calls PATCH /orgs/:org

Parameters

- **billing_email** – string
- **blog** – string
- **company** – string
- **description** – string
- **email** – string
- **location** – string
- **name** – string

Return type None

edit_hook(*id*, *name*, *config*, *events=NotSet*, *active=NotSet*)

Calls PATCH /orgs/:owner/hooks/:id

Parameters

- **id** – integer
- **name** – string
- **config** – dict
- **events** – list of string
- **active** – bool

Return type *github.Hook.Hook*

get_events()

Calls GET /orgs/:org/events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_hook(*id*)

Calls GET /orgs/:owner/hooks/:id

Parameters **id** – integer

Return type *github.Hook.Hook*

get_hooks()

Calls GET /orgs/:owner/hooks

Return type *github.PaginatedList.PaginatedList* of *github.Hook.Hook*

get_issues(*filter=NotSet*, *state=NotSet*, *labels=NotSet*, *sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /orgs/:org/issues

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

Parameters

- **filter** – string
- **state** – string
- **labels** – list of *github.Label.Label*
- **sort** – string
- **direction** – string
- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

get_members(*filter_=NotSet*, *role=NotSet*)

Calls GET /orgs/:org/members

Parameters

- **filter** – string
- **role** – string

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_projects(*state=NotSet*)

Calls GET /orgs/:org/projects

Return type *github.PaginatedList.PaginatedList* of *github.Project.Project*

Parameters **state** – string

get_public_members()

Calls GET /orgs/:org/public_members

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_outside_collaborators(*filter_=NotSet*)

Calls GET /orgs/:org/outside_collaborators

Parameters **filter** – string

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

remove_outside_collaborator(*collaborator*)

Calls DELETE /orgs/:org/outside_collaborators/:username

Parameters **collaborator** – *github.NamedUser.NamedUser*

Return type None

convert_to_outside_collaborator(*member*)

Calls PUT /orgs/:org/outside_collaborators/:username

Parameters **member** – *github.NamedUser.NamedUser*

Return type None

get_repo(*name*)

Calls GET /repos/:owner/:repo

Parameters **name** – string

Return type *github.Repository.Repository*

get_repos(*type=NotSet*, *sort=NotSet*, *direction=NotSet*)

Calls GET /orgs/:org/repos

Parameters

- **type** – string ('all', 'public', 'private', 'forks', 'sources', 'member')
- **sort** – string ('created', 'updated', 'pushed', 'full_name')
- **direction** – string ('asc', desc')

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_team(*id*)

Calls GET /teams/:id

Parameters **id** – integer

Return type *github.Team.Team*

get_team_by_slug(*slug*)

Calls GET /orgs/:org/teams/:team_slug

Parameters **slug** – string

Return type *github.Team.Team*

get_teams()

Calls GET /orgs/:org/teams

Return type *github.PaginatedList.PaginatedList* of *github.Team.Team*

invite_user(*user=NotSet*, *email=NotSet*, *role=NotSet*, *teams=NotSet*)

Calls POST /orgs/:org/invitations

Parameters

- **user** – *github.NamedUser.NamedUser*
- **email** – string

- **role** – string
- **teams** – array of *github.Team.Team*

Return type None

has_in_members(*member*)

Calls GET /orgs/:org/members/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type bool

has_in_public_members(*public_member*)

Calls GET /orgs/:org/public_members/:user

Parameters **public_member** – *github.NamedUser.NamedUser*

Return type bool

remove_from_membership(*member*)

Calls DELETE /orgs/:org/memberships/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type None

remove_from_members(*member*)

Calls DELETE /orgs/:org/members/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type None

remove_from_public_members(*public_member*)

Calls DELETE /orgs/:org/public_members/:user

Parameters **public_member** – *github.NamedUser.NamedUser*

Return type None

create_migration(*repos*, *lock_repositories=NotSet*, *exclude_attachments=NotSet*)

Calls **'POST /orgs/:org/migrations'_**

Parameters

- **repos** – list or tuple of str
- **lock_repositories** – bool
- **exclude_attachments** – bool

Return type *github.Migration.Migration*

get_migrations()

Calls **'GET /orgs/:org/migrations'_**

Return type *github.PaginatedList.PaginatedList* of *github.Migration.Migration*

3.4.50 Path

class github.Path.**Path**

This class represents a popular Path for a GitHub repository. The reference can be found here https://developer.github.com/v3/repos/traffic/

path

Type string

title

Type string

count

Type integer

uniques

Type integer

3.4.51 Permissions

class github.Permissions.**Permissions**

This class represents Permissions

admin

Type bool

pull

Type bool

push

Type bool

3.4.52 Plan

class github.Plan.**Plan**

This class represents Plans

collaborators

Type integer

name

Type string

private_repos

Type integer

space

Type integer

3.4.53 Project

class github.Project.**Project**

This class represents Projects. The reference can be found here http://developer.github.com/v3/projects

body

Type string

columns_url

Type string

created_at

Type datetime.datetime

creator

Type *github.NamedUser.NamedUser*

html_url

Type string

id

Type integer

name

Type string

node_id

Type string

number

Type integer

owner_url

Type string

state

Type string

updated_at

Type datetime.datetime

url

Type string

get_columns()

Calls GET /projects/:project_id/columns

Return type *github.PaginatedList.PaginatedList* of *github.ProjectColumn.ProjectColumn*

create_column(*name*)

calls: **'POST https://developer.github.com/v3/projects/columns/#create-a-project-column>'_** :param name: string

3.4.54 ProjectCard

class github.ProjectCard.**ProjectCard**

This class represents Project Cards. The reference can be found here https://developer.github.com/v3/projects/cards

archived

Type bool

column_url

Type string

content_url

Type string

created_at

Type datetime.datetime

creator

Type *github.NamedUser.NamedUser*

id

Type integer

node_id

Type string

note

Type string

updated_at

Type datetime.datetime

url

Type string

get_content(*content_type=NotSet*)

Calls GET /repos/:owner/:repo/pulls/:number

Return type *github.PullRequest.PullRequest* or *github.Issue.Issue*

3.4.55 ProjectColumn

class github.ProjectColumn.**ProjectColumn**

This class represents Project Columns. The reference can be found here http://developer.github.com/v3/projects/columns

cards_url

Type string

created_at

Type datetime.datetime

id

Type integer

name

Type string

node_id

Type string

project_url

Type string

updated_at

Type datetime.datetime

url

Type string

get_cards(*archived_state=NotSet*)

Calls GET /projects/columns/:column_id/cards

Return type *github.PaginatedList.PaginatedList* of *github.ProjectCard.ProjectCard*

Parameters **archived_state** – string

create_card(*note=NotSet*, *content_id=NotSet*, *content_type=NotSet*)

Calls POST /projects/columns/:column_id/cards

Parameters

- **note** – string
- **content_id** – integer
- **content_type** – string

3.4.56 PullRequest

class github.PullRequest.**PullRequest**

This class represents PullRequests. The reference can be found here http://developer.github.com/v3/pulls/

additions

Type integer

assignee

Type *github.NamedUser.NamedUser*

assignees

Type list of *github.NamedUser.NamedUser*

base

Type *github.PullRequestPart.PullRequestPart*

body

Type string

changed_files

Type integer

closed_at

Type datetime.datetime

comments

Type integer

comments_url

Type string

commits

Type integer

commits_url

Type string

created_at

Type datetime.datetime

deletions

Type integer

diff_url

Type string

head

Type *github.PullRequestPart.PullRequestPart*

html_url

Type string

id

Type integer

issue_url

Type string

labels

Type list of *github.Label.Label*

merge_commit_sha

Type string

mergeable

Type bool

mergeable_state

Type string

merged

Type bool

merged_at

Type datetime.datetime

merged_by

Type *github.NamedUser.NamedUser*

milestone

Type *github.Milestone.Milestone*

number

Type integer

patch_url

Type string

review_comment_url

Type string

review_comments

Type integer

review_comments_url

Type string

state

Type string

title

Type string

updated_at

Type datetime.datetime

url

Type string

user

Type *github.NamedUser.NamedUser*

as_issue()

Calls GET /repos/:owner/:repo/issues/:number

Return type *github.Issue.Issue*

create_comment(*body*, *commit_id*, *path*, *position*)

Calls POST /repos/:owner/:repo/pulls/:number/comments

Parameters

- **body** – string
- **commit_id** – *github.Commit.Commit*
- **path** – string
- **position** – integer

Return type *github.PullRequestComment.PullRequestComment*

create_review_comment(*body*, *commit_id*, *path*, *position*)

Calls POST /repos/:owner/:repo/pulls/:number/comments

Parameters

- **body** – string
- **commit_id** – *github.Commit.Commit*
- **path** – string
- **position** – integer

Return type *github.PullRequestComment.PullRequestComment*

create_issue_comment(*body*)

Calls POST /repos/:owner/:repo/issues/:number/comments

Parameters **body** – string

Return type *github.IssueComment.IssueComment*

create_review(*commit=NotSet*, *body=None*, *event=NotSet*, *comments=NotSet*)

Calls POST /repos/:owner/:repo/pulls/:number/reviews

Parameters

- **commit** – github.Commit.Commit
- **body** – string
- **event** – string
- **comments** – list

Return type *github.PullRequestReview.PullRequestReview*

create_review_request(*reviewers=NotSet*, *team_reviewers=NotSet*)

Calls POST /repos/:owner/:repo/pulls/:number/requested_reviewers

Parameters

- **reviewers** – list of strings
- **team_reviewers** – list of strings

Return type None

delete_review_request(*reviewers=NotSet*, *team_reviewers=NotSet*)

Calls DELETE /repos/:owner/:repo/pulls/:number/requested_reviewers

Parameters

- **reviewers** – list of strings
- **team_reviewers** – list of strings

Return type None

edit(*title=NotSet*, *body=NotSet*, *state=NotSet*, *base=NotSet*)

Calls PATCH /repos/:owner/:repo/pulls/:number

Parameters

- **title** – string

- **body** – string
- **state** – string
- **base** – string

Return type None

get_comment(*id*)

Calls GET /repos/:owner/:repo/pulls/comments/:number

Parameters **id** – integer

Return type *github.PullRequestComment.PullRequestComment*

get_review_comment(*id*)

Calls GET /repos/:owner/:repo/pulls/comments/:number

Parameters **id** – integer

Return type *github.PullRequestComment.PullRequestComment*

get_comments()

Warning: this only returns review comments. For normal conversation comments, use get_issue_comments.

Calls GET /repos/:owner/:repo/pulls/:number/comments

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestComment.PullRequestComment*

get_review_comments(*since=NotSet*)

Calls GET /repos/:owner/:repo/pulls/:number/comments

Parameters **since** – datetime.datetime format YYYY-MM-DDTHH:MM:SSZ

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestComment.PullRequestComment*

get_single_review_comments(*id*)

Calls GET /repos/:owner/:repo/pulls/:number/review/:id/comments

Parameters **id** – integer

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestComment.PullRequestComment*

get_commits()

Calls GET /repos/:owner/:repo/pulls/:number/commits

Return type *github.PaginatedList.PaginatedList* of *github.Commit.Commit*

get_files()

Calls GET /repos/:owner/:repo/pulls/:number/files

Return type *github.PaginatedList.PaginatedList* of *github.File.File*

get_issue_comment(*id*)

Calls GET /repos/:owner/:repo/issues/comments/:id

Parameters **id** – integer

Return type *github.IssueComment.IssueComment*

get_issue_comments()

Calls GET /repos/:owner/:repo/issues/:number/comments

Return type *github.PaginatedList.PaginatedList* of *github.IssueComment.IssueComment*

get_review(*id*)

Calls GET /repos/:owner/:repo/pulls/:number/reviews/:id

Parameters **id** – integer

Return type *github.PullRequestReview.PullRequestReview*

get_reviews()

Calls GET /repos/:owner/:repo/pulls/:number/reviews

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestReview.PullRequestReview*

get_review_requests()

Calls GET /repos/:owner/:repo/pulls/:number/requested_reviewers

Return type tuple of *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser* and of *github.PaginatedList.PaginatedList* of *github.Team.Team*

get_labels()

Calls GET /repos/:owner/:repo/issues/:number/labels

Return type *github.PaginatedList.PaginatedList* of *github.Label.Label*

add_to_labels(**labels*)

Calls POST /repos/:owner/:repo/issues/:number/labels

Parameters **label** – *github.Label.Label* or string

Return type None

delete_labels()

Calls DELETE /repos/:owner/:repo/issues/:number/labels

Return type None

remove_from_labels(*label*)

Calls DELETE /repos/:owner/:repo/issues/:number/labels/:name

Parameters **label** – *github.Label.Label* or string

Return type None

set_labels(**labels*)

Calls PUT /repos/:owner/:repo/issues/:number/labels

Parameters **labels** – list of *github.Label.Label* or strings

Return type None

is_merged()

Calls GET /repos/:owner/:repo/pulls/:number/merge

Return type bool

merge(*commit_message=NotSet*, *commit_title=NotSet*, *merge_method=NotSet*, *sha=NotSet*)

Calls PUT /repos/:owner/:repo/pulls/:number/merge

Parameters **commit_message** – string

Return type *github.PullRequestMergeStatus.PullRequestMergeStatus*

3.4.57 PullRequestComment

class github.PullRequestComment.**PullRequestComment**

This class represents PullRequestComments. The reference can be found here http://developer.github.com/v3/pulls/comments/

body

Type string

commit_id

Type string

created_at

Type datetime.datetime

diff_hunk

Type string

id

Type integer

in_reply_to_id

Type integer

original_commit_id

Type string

original_position

Type integer

path

Type string

position

Type integer

pull_request_url

Type string

updated_at

Type datetime.datetime

url

Type string

html_url

Type string

user

Type *github.NamedUser.NamedUser*

delete()

Calls DELETE /repos/:owner/:repo/pulls/comments/:number

Return type None

edit(*body*)

Calls PATCH /repos/:owner/:repo/pulls/comments/:number

Parameters **body** – string

Return type None

get_reactions()

Calls GET /repos/:owner/:repo/pulls/comments/:number/reactions

Returns

class *github.PaginatedList.PaginatedList* of *github.Reaction.Reaction*

create_reaction(*reaction_type*)

Calls POST /repos/:owner/:repo/pulls/comments/:number/reactions

Parameters **reaction_type** – string

Return type *github.Reaction.Reaction*

3.4.58 PullRequestMergeStatus

class github.PullRequestMergeStatus.**PullRequestMergeStatus**

This class represents PullRequestMergeStatuses. The reference can be found here http://developer.github.com/v3/pulls/#get-if-a-pull-request-has-been-merged

merged

Type bool

message

Type string

sha

Type string

3.4.59 PullRequestPart

class github.PullRequestPart.**PullRequestPart**

This class represents PullRequestParts

label

Type string

ref

Type string

repo

Type *github.Repository.Repository*

sha

Type string

user

Type *github.NamedUser.NamedUser*

3.4.60 PullRequestReview

class github.PullRequestReview.**PullRequestReview**

This class represents PullRequestReviews. The reference can be found here https://developer.github.com/v3/pulls/reviews/

id

Type integer

user

Type *github.NamedUser.NamedUser*

body

Type string

commit_id

Type string

state

Type string

url

Type string

html_url

Type string

pull_request_url

Type string

submitted_at

Type datetime.datetime

dismiss(*message*)

Calls PUT /repos/:owner/:repo/pulls/:number/reviews/:review_id/dismissals

Return type None

3.4.61 Rate

class github.Rate.**Rate**

This class represents Rates. The reference can be found here http://developer.github.com/v3/rate_limit

limit

Type integer

remaining

Type integer

reset

Type datetime.datetime

3.4.62 RateLimit

class github.RateLimit.**RateLimit**

This class represents RateLimits. The reference can be found here http://developer.github.com/v3/rate_limit

rate

(Deprecated) Rate limit for non-search-related API, use *core* instead

Type class:*github.Rate.Rate*

core

Rate limit for the non-search-related API

Type class:*github.Rate.Rate*

search

Rate limit for the Search API.

Type class:*github.Rate.Rate*

graphql

(Experimental) Rate limit for GraphQL API, use with caution.

Type class:*github.Rate.Rate*

3.4.63 Reaction

class github.Reaction.**Reaction**

This class represents Reactions. The reference can be found here https://developer.github.com/v3/reactions/

content

Type string

created_at

Type datetime.datetime

id

Type integer

user

Type *github.NamedUser.NamedUser*

delete()

Calls DELETE /reactions/:id

Return type None

3.4.64 Referrer

class github.Referrer.**Referrer**

This class represents a popylar Referrer for a GitHub repository. The reference can be found here https://developer.github.com/v3/repos/traffic/

referrer

Type string

count

Type integer

uniques

Type integer

3.4.65 Repository

class github.Repository.**Repository**

This class represents Repositories. The reference can be found here http://developer.github.com/v3/repos/

allow_merge_commit

Type bool

allow_rebase_merge

Type bool

allow_squash_merge

Type bool

archived

Type bool

archive_url

Type string

assignees_url

Type string

blobs_url

Type string

branches_url

Type string

clone_url

Type string

collaborators_url

Type string

comments_url

Type string

commits_url

Type string

compare_url

Type string

contents_url

Type string

contributors_url

Type string

created_at

Type datetime.datetime

default_branch

Type string

description

Type string

downloads_url

Type string

events_url

Type string

fork

Type bool

forks

Type integer

forks_count

Type integer

forks_url

Type string

full_name

Type string

git_commits_url

Type string

git_refs_url

Type string

git_tags_url

Type string

git_url

Type string

has_downloads

Type bool

has_issues

Type bool

has_projects

Type bool

has_wiki

Type bool

homepage

Type string

hooks_url

Type string

html_url

Type string

id

Type integer

issue_comment_url

Type string

issue_events_url

Type string

issues_url

Type string

keys_url

Type string

labels_url

Type string

language

Type string

languages_url

Type string

master_branch

Type string

merges_url

Type string

milestones_url

Type string

mirror_url

Type string

name

Type string

network_count

Type integer

notifications_url

Type string

open_issues

Type integer

open_issues_count

Type integer

organization

Type *github.Organization.Organization*

owner

Type *github.NamedUser.NamedUser*

parent

Type *github.Repository.Repository*

permissions

Type *github.Permissions.Permissions*

private

Type bool

pulls_url

Type string

pushed_at

Type datetime.datetime

size

Type integer

source

Type *github.Repository.Repository*

ssh_url

Type string

stargazers_count

Type integer

stargazers_url

Type string

statuses_url

Type string

subscribers_url

Type string

subscribers_count

Type integer

subscription_url

Type string

svn_url

Type string

tags_url

Type string

teams_url

Type string

topics

Type list of strings

trees_url

Type string

updated_at

Type datetime.datetime

url

Type string

watchers

Type integer

watchers_count

Type integer

add_to_collaborators(*collaborator*, *permission=NotSet*)

Calls PUT /repos/:owner/:repo/collaborators/:user

Parameters

- **collaborator** – string or *github.NamedUser.NamedUser*
- **permission** – string 'pull', 'push' or 'admin'

Return type None

get_collaborator_permission(*collaborator*)

Calls GET /repos/:owner/:repo/collaborators/:username/permission

Parameters **collaborator** – string or *github.NamedUser.NamedUser*

Return type string

compare(*base*, *head*)

Calls GET /repos/:owner/:repo/compare/:base...:head

Parameters

- **base** – string
- **head** – string

Return type *github.Comparison.Comparison*

create_git_blob(*content*, *encoding*)

Calls POST /repos/:owner/:repo/git/blobs

Parameters

- **content** – string
- **encoding** – string

Return type *github.GitBlob.GitBlob*

create_git_commit(*message*, *tree*, *parents*, *author=NotSet*, *committer=NotSet*)

Calls POST /repos/:owner/:repo/git/commits

Parameters

- **message** – string
- **tree** – *github.GitTree.GitTree*
- **parents** – list of *github.GitCommit.GitCommit*
- **author** – *github.InputGitAuthor.InputGitAuthor*
- **committer** – *github.InputGitAuthor.InputGitAuthor*

Return type *github.GitCommit.GitCommit*

create_git_ref(*ref*, *sha*)

Calls POST /repos/:owner/:repo/git/refs

Parameters

- **ref** – string
- **sha** – string

Return type *github.GitRef.GitRef*

create_git_release(*tag*, *name*, *message*, *draft=False*, *prerelease=False*, *target_commitish=NotSet*)

Calls POST /repos/:owner/:repo/releases

Parameters

- **tag** – string
- **name** – string
- **message** – string
- **draft** – bool

- **prerelease** – bool
- **target_commitish** – string or *github.Branch.Branch* or *github.Commit.Commit* or *github.GitCommit.GitCommit*

Return type *github.GitRelease.GitRelease*

create_git_tag(*tag*, *message*, *object*, *type*, *tagger=NotSet*)

Calls POST /repos/:owner/:repo/git/tags

Parameters

- **tag** – string
- **message** – string
- **object** – string
- **type** – string
- **tagger** – *github.InputGitAuthor.InputGitAuthor*

Return type *github.GitTag.GitTag*

create_git_tree(*tree*, *base_tree=NotSet*)

Calls POST /repos/:owner/:repo/git/trees

Parameters

- **tree** – list of *github.InputGitTreeElement.InputGitTreeElement*
- **base_tree** – *github.GitTree.GitTree*

Return type *github.GitTree.GitTree*

create_hook(*name*, *config*, *events=NotSet*, *active=NotSet*)

Calls POST /repos/:owner/:repo/hooks

Parameters

- **name** – string
- **config** – dict
- **events** – list of string
- **active** – bool

Return type *github.Hook.Hook*

create_issue(*title*, *body=NotSet*, *assignee=NotSet*, *milestone=NotSet*, *labels=NotSet*, *assignees=NotSet*)

Calls POST /repos/:owner/:repo/issues

Parameters

- **title** – string
- **body** – string
- **assignee** – string or *github.NamedUser.NamedUser*
- **assignees** – list (of string or *github.NamedUser.NamedUser*)
- **milestone** – *github.Milestone.Milestone*
- **labels** – list of *github.Label.Label*

Return type *github.Issue.Issue*

create_key(*title*, *key*, *read_only=False*)

Calls POST /repos/:owner/:repo/keys

Parameters

- **title** – string
- **key** – string
- **read_only** – bool

Return type *github.RepositoryKey.RepositoryKey*

create_label(*name*, *color*, *description=NotSet*)

Calls POST /repos/:owner/:repo/labels

Parameters

- **name** – string
- **color** – string
- **description** – string

Return type *github.Label.Label*

create_milestone(*title*, *state=NotSet*, *description=NotSet*, *due_on=NotSet*)

Calls POST /repos/:owner/:repo/milestones

Parameters

- **title** – string
- **state** – string
- **description** – string
- **due_on** – datetime

Return type *github.Milestone.Milestone*

create_project(*name*, *body=NotSet*)
calls: POST /repos/:owner/:repo/projects :param name: string :param body: string

create_pull(**args*, ***kwds*)

Calls POST /repos/:owner/:repo/pulls

Parameters

- **title** – string
- **body** – string
- **issue** – *github.Issue.Issue*
- **base** – string
- **head** – string
- **maintainer_can_modify** – bool

Return type *github.PullRequest.PullRequest*

create_source_import(*vcs*, *vcs_url*, *vcs_username=NotSet*, *vcs_password=NotSet*)

Calls PUT /repos/:owner/:repo/import

Parameters

- **vcs** – string
- **vcs_url** – string
- **vcs_username** – string
- **vcs_password** – string

Return type *github.SourceImport.SourceImport*

delete()

Calls DELETE /repos/:owner/:repo

Return type None

edit(*name=None*, *description=NotSet*, *homepage=NotSet*, *private=NotSet*, *has_issues=NotSet*, *has_projects=NotSet*, *has_wiki=NotSet*, *has_downloads=NotSet*, *default_branch=NotSet*, *allow_squash_merge=NotSet*, *allow_merge_commit=NotSet*, *allow_rebase_merge=NotSet*, *archived=NotSet*)

Calls PATCH /repos/:owner/:repo

Parameters

- **name** – string
- **description** – string
- **homepage** – string
- **private** – bool
- **has_issues** – bool
- **has_projects** – bool
- **has_wiki** – bool
- **has_downloads** – bool
- **default_branch** – string
- **allow_squash_merge** – bool
- **allow_merge_commit** – bool
- **allow_rebase_merge** – bool
- **archived** – bool. Unarchiving repositories is currently not supported through API (https://developer.github.com/v3/repos/#edit)

Return type None

get_archive_link(*archive_format*, *ref=NotSet*)

Calls GET /repos/:owner/:repo/:archive_format/:ref

Parameters

- **archive_format** – string
- **ref** – string

Return type string

get_assignees()

Calls GET /repos/:owner/:repo/assignees

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_branch(*branch*)

Calls GET /repos/:owner/:repo/branches/:branch

Parameters **branch** – string

Return type *github.Branch.Branch*

get_branches()

Calls GET /repos/:owner/:repo/branches

Return type *github.PaginatedList.PaginatedList* of *github.Branch.Branch*

get_collaborators(*affiliation=NotSet*)

Calls GET /repos/:owner/:repo/collaborators

Parameters **affiliation** – string

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_comment(*id*)

Calls GET /repos/:owner/:repo/comments/:id

Parameters **id** – integer

Return type *github.CommitComment.CommitComment*

get_comments()

Calls GET /repos/:owner/:repo/comments

Return type *github.PaginatedList.PaginatedList* of *github.CommitComment.CommitComment*

get_commit(*sha*)

Calls GET /repos/:owner/:repo/commits/:sha

Parameters **sha** – string

Return type *github.Commit.Commit*

get_commits(*sha=NotSet*, *path=NotSet*, *since=NotSet*, *until=NotSet*, *author=NotSet*)

Calls GET /repos/:owner/:repo/commits

Parameters

- **sha** – string
- **path** – string
- **since** – datetime.datetime
- **until** – datetime.datetime
- **author** – string or *github.NamedUser.NamedUser* or *github.AuthenticatedUser.AuthenticatedUser*

Return type *github.PaginatedList.PaginatedList* of *github.Commit.Commit*

get_contents(*path*, *ref=NotSet*)

Calls GET /repos/:owner/:repo/contents/:path

Parameters

- **path** – string
- **ref** – string

Return type *github.ContentFile.ContentFile*

get_top_referrers()

Calls GET /repos/:owner/:repo/traffic/popular/referrers

Return type list of *github.Referrer.Referrer*

get_top_paths()

Calls GET /repos/:owner/:repo/traffic/popular/paths

Return type list of *github.Path.Path*

get_views_traffic(*per=NotSet*)

Calls GET /repos/:owner/:repo/traffic/views

Parameters **per** – string, must be one of day or week, day by default

Return type None or list of *github.View.View*

get_clones_traffic(*per=NotSet*)

Calls GET /repos/:owner/:repo/traffic/clones

Parameters **per** – string, must be one of day or week, day by default

Return type None or list of `github.Clone.Clone`

get_projects(*state=NotSet*)

Calls GET /repos/:owner/:repo/projects

Return type *github.PaginatedList.PaginatedList* of *github.Project.Project*

Parameters **state** – string

create_file(*path*, *message*, *content*, *branch=NotSet*, *committer=NotSet*, *author=NotSet*)

Create a file in this repository.

Calls PUT /repos/:owner/:repo/contents/:path

Parameters

- **path** – string, (required), path of the file in the repository
- **message** – string, (required), commit message
- **content** – string, (required), the actual data in the file
- **branch** – string, (optional), branch to create the commit on. Defaults to the default branch of the repository
- **committer** – InputGitAuthor, (optional), if no information is given the authenticated user's information will be used. You must specify both a name and email.

- **author** – InputGitAuthor, (optional), if omitted this will be filled in with committer information. If passed, you must specify both a name and email.

Return type

{ 'content': *ContentFile*:, 'commit': *Commit*}

update_file(*path*, *message*, *content*, *sha*, *branch=NotSet*, *committer=NotSet*, *author=NotSet*)

This method updates a file in a repository

Calls PUT /repos/:owner/:repo/contents/:path

Parameters

- **path** – string, Required. The content path.
- **message** – string, Required. The commit message.
- **content** – string, Required. The updated file content, Base64 encoded.
- **sha** – string, Required. The blob SHA of the file being replaced.
- **branch** – string. The branch name. Default: the repository's default branch (usually master)
- **committer** – InputGitAuthor, (optional), if no information is given the authenticated user's information will be used. You must specify both a name and email.
- **author** – InputGitAuthor, (optional), if omitted this will be filled in with committer information. If passed, you must specify both a name and email.

Return type

{ 'content': *ContentFile*:, 'commit': *Commit*}

delete_file(*path*, *message*, *sha*, *branch=NotSet*, *committer=NotSet*, *author=NotSet*)

This method deletes a file in a repository

Calls DELETE /repos/:owner/:repo/contents/:path

Parameters

- **path** – string, Required. The content path.
- **message** – string, Required. The commit message.
- **sha** – string, Required. The blob SHA of the file being replaced.
- **branch** – string. The branch name. Default: the repository's default branch (usually master)
- **committer** – InputGitAuthor, (optional), if no information is given the authenticated user's information will be used. You must specify both a name and email.
- **author** – InputGitAuthor, (optional), if omitted this will be filled in with committer information. If passed, you must specify both a name and email.

Return type

{ 'content': `null`:, 'commit': *Commit*}

get_dir_contents(*path*, *ref=NotSet*)

Calls GET /repos/:owner/:repo/contents/:path

Parameters

- **path** – string

- **ref** – string

Return type list of *github.ContentFile.ContentFile*

get_contributors(*anon=NotSet*)

Calls GET /repos/:owner/:repo/contributors

Parameters **anon** – string

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_download(*id*)

Calls GET /repos/:owner/:repo/downloads/:id

Parameters **id** – integer

Return type *github.Download.Download*

get_downloads()

Calls GET /repos/:owner/:repo/downloads

Return type *github.PaginatedList.PaginatedList* of *github.Download.Download*

get_events()

Calls GET /repos/:owner/:repo/events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_forks()

Calls GET /repos/:owner/:repo/forks

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

get_git_blob(*sha*)

Calls GET /repos/:owner/:repo/git/blobs/:sha

Parameters **sha** – string

Return type *github.GitBlob.GitBlob*

get_git_commit(*sha*)

Calls GET /repos/:owner/:repo/git/commits/:sha

Parameters **sha** – string

Return type *github.GitCommit.GitCommit*

get_git_ref(*ref*)

Calls GET /repos/:owner/:repo/git/refs/:ref

Parameters **ref** – string

Return type *github.GitRef.GitRef*

get_git_refs()

Calls GET /repos/:owner/:repo/git/refs

Return type *github.PaginatedList.PaginatedList* of *github.GitRef.GitRef*

get_git_tag(*sha*)

Calls GET /repos/:owner/:repo/git/tags/:sha

Parameters **sha** – string

Return type *github.GitTag.GitTag*

get_git_tree(*sha*, *recursive=NotSet*)

Calls GET /repos/:owner/:repo/git/trees/:sha

Parameters

- **sha** – string
- **recursive** – bool

Return type *github.GitTree.GitTree*

get_hook(*id*)

Calls GET /repos/:owner/:repo/hooks/:id

Parameters **id** – integer

Return type *github.Hook.Hook*

get_hooks()

Calls GET /repos/:owner/:repo/hooks

Return type *github.PaginatedList.PaginatedList* of *github.Hook.Hook*

get_issue(*number*)

Calls GET /repos/:owner/:repo/issues/:number

Parameters **number** – integer

Return type *github.Issue.Issue*

get_issues(*milestone=NotSet*, *state=NotSet*, *assignee=NotSet*, *mentioned=NotSet*, *labels=NotSet*, *sort=NotSet*, *direction=NotSet*, *since=NotSet*, *creator=NotSet*)

Calls GET /repos/:owner/:repo/issues

Parameters

- **milestone** – *github.Milestone.Milestone* or “none” or “*”
- **state** – string. *open*, *closed*, or *all*. If this is not set the GitHub API default behavior will be used. At the moment this is to return only open issues. This might change anytime on GitHub API side and it could be clever to explicitly specify the state value.
- **assignee** – string or *github.NamedUser.NamedUser* or “none” or “*”
- **mentioned** – *github.NamedUser.NamedUser*
- **labels** – list of *github.Label.Label*
- **sort** – string
- **direction** – string
- **since** – datetime.datetime
- **creator** – string or *github.NamedUser.NamedUser*

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

get_issues_comments(*sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /repos/:owner/:repo/issues/comments

Parameters

- **sort** – string
- **direction** – string
- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.IssueComment.IssueComment*

get_issues_event(*id*)

Calls GET /repos/:owner/:repo/issues/events/:id

Parameters **id** – integer

Return type *github.IssueEvent.IssueEvent*

get_issues_events()

Calls GET /repos/:owner/:repo/issues/events

Return type *github.PaginatedList.PaginatedList* of *github.IssueEvent.IssueEvent*

get_key(*id*)

Calls GET /repos/:owner/:repo/keys/:id

Parameters **id** – integer

Return type *github.RepositoryKey.RepositoryKey*

get_keys()

Calls GET /repos/:owner/:repo/keys

Return type *github.PaginatedList.PaginatedList* of *github.RepositoryKey.RepositoryKey*

get_label(*name*)

Calls GET /repos/:owner/:repo/labels/:name

Parameters **name** – string

Return type *github.Label.Label*

get_labels()

Calls GET /repos/:owner/:repo/labels

Return type *github.PaginatedList.PaginatedList* of *github.Label.Label*

get_languages()

Calls GET /repos/:owner/:repo/languages

Return type dict of string to integer

get_license()

Calls GET /repos/:owner/:repo/license

Return type *github.ContentFile.ContentFile*

get_milestone(*number*)

Calls GET /repos/:owner/:repo/milestones/:number

Parameters **number** – integer

Return type *github.Milestone.Milestone*

get_milestones(*state=NotSet*, *sort=NotSet*, *direction=NotSet*)

Calls GET /repos/:owner/:repo/milestones

Parameters

- **state** – string
- **sort** – string
- **direction** – string

Return type *github.PaginatedList.PaginatedList* of *github.Milestone.Milestone*

get_network_events()

Calls GET /networks/:owner/:repo/events

Return type *github.PaginatedList.PaginatedList* of *github.Event.Event*

get_pull(*number*)

Calls GET /repos/:owner/:repo/pulls/:number

Parameters **number** – integer

Return type *github.PullRequest.PullRequest*

get_pulls(*state=NotSet*, *sort=NotSet*, *direction=NotSet*, *base=NotSet*, *head=NotSet*)

Calls GET /repos/:owner/:repo/pulls

Parameters

- **state** – string
- **sort** – string
- **direction** – string
- **base** – string
- **head** – string

Return type *github.PaginatedList.PaginatedList* of *github.PullRequest.PullRequest*

get_pulls_comments(*sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /repos/:owner/:repo/pulls/comments

Parameters

- **sort** – string
- **direction** – string
- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestComment.PullRequestComment*

get_pulls_review_comments(*sort=NotSet*, *direction=NotSet*, *since=NotSet*)

Calls GET /repos/:owner/:repo/pulls/comments

Parameters

- **sort** – string
- **direction** – string
- **since** – datetime.datetime

Return type *github.PaginatedList.PaginatedList* of *github.PullRequestComment.PullRequestComment*

get_readme(*ref=NotSet*)

Calls GET /repos/:owner/:repo/readme

Parameters **ref** – string

Return type *github.ContentFile.ContentFile*

get_source_import()

Calls GET /repos/:owner/:repo/import

Return type *github.SourceImport.SourceImport*

get_stargazers()

Calls GET /repos/:owner/:repo/stargazers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_stargazers_with_dates()

Calls GET /repos/:owner/:repo/stargazers

Return type *github.PaginatedList.PaginatedList* of *github.Stargazer.Stargazer*

get_stats_contributors()

Calls GET /repos/:owner/:repo/stats/contributors

Return type None or list of *github.StatsContributor.StatsContributor*

get_stats_commit_activity()

Calls GET /repos/:owner/:repo/stats/commit_activity

Return type None or list of *github.StatsCommitActivity.StatsCommitActivity*

get_stats_code_frequency()

Calls GET /repos/:owner/:repo/stats/code_frequency

Return type None or list of *github.StatsCodeFrequency.StatsCodeFrequency*

get_stats_participation()

Calls GET /repos/:owner/:repo/stats/participation

Return type None or *github.StatsParticipation.StatsParticipation*

get_stats_punch_card()

Calls GET /repos/:owner/:repo/stats/punch_card

Return type None or *github.StatsPunchCard.StatsPunchCard*

get_subscribers()

Calls GET /repos/:owner/:repo/subscribers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_tags()

Calls GET /repos/:owner/:repo/tags

Return type *github.PaginatedList.PaginatedList* of *github.Tag.Tag*

get_releases()

Calls GET /repos/:owner/:repo/releases

Return type *github.PaginatedList.PaginatedList* of *github.GitRelease.GitRelease*

get_release(*id*)

Calls GET /repos/:owner/:repo/releases/:id

Parameters **id** – int (release id), str (tag name)

Return type None or *github.GitRelease.GitRelease*

get_latest_release()

Calls GET /repos/:owner/:repo/releases/latest

Return type *github.GitRelease.GitRelease*

get_teams()

Calls GET /repos/:owner/:repo/teams

Return type *github.PaginatedList.PaginatedList* of *github.Team.Team*

get_topics()

Calls GET /repos/:owner/:repo/topics

Return type list of strings

get_watchers()

Calls GET /repos/:owner/:repo/watchers

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

has_in_assignees(*assignee*)

Calls GET /repos/:owner/:repo/assignees/:assignee

Parameters **assignee** – string or *github.NamedUser.NamedUser*

Return type bool

has_in_collaborators(*collaborator*)

Calls GET /repos/:owner/:repo/collaborators/:user

Parameters **collaborator** – string or *github.NamedUser.NamedUser*

Return type bool

legacy_search_issues(*state*, *keyword*)

Calls GET /legacy/issues/search/:owner/:repository/:state/:keyword

Parameters

- **state** – "open" or "closed"
- **keyword** – string

Return type *github.PaginatedList.PaginatedList* of *github.Issue.Issue*

mark_notifications_as_read(*last_read_at=datetime.datetime(2019, 6, 24, 7, 15, 21, 307824)*)

Calls PUT /repos/:owner/:repo/notifications

Parameters **last_read_at** – datetime

merge(*base*, *head*, *commit_message=NotSet*)

Calls POST /repos/:owner/:repo/merges

Parameters

- **base** – string
- **head** – string
- **commit_message** – string

Return type *github.Commit.Commit*

replace_topics(*topics*)

Calls PUT /repos/:owner/:repo/topics

Parameters **topics** – list of strings

Return type None

remove_from_collaborators(*collaborator*)

Calls DELETE /repos/:owner/:repo/collaborators/:user

Parameters **collaborator** – string or *github.NamedUser.NamedUser*

Return type None

subscribe_to_hub(*event*, *callback*, *secret=NotSet*)

Calls POST /hub

Parameters

- **event** – string
- **callback** – string
- **secret** – string

Return type None

unsubscribe_from_hub(*event*, *callback*)

Calls POST /hub

Parameters

- **event** – string
- **callback** – string
- **secret** – string

Return type None

3.4.66 RepositoryKey

class github.RepositoryKey.**RepositoryKey**

This class represents RepositoryKeys. The reference can be found here http://developer.github.com/v3/repos/keys/

created_at

Type datetime.datetime

id

Type integer

key

Type string

title

Type string

url

Type string

verified

Type bool

read_only

Type bool

delete()

Calls DELETE /repos/:owner/:repo/keys/:id

Return type None

3.4.67 RequiredPullRequestReviews

class github.RequiredPullRequestReviews.**RequiredPullRequestReviews**

This class represents Required Pull Request Reviews. The reference can be found here https://developer.github.com/v3/repos/branches/#get-pull-request-review-enforcement-of-protected-branch

dismiss_stale_reviews

Type bool

require_code_owner_reviews

Type bool

required_approving_review_count

Type int

url

Type string

dismissal_users

Type list of *github.NamedUser.NamedUser*

dismissal_teams

Type list of *github.Team.Team*

3.4.68 RequiredStatusChecks

class github.RequiredStatusChecks.**RequiredStatusChecks**

This class represents Required Status Checks. The reference can be found here https://developer.github.com/v3/repos/branches/#get-required-status-checks-of-protected-branch

strict

Type bool

contexts

Type list of string

url

Type string

3.4.69 SourceImport

class github.SourceImport.**SourceImport**

This class represents SourceImports. The reference can be found here https://developer.github.com/v3/migration/source_imports/

authors_count

Type integer

authors_url

Type string

has_large_files

Type bool

html_url

Type string

large_files_count

Type integer

large_files_size

Type integer

repository_url

Type string

status

Type string

status_text

Type string

url

Type string

use_lfs

Type string

vcs

Type string

vcs_url

Type string

3.4.70 Stargazer

class github.Stargazer.**Stargazer**

This class represents Stargazers. The reference can be found here https://developer.github.com/v3/activity/starring/#alternative-response-with-star-creation-timestamps

starred_at

Type datetime.datetime

user

Type github.NamedUser

3.4.71 StatsCodeFrequency

class github.StatsCodeFrequency.**StatsCodeFrequency**

This class represents statistics of StatsCodeFrequencies. The reference can be found here http://developer.github.com/v3/repos/statistics/#get-the-number-of-additions-and-deletions-per-week

week

Type datetime.datetime

additions

Type int

deletions

Type int

3.4.72 StatsCommitActivity

class github.StatsCommitActivity.**StatsCommitActivity**

This class represents StatsCommitActivities. The reference can be found here http://developer.github.com/v3/repos/statistics/#get-the-last-year-of-commit-activity-data

week

Type datetime.datetime

total

Type int

days

Type list of int

3.4.73 StatsContributor

class github.StatsContributor.**StatsContributor**

This class represents StatsContributors. The reference can be found here http://developer.github.com/v3/repos/statistics/#get-contributors-list-with-additions-deletions-and-commit-counts

class Week(*requester*, *headers*, *attributes*, *completed*)

This class represents weekly statistics of a contibutor.

w

Type datetime.datetime

a

Type int

d

Type int

c

Type int

author

Type *github.NamedUser.NamedUser*

total

Type int

weeks

Type list of *Week*

3.4.74 StatsParticipation

class github.StatsParticipation.**StatsParticipation**

This class represents StatsParticipations. The reference can be found here http://developer.github.com/v3/repos/statistics/#get-the-weekly-commit-count-for-the-repo-owner-and-everyone-else

all

Type list of int

owner

Type list of int

3.4.75 StatsPunchCard

class github.StatsPunchCard.**StatsPunchCard**

This class represents StatsPunchCards. The reference can be found here http://developer.github.com/v3/repos/statistics/#get-the-number-of-commits-per-hour-in-each-day

get(*day*, *hour*)

Get a specific element :param day: int :param hour: int :rtype: int

3.4.76 Tag

class github.Tag.**Tag**

This class represents Tags. The reference can be found here https://developer.github.com/v3/repos/#list-tags

commit

Type *github.Commit.Commit*

name

Type string

tarball_url

Type string

zipball_url

Type string

3.4.77 Team

class github.Team.**Team**

This class represents Teams. The reference can be found here http://developer.github.com/v3/orgs/teams/

id

Type integer

members_count

Type integer

members_url

Type string

name

Type string

description

Type string

permission

Type string

repos_count

Type integer

repositories_url

Type string

slug

Type string

url

Type string

organization

Type *github.Organization.Organization*

privacy

Type string

add_to_members(*member*)

This API call is deprecated. Use *add_membership* instead. https://developer.github.com/v3/teams/members/#deprecation-notice-1

Calls PUT /teams/:id/members/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type None

add_membership(*member*, *role=NotSet*)

Calls PUT /teams/:id/memberships/:user

Parameters

- **member** – github.Nameduser.NamedUser
- **role** – string

Return type None

add_to_repos(*repo*)

Calls PUT /teams/:id/repos/:org/:repo

Parameters **repo** – *github.Repository.Repository*

Return type None

set_repo_permission(*repo*, *permission*)

Calls PUT /teams/:id/repos/:org/:repo

Parameters

- **repo** – *github.Repository.Repository*
- **permission** – string

Return type None

delete()

Calls DELETE /teams/:id

Return type None

edit(*name*, *description=NotSet*, *permission=NotSet*, *privacy=NotSet*)

Calls PATCH /teams/:id

Parameters

- **name** – string
- **description** – string
- **permission** – string
- **privacy** – string

Return type None

get_members(*role=NotSet*)

Calls GET /teams/:id/members

Parameters **role** – string

Return type *github.PaginatedList.PaginatedList* of *github.NamedUser.NamedUser*

get_repos()

Calls GET /teams/:id/repos

Return type *github.PaginatedList.PaginatedList* of *github.Repository.Repository*

has_in_members(*member*)

Calls GET /teams/:id/members/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type bool

has_in_repos(*repo*)

Calls GET /teams/:id/repos/:owner/:repo

Parameters **repo** – *github.Repository.Repository*

Return type bool

remove_membership(*member*)

Calls *DELETE /teams/:team_id/memberships/:username <https://developer.github.com/v3/teams/members/#remove-team-membership>*

Parameters **member** –

Returns

remove_from_members(*member*)

This API call is deprecated. Use *remove_membership* instead: https://developer.github.com/v3/teams/members/#deprecation-notice-2

Calls DELETE /teams/:id/members/:user

Parameters **member** – *github.NamedUser.NamedUser*

Return type None

remove_from_repos(*repo*)

Calls DELETE /teams/:id/repos/:owner/:repo

Parameters **repo** – *github.Repository.Repository*

Return type None

3.4.78 Topic

class github.Topic.**Topic**

This class represents topics as used by https://github.com/topics. The object refereence can be found here https://developer.github.com/v3/search/#search-topics

name

Type string

display_name

Type string

short_description

Type string

description

Type string

3.4.79 UserKey

class github.UserKey.**UserKey**

This class represents UserKeys. The reference can be found here http://developer.github.com/v3/users/keys/

id

Type integer

key

Type string

title

Type string

url

Type string

verified

Type bool

delete()

Calls DELETE /user/keys/:id

Return type None

3.4.80 View

class github.View.**View**

This class represents a popular Path for a GitHub repository. The reference can be found here https://developer.github.com/v3/repos/traffic/

timestamp

Type datetime.datetime

count

Type integer

uniques

Type integer

CHAPTER 4

Change log

4.1 Stable versions

4.1.1 Version 1.43.7 (April 16, 2019)

- Exclude tests from PyPI distribution (#1031) (78d283b9)
- Add codecov badge (#1090) (4c0b54c0)

4.1.2 Version 1.43.6 (April 05, 2019)

New features

- Add support for Python 3.7 (#1028) (6faa00ac)
- Adding HTTP retry functionality via urllib3 (#1002) (5ae7af55)
- Add new dismiss() method on PullRequestReview (#1053) (8ef71b1b)
- Add since and before to *get_notifications* (#1074) (7ee6c417)
- Add url parameter to include anonymous contributors in *get_contributors* (#1075) (293846be)
- Provide option to extend expiration of jwt token (#1068) (86a9d8e9)

Bug Fixes & Improvements

- Fix the default parameter for *PullRequest.create_review* (#1058) (118def30)
- Fix *get_access_token* (#1042) (6a89eb64)
- Fix *Organization.add_to_members* role passing (#1039) (480f91cf)

Deprecation

- Remove Status API (6efd6318)

4.1.3 Version 1.43.5 (January 29, 2019)

- Add project column create card (#1003) (5f5c2764)
- Fix request got an unexpected keyword argument body (#1012) (ff789dcc)
- Add missing import to PullRequest (#1007) (b5122768)

4.1.4 Version 1.43.4 (December 21, 2018)

New features

- Add Migration API (#899) (b4d895ed)
- Add Traffic API (#977) (a433a2fe)
- New in Project API: create repository project, create project column (#995) (1c0fd97d)

Bug Fixes & Improvements

- Change type of GitRelease.author to NamedUser (#969) (aca50a75)
- Use total_count from data in PaginatedList (#963) (ec177610)

4.1.5 Version 1.43.3 (October 31, 2018)

New features

- Add support for JWT authentication (#948) (8ccf9a94)
- Added support for required signatures on protected branches (#939) (8ee75a28)
- Ability to filter repository collaborators (#938) (5687226b)
- Mark notification as read (#932) (0a10d7cd)
- Add highlight search to `search_code` function (#925) (1fa25670)
- Adding `suspended_at` property to NamedUSer (#922) (c13b43ea)
- Add since parameter for Gists (#914) (e18b1078)

Bug Fixes & Improvements

- Fix missing parameters when reversing `PaginatedList` (#946) (60a684c5)
- Fix unable to trigger `RateLimitExceededException`. (#943) (972446d5)
- Fix inconsistent behavior of trailing slash usage in file path (#931) (ee9f098d)
- Fix handling of 301 redirects (#916) (6833245d)
- Fix missing attributes of `get_repos` for authenticated users (#915) (c411196f)
- Fix `Repository.edit` (#904) (7286eec0)
- Improve `__repr__` method of Milestone class (#921) (562908cb)
- Fix rate limit documentation change (#902) (974d1ec5)
- Fix comments not posted in create_review() (#909) (a18eeb3a)

4.1.6 Version 1.43.2 (September 12, 2018)

- Restore `RateLimit.rate` attribute, raise deprecation warning instead (d92389be)

4.1.7 Version 1.43.1 (September 11, 2018)

New feature:

- Add support for Projects (#854) (faca4ce1)

4.1.8 Version 1.43 (September 08, 2018)

BUGFIX

- `Repository.get_archive_link` will now NOT follow HTTP redirect and return the url instead (#858) (43d325a5)
- Fixed `Gistfile.content` (#486) (e1df09f7)
- Restored NamedUser.contributions attribute (#865) (b91dee8d)

New features

- Add support for repository topics (#832) (c6802b51)
- Add support for required approving review count (#888) (ef16702)
- Add `Organization.invite_user` (880)(eb80564)
- Add support for search/graphql rate limit (fd8a036)
  - Deprecated `RateLimit.rate`
  - Add RateLimit.core, RateLimit.search and RateLimit.graphql
- Add Support search by topics (#893) (3ce0418)
- Branch Protection API overhaul (#790) (171cc567)
  - (**breaking**) Removed Repository.protect_branch
  - Add BranchProtection
  - Add RequiredPullRequestReviews
  - Add RequiredStatusChecks
  - Add `Branch.get_protection`, `Branch.get_required_pull_request_reviews`, `Branch.get_required_status_checks`, etc

Improvements

- Add missing arguments to `Repository.edit` (#844) (29d23151)
- Add missing attributes to Repository (#842) (2b352fb3)
- Adding archival support for `Repository.edit` (#843) (1a90f5db)
- Add `tag_name` and `target_commitish` arguments to `GitRelease.update_release` (#834) (790f7dae)
- Allow editing of Team descriptions (#839) (c0021747)
- Add description to Organizations (#838) (1d918809)

- Add missing attributes for IssueEvent (#857) (7ac2a2a)
- Change `MainClass.get_repo` default laziness (#882) (6732517)

Deprecation

- Removed Repository.get_protected_branch (#871) (49db6f8)

4.1.9 Version 1.42 (August 19, 2018)

- Fix travis upload issue

BUGFIX

- `Repository.get_archive_link` will now NOT follow HTTP redirect and return the url instead (#858) (43d325a5)
- Fixed `Gistfile.content` (#486) (e1df09f7)
- Restored NamedUser.contributions attribute (#865) (b91dee8d)

New features

- Add support for repository topics (#832) (c6802b51)
- Branch Protection API overhaul (#790) (171cc567)
  - (**breaking**) Removed Repository.protect_branch
  - Add BranchProtection
  - Add RequiredPullRequestReviews
  - Add RequiredStatusChecks
  - Add `Branch.get_protection`, `Branch.get_required_pull_request_reviews`, `Branch.get_required_status_checks`, etc

Improvements

- Add missing arguments to `Repository.edit` (#844) (29d23151)
- Add missing properties to Repository (#842) (2b352fb3)
- Adding archival support for `Repository.edit` (#843) (1a90f5db)
- Add `tag_name` and `target_commitish` arguments to `GitRelease.update_release` (#834) (790f7dae)
- Allow editing of Team descriptions (#839) (c0021747)
- Add description to Organizations (#838) (1d918809)

4.1.10 Version 1.41 (August 19, 2018)

BUGFIX

- `Repository.get_archive_link` will now NOT follow HTTP redirect and return the url instead (#858) (43d325a5)
- Fixed `Gistfile.content` (#486) (e1df09f7)
- Restored NamedUser.contributions attribute (#865) (b91dee8d)

New features

- Add support for repository topics (#832) (c6802b51)
- Branch Protection API overhaul (#790) (171cc567)
  - (**breaking**) Removed Repository.protect_branch
  - Add BranchProtection
  - Add RequiredPullRequestReviews
  - Add RequiredStatusChecks
  - Add `Branch.get_protection`, `Branch.get_required_pull_request_reviews`, `Branch.get_required_status_checks`, etc

Improvements

- Add missing arguments to `Repository.edit` (#844) (29d23151)
- Add missing properties to Repository (#842) (2b352fb3)
- Adding archival support for `Repository.edit` (#843) (1a90f5db)
- Add `tag_name` and `target_commitish` arguments to `GitRelease.update_release` (#834) (790f7dae)
- Allow editing of Team descriptions (#839) (c0021747)
- Add description to Organizations (#838) (1d918809)

4.1.11 Version 1.40 (June 26, 2018)

- Major enhancement: use requests for HTTP instead of httplib (#664) (9aed19dd)
- Test Framework improvement (#795) (faa8f205)
- Handle HTTP 202 HEAD & GET with a retry (#791) (3aead158)
- Fix github API requests after asset upload (#771) (8bdac23c)
- Add remove_membership() method to Teams class (#807) (817f2230)
- Add check-in to projects using PyGithub (#814) (05f49a59)
- Include target_commitish in GitRelease (#788) (ba5bf2d7)
- Fix asset upload timeout, increase default timeout from 10s to 15s (#793) (140c6480)
- Fix Team.description (#797) (0e8ae376)
- Fix Content-Length invalid headers exception (#787) (23395f5f)
- Remove NamedUser.contributions (#774) (a519e467)
- Add ability to skip SSL cert verification for Github Enterprise (#758) (85a9124b)
- Correct Repository.get_git_tree recursive use (#767) (bd0cf309)
- Re-work PullRequest reviewer request (#765) (e2e29918)
- Add support for team privacy (#763) (1f23c06a)
- Add support for organization outside collaborators (#533) (c4446996)
- PullRequest labels should use Issues URL (#754) (678b6b20)
- Support labels for PullRequests (#752) (a308dc92)
- Add get_organizations() (#748) (1e0150b5)

4.1.12 Version 1.39 (April 10, 2018)

- Add documentation to github.Repository.Repository.create_git_release() (#747) (a769c2ff)
- Add add_to_members() and remove_from_membership() (#741) (4da483d1)
- Documentation: clarify semantics of get_comments (#743) (fec3c943)
- Add download_url to ContentFile, closes #575 (ca6fbc45)
- Add PullRequestComment.in_reply_to_id (#718) (eaa6a508)
- Add team privacy parameter to create team (#702) (5cb5ab71)
- Implement License API (#734) (b54ccc78)
- Fix delete method for RepositoryKey (911bf615)
- Remove edit for UserKey (722f2534)
- Labels API: support description (#738) (42e75938)
- Added Issue.as_pull_request() and PullReqest.as_issue() (#630) (6bf2acc7)
- Documentation: sort the Github Objects (#735) (1497e826)
- Add support for getting PR single review's comments. (#670) (612c3500)
- Update the RepositoryKey class (#530) (5e8c6832)
- Added since to PR review comments get (#577) (d8508285)
- Remove some duplicate attributes introduced in #522 (566b28d3)
- Added tarball_url, zipball_url, prerelease and draft property (#522) (c76e67b7)
- Source Import API (#673) (864c663a)

4.1.13 Version 1.38 (March 21, 2018)

- Updated readthedocs, PyPI to reflect latest version
- Added option to create review for Pull request (#662) (162f0397)
- Depreciate legacy search API (3cd176e3)
- Filter team members by role (#491) (10ee17a2)
- Add url attribute to PullRequestReview object (#731) (0fb176fd)
- Added target_commitish option to Repository.create_git_release() (#625) (0f0a7d4e)
- Fix broken Github reference link in class docstrings (a32a17bf)
- Add hook support for organizations (#729) (c7f6563c)
- Get organization from the team (#590) (d9c5a07f)
- Added search_commits (#727) (aa556f85)
- Collaborator site admin (#719) (f8b23505)
- Fix add_to_watched for AuthenticatedUser (#716) (6109eb3c)

4.1.14 Version 1.37 (March 03, 2018)

- Add __eq__ and __hash__ to NamedUser (#706) (8a13b274)
- Add maintainer can modify flag to create pull request (#703) (0e5a1d1d)
- Fix typo in Design.md (#701) (98d32af4)
- Add role parameter to Team.add_membership method (#638) (01ab4cc6)
- Add add_membership testcase (#637) (5a1424bb)

4.1.15 Version 1.36 (February 02, 2018)

- Fix changelog generation (5d911e22)
- Add collaborator permission support (#699) (167f85ef)
- Use datetime object in create_milestone (#698) (cef98416)
- Fix date format for milestone creation (#593) (e671fdd0)
- Remove the default “null” input send during GET request (#691) (cbfe8d0f)
- Updated PullRequest reviewer request according to API changes (#690) (5c9c2f75)
- make created_at/published_at attrs available for Release objects (#689) (2f9b1e01)
- Add committer/author to Repository.delete_file (#678) (3baa682c)
- Add method to get latest release of a repository (#609) (45d18436)
- Add permissions field to NamedUser (#676) (6cfe46b7)
- Fix all pep8 coding conventions (6bc804dc)
- Add new params for /users/:user/repos endpoint (89834a9b)
- Add support for changing PR head commit (#632) (3f77e537)
- Use print() syntax in README (#681) (c5988c39)
- Add PyPI badge and installation instructions to README (#682) (3726f686)
- Drop support for EOL Python 2.5-2.6 and 3.2-3.3 (#674) (6735be49)
- Add Reactions feature (#671) (ba50af53)
- Add ping_url and ping to Hook (#669) (6169d8ea)
- Add Repository.archived property (#657) (35333e03)
- Add unit test for tree attribute of GitCommit (#668) (e5bfdbeb)
- Add read_only attribute to Deploy Keys (#570) (dbc6f5ab)
- Doc create instance from token (#667) (c33a3883)
- Fix uploading binary files on Python 3 (#621) (317079ef)
- Decode jwt bytes object in Python 3 (#633) (84b43da7)
- Remove broken downloads badge (#644) (15cdc2f8)
- Added missing parameters for repo creation (#623) (5c41120a)
- Add ability to access github Release Asset API. (#525) (52449649)

- Add 'submitted at' to PullRequestReview (#565) (ebe7277a)
- Quote path for /contents API (#614) (554c1ab1)
- Add Python 3.6 (2533bed9)
- Add Python 3.6 (e78f0ece)
- Updated references in introduction.rst (d2c72bb3)
- fix failing tests on py26 (291f6dde)
- Import missing exception (67b078e9)

4.1.16 Version 1.35 (July 10, 2017)

- Add Support for repository collaborator invitations.

4.1.17 Version 1.34 (abril 04, 2017)

- Add Support for Pull Request Reviews feature.

4.1.18 Version 1.32 (February 1, 2017)

- Support for Integrations installation endpoint (656e70e1)

4.1.19 Version 1.31 (January 30, 2017)

- Support HTTP 302 redirect in Organization.has_in_members (0154c6b)
- Add details of repo type for get_repos documentation (f119147)
- Note explicit support for Python 3.5 (3ae55f0)
- Fix README instructions (5b0224e)
- An easier to see link to the documentation in response to issue #480. (6039a4b)
- Encode GithubObject repr values in utf-8 when using Python2 (8ab9082)
- Updated documentation (4304ccd)
- Added a subscribers count field (a2da7f9)
- Added "add_to_assignees" & "remove_from_assignees" method to Issue object. (66430d7)
- Added "assignees" attribute to PullRequest object. (c0de6be)
- add html_url to GitRelease (ec633aa)
- Removed unused imports (65afc3f)
- Fix typo in a constant (10a28e0)
- Fix changelog formatting glitch (03a9227)
- Added "assignees" argument in Repository.create_issue() (ba007dc)
- Enhance support of "assignees" argument in Issue.edit() (14dd9f0)
- Added "assignees" attribute to Issue object. (e0e5fdf)

4.1.20 Version 1.30 (January 30, 2017)

- adds GitHub integrations (d60943d)

4.1.21 Version 1.29 (October 10, 2016)

- add issue assignee param (3a8edc7)
- Fix diffrerent case (fcf6cfb)
- DOC: remove easy_install suggestion; update links (45e76d9)
- Add permission param documentation (9347345)
- Add ability to set permission for team repo (5dddea7)
- Fix status check (073bb44)
- adds support for content dirs (0799753)

4.1.22 Version 1.28 (September 09, 2016)

- test against python 3.5 (5d35284)
- sort params and make them work on py3 (78374b9)
- adds a nicer __repr__ (8571d87)
- Add missing space (464259d)
- Properly handle HTTP Proxy authentication with Python 3 (d015154)
- Fix small typo (987bca0)
- push to 'origin' instead of 'github' (d640666)

4.1.23 Version 1.27.1 (August 12, 2016)

- upgrade release process based on travis (3c20a66)
- change file content encoding to support unicode(like chinese), py2 (5404030)
- adds missing testfile corrections (9134aa2)
- fixed file API return values (0f29a53)
- assert by str and unicode to make it more py3 friendly (7390827)
- Patch issue 358 status context (#428) (70e30c5)
- Adding "since" param to Issue.get_comments() (#426) (3c6f99f)
- update doc url everywhere (#420) (cb0cf0a)
- fix a couple typos to be clearer (#419) (23c0e75)
- Document how one gets an AuthenticatedUser object (ba66862)
- fix wrong expectance on requestJsonAndCheck() returning {} if no data (8985368)
- Add previous_filename property to File (e1be1e6)
- add changelog entry for 1.26.0 (a1f3de2)

- update project files (be2e98b)
- fix update/create/delete file api return value issue (8bb765a)
- fix typo (a7929ac)
- fix update/delete/create content return value invalid issue (a0a4511)
- Follow redirects in the case of a 301 status code (c29f533)
- Fix for pickling exception when deserializing GithubException. (8f8b455)
- add support for the head parameter in Repository.get_pulls (397a74d)
- Add: - CommitCombinedStatus class - get_combined_status() to Commit class to return combined status - Add test for combined status. (5823ed7)
- fix python3 compatibility issue for using json/base64 (5b7f0bb)
- remove not covered API from readme (9c6f881)
- change replay data for update file test case (46895df)
- fix python3 compatability error in test case (00777db)
- Add repo content create/update/delete testcase (4aaeb9e)
- add MAINTAINERS file (a16b55b)
- travis: disable email (6347157)
- fix protect branch tests (65360b0)
- Add branch protection endpoint (737f0c3)
- fix request parameters issue (ae37d44)
- add content file create/update/delete api (b83ffbf)
- Add travis button on README. (a83649b)
- fix misspelling: https://github.com/PyGithub/PyGithub/issues/363 (a06b5ec)
- Adding base parameter to get_pulls() method. (71593a8)
- add support for the direction parameter in Repository.get_pulls (70bcb6d)
- added creator parameter (ca9af4f)

4.1.24 Version 1.27.0 (August 12, 2016)

- this version was never released to PyPi due to a problem with the deployment

4.1.25 Version 1.26.0 (November 5th, 2015)

- Added context parameter to Status API
- Changed InputGitAuthor to reflect that time is an optional parameter
- Added sort option to get_pulls
- Added api_preview parameter to Requester class
- Return empty list instead of None for pagination with no pages
- Removed URL scheme validation that broke GitHub Enterprise

- Added "add_membership" call to Teams
- Added support to lazily load repositories
- Updated test suite to record with oauth tokens
- Added support for http_proxy
- Add support for filter/role options in Organization.get_members()
- Changed Organization.get_members's filter parameter to _filter
- Fix escaping so that labels now support whitespaces
- Updated create_issue to support taking a list of strings for labels
- Added support for long integers in get_repo
- Fixed pagination to thread headers between requests
- Added repo.get_stargazers_with_dates()

4.1.26 Version 1.25.2 (October 7th, 2014)

- Work around the GitHub API v3 returning *null*s in some paginated responses, erichaase for the bug report

4.1.27 Version 1.25.1 (September 28th, 2014)

- Fix two-factor authentication header, thanks to tradej for the pull request

4.1.28 Version 1.25.0 (May 4th, 2014)

- Implement getting repos by id, thanks to tylertreat for the pull request
- Add `Gist.owner`, thanks to dalejung for the pull request

4.1.29 Version 1.24.1 (March 16th, 2014)

- Fix urlquoting in search, thanks to cro for the pull request

4.1.30 Version 1.24.0 (March 2nd, 2014)

- Implement search, thanks to thialfihar for the pull request

4.1.31 Version 1.23.0 (December 23th, 2013)

- Fix all that is based on headers in Python 3 (pagination, conditional request, rate_limit...), huge thanks to cwarren-mw for finding the bug
- Accept strings for assignees and collaborators, thanks to farrd
- Ease two-factor authentication by adding 'onetime_password' to AuthenticatedUser.create_authorization, thanks to cameronbwhite

4.1.32 Version 1.22.0 (December 15th, 2013)

- Emojis, thanks to evolvelight
- Repository.stargazers_count, thanks to cameronbwhite
- User.get_teams, thanks to poulp

4.1.33 Version 1.21.0 (November ??th, 2013)

- Accept strings as well as `Label` objects in `Issue.add_to_labels`, `Issue.remove_from_labels` and `Issue.set_labels`. Thank you acdha for asking
- Implement equality comparison for *completable* github objects (ie. those who have a `url` attribute). Warning, comparison is still not implemented for non-completable objects. This will be done in version 2.0 of PyGithub. Thank you OddBloke for asking
- Add parameter `author` to `Repository.get_commits`. Thank you naorrosenberg for asking
- Implement the statistics end points. Thank you naorrosenberg for asking

4.1.34 Version 1.20.0 (October 20th, 2013) (First Seattle edition)

- Implement `Github.get_hook(name)`. Thank you klmitch for asking
- In case bad data is returned by Github API v3, raise an exception only when the user accesses the faulty attribute, not when constructing the object containing this attribute. Thank you klmitch for asking
- Fix parameter public/private of `Repository.edit`. Thank you daireobroin449 for reporting the issue
- Remove `Repository.create_download` and `NamedUser.create_gist` as the corrensponding APIs are not documented anymore

4.1.35 Version 1.19.0 (September 8th, 2013) (AKFish's edition)

- Implement conditional requests by the method `GithubObject.update`. Thank you very much akfish for the pull request and your collaboration!
- Implement persistence of PyGithub objects: `Github.save` and `Github.load`. Don't forget to `update` your objects after loading them, it won't decrease your rate limiting quota if nothing has changed. Again, thank you akfish
- Implement `Github.get_repos` to get all public repositories
- Implement `NamedUser.has_in_following`
- Implement `Github.get_api_status`, `Github.get_last_api_status_message` and `Github.get_api_status_messages`. Thank you ruxandraburtica for asking
- Implement `Github.get_rate_limit`
- Add many missing attributes
- Technical change: HTTP headers are now stored in retrieved objects. This is a base for new functionalities. Thank you akfish for the pull request
- Use the new URL to fork gists (minor change)
- Use the new URL to test hooks (minor change)

4.1.36 Version 1.18.0 (August 21st, 2013) (Bénodet edition)

- Issues' `repository` attribute will never be `None`. Thank you stuglaser for the pull request
- No more false assumption on rate_limiting, and creation of `rate_limiting_resettime`. Thank you edjackson for the pull request
- New parameters `since` and `until` to `Repository.get_commits`. Thank you apetresc for the pull request
- Catch Json parsing exception for some internal server errors, and throw a better exception. Thank you MarkRoddy for the pull request
- Allow reversed iteration of `PaginatedList`. Thank you davidbrai for the pull request

4.1.37 Version 1.17.0 (Jully 7th, 2013) (Hamburg edition)

- Fix bug in `Repository.get_comment` when using custom `per_page`. Thank you davidbrai
- Handle Http redirects in `Repository.get_dir_contents`. Thank you MarkRoddy
- Implement API `/user` in `Github.get_users`. Thank you rakeshcusat for asking
- Improve the documentation. Thank you martinqt

4.1.38 Version 1.16.0 (May 31th, 2013) (Concarneau edition)

- Add the html_url attribute to IssueComment and PullRequestComment

4.1.39 Version 1.15.0 (May 17th, 2013) (Switzerland edition)

- Implement listing of user issues with all parameters. Thank you Daehyok Shin for reporting
- Raise two new specific exceptions

4.1.40 Version 1.14.2 (April 25th, 2013)

- Fix paginated requests when using secret-key oauth. Thank you jseabold for analysing the bug

4.1.41 Version 1.14.1 (April 25th, 2013)

- Set the default User-Agent header to "PyGithub/Python". (Github has enforced the User Agent header yesterday.) Thank you jjh42 for the fix, thank you jasenmh and pconrad for reporting the issue.

4.1.42 Version 1.14.0 (April 22nd, 2013)

- Improve gist edition. Thank you jasonwiener for asking:
  - Delete a file with `gist.edit(files={"name.txt": None})`
  - Rename a file with `gist.edit(files={"old_name.txt": github.InputFileContent(gist.files["old_name.txt"].content, new_name="new_name.txt")})`

- Raise specific exceptions. Thank you pconrad for giving me the idea

4.1.43 Version 1.13.1 (March 28nd, 2013)

- Fix login/password authentication for Python 3. Thank you sebastianstigler for reporting

4.1.44 Version 1.13.0 (March 22nd, 2013)

- Fix for Python 3 on case-insensitive file-systems. Thank you ptwobrussell for reporting
- Expose raw data returned by Github for all objects. Thank you ptwobrussell for asking
- Add a property `github.MainClass.Github.per_page` (and a parameter to the constructor) to change the number of items requested in paginated requests. Thank you again ptwobrussell for asking
- Implement the first part of the Notifications API. Thank you pgolm
- Fix automated tests on Python 3.3. Thank you bkabrda for reporting

4.1.45 Version 1.12.2 (March 3rd, 2013)

- Fix major issue with Python 3: Json decoding was broken. Thank you bilderbuchi for reporting

4.1.46 Version 1.12.1 (February 20th, 2013)

- Nothing, but packaging/upload of 1.12.0 failed

4.1.47 Version 1.12.0 (February 20th, 2013)

- Much better documentation: http://jacquev6.github.com/PyGithub
- Implement `github.Repository.Repository.get_dir_contents()`. Thank you ksookocheff-va for asking

4.1.48 Version 1.11.1 (February 9th, 2013) (London edition)

- Fix bug in lazy completion. Thank you ianozsvald for pinpointing it

4.1.49 Version 1.11.0 (February 7th, 2013)

- Fix bug in PaginatedList without url parameters. Thank you llimllib for the contribution
- Implement `github.NamedUser.NamedUser.get_keys()`
- Support PubSubHub: `github.Repository.Repository.subscribe_to_hub()` and `github.Repository.Repository.unsubscribe_from_hub()`
- Publish the oauth scopes in `github.MainClass.Github.oauth_scopes`, thank you bilderbuchi for asking

4.1.50 Version 1.10.0 (December 25th, 2012) (Christmas 2012 edition)

- Major improvement: support Python 3! PyGithub is automaticaly tested on Travis with versions 2.5, 2.6, 2.7, 3.1 and 3.2 of Python
- Add a shortcut function *github.MainClass.Github.get_repo()* to get a repo directly from its full name. thank you lwc for the contribution
- *github.MainClass.Github.get_gitignore_templates()* and *github.MainClass.Github.get_gitignore_template()* for APIs `/gitignore/templates`
- Add the optional `ref` parameter to *github.Repository.Repository.get_contents()* and *github.Repository.Repository.get_readme()*. Thank you fixxxeruk for the contribution
- Get comments for all issues and all pull requests on a repository (`GET /repos/:owner/:repo/pulls/comments`: *github.Repository.Repository.get_pulls_comments()* or *github.Repository.Repository.get_pulls_review_comments()*; `GET /repos/:owner/:repo/issues/comments`: *github.Repository.Repository.get_issues_comments()*)

4.1.51 Version 1.9.1 (November 20th, 2012)

- Fix an assertion failure when integers returned by Github do not fit in a Python `int`

4.1.52 Version 1.9.0 (November 19th, 2012)

- You can now use your client_id and client_secret to increase rate limiting without authentication
- You can now send a custom User-Agent
- PullRequest now has its 'assignee' attribute, thank you mstead
- Repository.edit now has 'default_branch' parameter
- create_repo has 'auto_init' and 'gitignore_template' parameters
- GistComment URL is changed (see http://developer.github.com/changes/2012-10-31-gist-comment-uris)
- A typo in the readme was fixed by tymofij, thank you
- Internal stuff:
  - Add encoding comment to Python files, thank you Zearin
  - Restore support of Python 2.5
  - Restore coverage measurement in setup.py test
  - Small refactoring

4.1.53 Version 1.8.1 (October 28th, 2012)

- Repository.get_git_ref prepends "refs/" to the requested references. Thank you simon-weber for noting the incoherence between documentation and behavior. If you feel like it's a breaking change, please see this issue

4.1.54 Version 1.8.0 (September 30th, 2012)

- Enable Travis CI
- Fix error 500 when json payload contains percent character (%). Thank you again quixotique for pointing that and reporting it to Github
- Enable debug logging. Logger name is *"github"*. Simple logging can be enabled by *github.enable_console_debug_logging()*. Thank you quixotique for the merge request and the advice
- Publish tests in the PyPi source archive to ease QA tests of the FreeBSD port. Thank you koobs for maintaining this port
- Switch to Semantic Versioning
- Respect pep8 Style Guide for Python Code

4.1.55 Version 1.7 (September 12th, 2012)

- Be able to clear the assignee and the milestone of an Issue. Thank you quixotique for the merge request
- Fix an AssertionFailure in *Organization.get_xxx* when using Github Enterprise. Thank you mnsanghvi for pointing that
- Expose pagination to users needing it (*PaginatedList.get_page*). Thank you kukuts for asking
- Improve handling of legacy search APIs
- Small refactoring (documentation, removal of old code generation artifacts)

4.1.56 Version 1.6 (September 8th, 2012)

- Restore support for Python 2.5
- Implement new APIS:
  - /hooks (undocumented, but mentioned in http://developer.github.com/v3/repos/hooks/#create-a-hook)
  - Merging
  - Starring and subscriptions
  - Assignees
  - Commit statuses
  - Contents, thank you berndca for asking
- Clarify issue and review comments on PullRequest, thank you nixoz2k7 for asking

4.1.57 Version 1.5 (September 5th, 2012)

- Add a timeout option, thank you much xobb1t for the merge request. *This drops Python 2.5 support*. I may be able to restore it in next version.
- Implement *Repository.delete*, thank you pmchen for asking

4.1.58 Version 1.4 (August 4th, 2012)

- Allow connection to a custom Github URL, for Github Enterprise, thank you very much engie for the merge request

4.1.59 Version 1.3 (July 13th, 2012)

- Implement markdown rendering
- *GitAuthor.date* is now a datetime, thank you bilderbuchi
- Fix documentation of *Github.get_gist*: *id* is a string, not an integer

4.1.60 Version 1.2 (June 29th, 2012)

- Implement legacy search APIs, thank you kukuts for telling me Github had released them
- Fix a bug with issue labels containing spaces, thank you philipkimmey for detecting the bug and fixing it
- Clarify how collections of objects are returned by *get_** methods, thank you bilderbuchi for asking

4.1.61 Version 1.1 (June 20th, 2012)

- Restore compatibility with Python 2.5, thank you pmuilu
- Use *package_data* instead of *data_files* for documentation files in *setup.py*, thank you malexw for reporting

4.1.62 Version 1.0 (June 3rd, 2012)

- Complete rewrite, with no more complicated meta-description
- Full typing of attributes and parameters
- Full documentation of attributes and parameters
- More usable exceptions raised in case on problems with the API
- Some bugs and limitations fixed, special thanks to bilderbuchi, roskakori and tallforasmurf for reporting them!

4.2 Pre-release versions

4.2.1 Version 0.7 (May 26th, 2012)

- Use PyGithub with OAuth authentication or with no authentication at all

4.2.2 Version 0.6 (April 17th, 2012)

- Fix issue 21 (KeyError when accessing repositories)
- Re-completed the API with NamedUser.create_gist

4.2.3 Version 0.5 (March 19th, 2012)

- Major achievement: **all APIs are implemented**
- More refactoring, of course

4.2.4 Version 0.4 (March 12th, 2012)

- The list of the not implemented APIs is shorter than the list of the implemented APIs
- APIs *not implemented*:
  - GET */gists/public*
  - GET */issues*
  - GET */repos/:owner/:repo/compare/:base...:head*
  - GET */repos/:owner/:repo/git/trees/:sha?recursive-1*
  - POST */repos/:owner/:repo/git/trees?base_tree-*
- Gists
- Autorizations
- Keys
- Hooks
- Events
- Merge pull requests
- More refactoring, one more time

4.2.5 Version 0.3 (February 26th, 2012)

- More refactoring
- Issues, milestones and their labels
- NamedUser:
  - emails
- Repository:
  - downloads
  - tags, branches, commits and comments (not the same as "Git objects" of version 0.2)
  - pull requests (no automatic merge yet)
- Automatic generation of the reference documentation of classes, with less "see API"s, and less errors

4.2.6 Version 0.2 (February 23rd, 2012)

- Refactoring
- Teams details and modification
  - basic attributes

    - list teams in organizations, on repositories
- Git objects
    - create and get tags, references, commits, trees, blobs
    - list and edit references

4.2.7 Version 0.1 (February 19th, 2012)

- User details and modification
    - basic attributes
    - followers, following, watching
    - organizations
    - repositories
- Repository details and modification
    - basic attributes
    - forking
    - collaborators, contributors, watchers
- Organization details and modification
    - basic attributes
    - members and public members

Python Module Index

g

Index

B

C

D

E

F

G

H

K

L

M

N

O

P

R

S

T

U

V

W

Z