
python-taiga Documentation

Release 1.3.3

Nephila

Aug 25, 2025

CONTENTS

1	python-taiga	1
1.1	Install	1
1.2	Getting Started	1
1.2.1	API Credentials	1
1.2.2	Get projects, user stories, task and issues	2
1.2.3	Create a project	2
1.2.4	Duplicate an existing project	2
1.2.5	Create a new user story	3
1.2.6	Create a swimlane	3
1.2.7	Create an issue	3
1.2.8	Create a custom attribute	3
1.2.9	List elements	3
1.2.10	Attach a file	4
1.2.11	Play with instances	5
1.2.12	Search	5
1.2.13	History	5
1.3	TaigaAPI documentation	5
1.4	Models documentation	6
1.5	Development & community	62
1.6	Contributing	62
1.6.1	Types of Contributions	62
1.6.2	Development tips	63
1.6.3	Testing tips	63
1.7	History	65
1.7.1	1.3.3 (2025-08-25)	65
1.7.2	1.3.2 (2025-01-22)	65
1.7.3	1.3.1 (2025-01-13)	65
1.7.4	1.3.0 (2024-01-02)	65
1.7.5	1.2.0 (2023-08-23)	65
1.7.6	1.1.0 (2023-04-23)	66
1.7.7	1.0.0 (2019-08-08)	66
1.7.8	0.9.0 (2018-01-31)	66
1.7.9	0.8.6 (2016-08-26)	66
1.8	Indices and tables	67
	Python Module Index	69
	Index	71

PYTHON-TAIGA

A python wrapper for the [Taiga REST API](#).

Documentation: <https://python-taiga.readthedocs.io/>

Usage: : <https://python-taiga.readthedocs.io/usage.html>

Warning

Version 1.1 change the signature of HistoryItem methods. Check the [documentation](#) and update your code if it uses kwargs for HistoryEntity.delete_comment / HistoryEntity.undelete_comment.

1.1 Install

```
pip install python-taiga
```

1.2 Getting Started

Getting started with the Taiga API couldn't be easier: create a TaigaAPI and you're ready to go.

Note

python-taiga is a python wrapper for the [Taiga REST API](#). Any data structure, argument and response matches exactly the Taiga REST API, so refer to it for any usage.

1.2.1 API Credentials

The TaigaAPI needs your Taiga credentials. You can pass these directly to the auth method (see the code below).

```
from taiga import TaigaAPI

api = TaigaAPI()

api.auth(
    username='user',
    password='psw'
)
```

Alternately, you can pass a token to the constructor TaigaAPI constructor.

```
from taiga import TaigaAPI

api = TaigaAPI(token='mytoken')
```

You can also specify a different host if you use Taiga somewhere else

```
from taiga import TaigaAPI

api = TaigaAPI(
    host='http://taiga.my.host.org'
)
```

To use LDAP or other authentication backends, use `auth_type` argument

```
from taiga import TaigaAPI

api = TaigaAPI(
    host='http://taiga.my.host.org',
    auth_type='ldap'
)
```

To ignore SSL certificate verification (use at your own risk!) use `tls_verify` argument

```
from taiga import TaigaAPI

api = TaigaAPI(
    host='http://taiga.my.host.org',
    tls_verify=False
)
```

1.2.2 Get projects, user stories, task and issues

You can get projects, user stories, tasks and issues using the primary key or using slug/ref

```
new_project = api.projects.get_by_slug('nephila')
print(new_project.get_issue_by_ref(1036))
print(new_project.get_userstory_by_ref(1111))
print(new_project.get_task_by_ref(1112))
```

1.2.3 Create a project

```
new_project = api.projects.create('TEST PROJECT', 'TESTING API')
```

1.2.4 Duplicate an existing project

If you have a project, you can duplicate it (duplicates most of the settings, but not the user stories or tasks)

```
old_project = api.projects.get_by_slug('nephila')
new_project = old_project.duplicate('TEST PROJECT', 'TESTING API')
```

1.2.5 Create a new user story

```
userstory = new_project.add_user_story(
    'New Story', description='Blablalabla'
)
```

You can also create a milestone and pass it to a story

```
jan_feb_milestone = new_project.add_milestone(
    'MILESTONE 1', '2015-01-26', '2015-02-26'
)

userstory = new_project.add_user_story(
    'New Story', description='Blablalabla',
    milestone=jan_feb_milestone.id
)
```

To add a task to your user story just run

```
userstory.add_task(
    'New Task 2',
    new_project.task_statuses[0].id
)
```

1.2.6 Create a swimlane

```
newlane = new_project.add_swimlane('New Swimlane')
```

1.2.7 Create an issue

```
newissue = new_project.add_issue(
    'New Issue',
    new_project.priorities.get(name='High').id,
    new_project.issue_statuses.get(name='New').id,
    new_project.issue_types.get(name='Bug').id,
    new_project.severities.get(name='Minor').id,
    description='Bug #5'
)
```

1.2.8 Create a custom attribute

```
new_project.add_issue_attribute(
    'Device', description='(iPad, iPod, iPhone, Desktop, etc.)'
)
newissue.set_attribute('1', 'Desktop')
```

1.2.9 List elements

```
projects = api.projects.list()
stories = api.user_stories.list()
```

You can also specify filters

```
tasks = api.tasks.list(project=1)
```

By default list returns all objects, eventually getting the paginated results behind the scenes.

Pagination

Pagination is controlled by three parameters as explained below:

paginate	page_size (default: 100)	page	Output
True (default)	<integer>	None	All results retrieved by using paginated results and loading them behind the scenes, using given page size (higher page size could yield better performances)
True (default)	<integer>	<integer>	Only results for the given page of the given size are retrieved
False	unused	unused	Current behavior: all results, ignoring pagination

Note

non numerical or false *page_size* values is casted to the default value

Examples

No pagination

```
tasks = api.tasks.list(paginate=False)
```

Warning

be aware that the unpaginated results may exceed the data the parser can handle and may result in an error.

Retrieve a single page

```
tasks_page_1 = api.tasks.list(page=1) # Will only return page 1
```

Specify the page size

```
tasks_page_1 = api.tasks.list(page=1, page_size=200) # Will 200 results from page 1
```

1.2.10 Attach a file

You can attach files to issues, user stories and tasks

```
newissue.attach('README.md', description='Read the README in Issue')
```

1.2.11 Play with instances

Instances can have actions, for example you can star a project just calling

```
new_project = api.projects.create('TEST PROJECT', 'TESTING API')
new_project.star()
```

Any instance can be updated and deleted

```
new_project.name = 'New name for my project'
new_project.update()
new_project.delete()
```

1.2.12 Search

Search function returns a SearchResult object, containing tasks, user stories and issues:

```
projects = api.projects.list()
search_result = api.search(projects[0].id, 'NEW')
for user_story in search_result.user_stories:
    print (user_story)
```

1.2.13 History

You can access the history of issues, tasks, userstories and wiki pages:

```
history = api.history.user_story.get(user_story.id)
```

1.3 TaigaAPI documentation

Contents:

```
class taiga.client.TaigaAPI(host='https://api.taiga.io', token=None, token_type='Bearer', tls_verify=True,
                           auth_type='normal', proxies=None)
```

Bases: object

TaigaAPI class

Parameters

- **host** – the host of your Taiga.io instance
- **token** – the token you may provide
- **token_type** – the token type
- **tls_verify** – verify server certificate
- **auth_type** – authentication type identifier
- **proxies** – a dictionary of proxies to use for requests

```
auth(username, password)
```

Authenticate you

Parameters

- **username** – your username

- **password** – your password

auth_app(*app_id*, *app_secret*, *auth_code*, *state=""*)

Authenticate an app

Parameters

- **app_id** – the app id
- **app_secret** – the app secret
- **auth_code** – the app auth code

me()

Get a *taiga.models.models.User* representing me

refresh_token(*token_refresh=""*)

Refresh auth token.

Passing a token_refresh will use passed token, otherwise it will try to use self.token_refresh.

Parameters

token_refresh – the refresh token to be used to refresh api token

search(*project*, *text=""*)

Search in your Taiga.io instance

Parameters

- **project** – the project id
- **text** – the query of your search

1.4 Models documentation

Contents:

class taiga.models.models.**Attachment**(*requester*, ***params*)

Bases: *InstanceResource*

Attachment base class

Parameters

- **object_id** – object_id of *Attachment*
- **project** – project of *Attachment*
- **attached_file** – attached_file of *Attachment*
- **description** – description of *Attachment*
- **is_deprecated** – is_deprecated of *Attachment*

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester*, *entry*)

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(args)**

Update the current InstanceResource

class taiga.models.models.**Attachments**(*requester*)

Bases: [ListResource](#)

Attachments factory base class

create(*project, object_id, attached_file, **attrs*)

Create a new [Attachment](#).

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the [Attachment](#)

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod **parse**(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**CommentableResource**(*requester, **params*)

Bases: [InstanceResource](#)

CommentableResource base class

add_comment(*comment*)

Add a comment to the current element

Parameters

comment – the comment you want to insert

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**CustomAttribute**(*requester, **params*)

Bases: *InstanceResource*

CustomAttribute base class

Parameters

- **requester** – Requester instance
- **name** – name of the custom attribute
- **description** – id of the current object
- **order** – order of the custom attribute
- **project** – *Project* id

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**CustomAttributeResource**(*requester, **params*)

Bases: *InstanceResource*

CustomAttributeResource base class

delete(*query=None*)

Delete the current InstanceResource

get_attributes()

Get all the attributes of the current object

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

set_attribute(*id, value, version=1*)

Set attribute to a specific value

Parameters

- **id** – id of the attribute
- **value** – value of the attribute
- **version** – version of the attribute (default = 1)

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**CustomAttributes**(*requester*)

Bases: *ListResource*

CustomAttributes factory base class

create(*project, name, **attrs*)

Create a new *CustomAttribute*.

Parameters

- **project** – *Project* id
- **name** – name of the custom attribute
- **attrs** – optional attributes of the custom attributes

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(entries)

Parse a JSON array into a list of model instances.

class `taiga.models.models.Epic(requester, **params)`

Bases: *CustomAttributeResource*, *CommentableResource*

Epic model

Parameters

- **assigned_to** – assigned to property of *Epic*
- **blocked_note** – blocked note of *Epic*
- **description** – description of *Epic* (not available in the *Epics.list()* response)
- **is_blocked** – is blocked property of *Epic*
- **is_closed** – is closed property of *Epic*
- **color** – the color of *Epic*
- **project** – the project of *TaskStatus*
- **subject** – subject of *TaskStatus*
- **tags** – tags of *TaskStatus*
- **watchers** – watchers of *TaskStatus*
- **version** – version of *Epic*

add_comment(comment)

Add a comment to the current element

Parameters

comment – the comment you want to insert

attach(attached_file, **attrs)

Attach a file to the *Epic*

Parameters

- **attached_file** – file path to attach
- **attrs** – optional attributes for the attached file

delete(query=None)

Delete the current InstanceResource

get_attributes()

Get all the attributes of the current object

list_attachments()

Get a list of *EpicAttachment*.

list_user_stories(queryparams)**

Returns the *UserStory* list of the project.

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

set_attribute(*id*, *value*, *version=1*)

Set attribute to a specific value

Parameters

- **id** – id of the attribute
- **value** – value of the attribute
- **version** – version of the attribute (default = 1)

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.EpicAttachment(requester, **params)`

Bases: [Attachment](#)

EpicAttachment class

delete(*query=None*)

Delete the current InstanceResource

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.EpicAttachments(requester)`

Bases: [Attachments](#)

EpicAttachments factory class

create(*project*, *object_id*, *attached_file*, ***attrs*)

Create a new [Attachment](#).

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the [Attachment](#)

instance

alias of *EpicAttachment*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.EpicAttribute(*requester, **params*)

Bases: *CustomAttribute*

EpicAttribute model

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.EpicAttributes(*requester*)

Bases: *CustomAttributes*

EpicAttributes factory

create(*project, name, **attrs*)

Create a new *CustomAttribute*.

Parameters

- **project** – *Project* id
- **name** – name of the custom attribute
- **attrs** – optional attributes of the custom attributes

instance

alias of *EpicAttribute*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**EpicStatus**(*requester, **params*)

Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Taiga Epic Status model

Parameters

- **color** – the color of *EpicStatus*
- **is_closed** – closed property of *EpicStatus*
- **name** – The name of *EpicStatus*
- **order** – order of *EpicStatus*
- **project** – the Taiga project of *EpicStatus*
- **slug** – the slug of *EpicStatus*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**EpicStatuses**(*requester*)

Bases: *MoveOnDestroyMixinList, ListResource*

create(*project, name, **attrs*)

Create a new *EpicStatus*.

Parameters

- **project** – *Project* id
- **name** – name of *EpicStatus*
- **attrs** – optional attributes of *EpicStatus*

instance

alias of *EpicStatus*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**Epics**(*requester*)

Bases: *ListResource*

Epics factory class

create(*project*, *subject*, ****attrs**)

Create a new *Epic*.

Parameters

- **project** – *Project* id
- **subject** – subject of *Epic*
- **attrs** – optional attributes of *Epic*

instance

alias of *Epic*

list(*pagination=True*, *page_size=None*, *page=None*, ****queryparams**)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**History**(*args, ****kwargs**)

Bases: *InstanceResource*

History model

delete(*query=None*)

Delete the current *InstanceResource*

classmethod parse(*requester*, *entry*)

Turns a JSON object into a model instance.

patch(*fields*, ****args**)

Patch the current *InstanceResource*

to_dict()

Get a dictionary representation of InstanceResource

update(args)**

Update the current InstanceResource

class taiga.models.models.**HistoryEntity**(*requester*)

Bases: object

HistoryEntity model

delete_comment(*resource_id, comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

resource_id – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undelete_comment(*resource_id, comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**HistoryEpic**(*args, **kwargs)

Bases: *HistoryEntity*

HistoryEpic model

delete_comment(*resource_id, comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

resource_id – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undeletemcomment(*resource_id, comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**HistoryIssue**(*args, **kwargs)

Bases: *HistoryEntity*

HistoryIssue model

deletemcomment(*resource_id, comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undeletemcomment(*resource_id, comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**HistoryTask**(*args, **kwargs)

Bases: *HistoryEntity*

HistoryTask model

deletemcomment(*resource_id, comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undeleate_comment(*resource_id*, *comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**HistoryUserStory**(*args, **kwargs)

Bases: *HistoryEntity*

HistoryUserStory model

delete_comment(*resource_id*, *comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undeleate_comment(*resource_id*, *comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**HistoryWiki**(*args, **kwargs)

Bases: *HistoryEntity*

HistoryWiki model

delete_comment(*resource_id*, *comment_id*)

Delete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to delete

get(*resource_id*)

Get a history element

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)

undelecomment(*resource_id, comment_id*)

Undelete a comment

Parameters

- **resource_id** – id of the resource object (resource type is defined by the HistoryEntity subclass used)
- **comment_id** – id of the comment to undelete

class taiga.models.models.**Issue**(*requester, **params*)

Bases: *CustomAttributeResource, CommentableResource*

Issue model

Parameters

- **requester** – Requester instance
- **assigned_to** – *User* id this issue is assigned to
- **description** – description of the issue (not available in the *Issues.list()* response)
- **is_blocked** – set if this issue is blocked or not
- **milestone** – *Milestone* id
- **project** – *Project* id
- **status** – Status id
- **severity** – class:*Severity* id
- **priority** – class:*Priority* id
- **type** – class:*Type* id
- **subject** – subject of the issue
- **tags** – array of tags
- **watchers** – array of watchers id
- **due_date** – *Issue* due date

add_comment(*comment*)

Add a comment to the current element

Parameters

comment – the comment you want to insert

attach(*attached_file, **attrs*)

Attach a file to the *Issue*

Parameters

- **attached_file** – file path to attach
- **attrs** – optional attributes for the attached file

delete(*query=None*)

Delete the current InstanceResource

downvote()

Downvote *Issue*.

get_attributes()

Get all the attributes of the current object

list_attachments()

Get a list of *IssueAttachment*.

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

set_attribute(*id, value, version=1*)

Set attribute to a specific value

Parameters

- **id** – id of the attribute
- **value** – value of the attribute
- **version** – version of the attribute (default = 1)

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

upvote()

Upvote *Issue*.

class taiga.models.models.**IssueAttachment**(*requester, **params*)

Bases: *Attachment*

IssueAttachment model

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**IssueAttachments**(*requester*)

Bases: *Attachments*

IssueAttachments factory

create(*project, object_id, attached_file, **attrs*)

Create a new *Attachment*.

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the *Attachment*

instance

alias of *IssueAttachment*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.IssueAttribute(*requester, **params*)

Bases: *CustomAttribute*

IssueAttribute model

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**IssueAttributes**(*requester*)

Bases: *CustomAttributes*

IssueAttributes factory

create(*project*, *name*, ****attrs**)

Create a new *CustomAttribute*.

Parameters

- **project** – *Project* id
- **name** – name of the custom attribute
- **attrs** – optional attributes of the custom attributes

instance

alias of *IssueAttribute*

list(*pagination=True*, *page_size=None*, *page=None*, ****queryparams**)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod **parse**(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**IssueStatus**(*requester*, ****params**)

Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Issue Status model

Parameters

- **name** – name of *IssueStatus*
- **color** – color of *IssueStatus*
- **order** – order of *IssueStatus*
- **project** – the taiga project of *IssueStatus*
- **is_closed** – is closed property of *IssueStatus*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**IssueStatuses**(*requester*)

Bases: *MoveOnDestroyMixinList, ListResource*

IssueStatuses factory

instance

alias of *IssueStatus*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**IssueType**(*requester, **params*)

Bases: *MoveOnDestroyMixinObject, InstanceResource*

IssueType model

Parameters

- **name** – name of *IssueType*

- **color** – color of *IssueType*
- **order** – order of *IssueType*
- **project** – the taiga project of *IssueType*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.IssueTypes(*requester*)

Bases: *MoveOnDestroyMixinList, ListResource*

IssueTypes factory

instance

alias of *IssueType*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.Issues(*requester*)

Bases: *ListResource*

create(*project, subject, priority, status, issue_type, severity, **attrs*)

Create a new *Task*.

Parameters

- **project** – *Project* id
- **subject** – subject of *Issue*
- **priority** – priority of *Issue*
- **status** – status of *Issue*
- **issue_type** – Issue type of *Issue*
- **severity** – severity of *Issue*
- **attrs** – optional attributes of *Task*

instance

alias of *Issue*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.Membership(*requester, **params*)

Bases: *InstanceResource*

Membership model

Parameters

- **email** – email of *Membership*
- **role** – role of *Membership*
- **project** – project of *Membership*

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**Memberships**(*requester*)

Bases: *ListResource*

Memberships factory class

create(*project, email, role, **attrs*)

Create a new *Membership*.

Parameters

- **project** – *Project* id
- **email** – email of *Membership*
- **role** – role of *Membership*
- **attrs** – optional attributes of *Membership*

instance

alias of *Membership*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**Milestone**(*requester, **params*)

Bases: *InstanceResource*

Milestone model

Parameters

- **name** – the name of *Milestone*
- **project** – the Taiga project of *Milestone*
- **estimated_start** – the estimated start of *Milestone*
- **estimated_finish** – the estimated finish of *Milestone*
- **disponibility** – the disponibility of *Milestone*

delete(*query=None*)

Delete the current *InstanceResource*

classmethod **parse**(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current *InstanceResource*

stats()

Get the stats for the current *Milestone*

to_dict()

Get a dictionary representation of *InstanceResource*

update(***args*)

Update the current *InstanceResource*

class taiga.models.models.**Milestones**(*requester*)

Bases: *ListResource*

Milestones factory

create(*project, name, estimated_start, estimated_finish, **attrs*)

Create a new *Milestone*.

Parameters

- **project** – *Project* id
- **name** – name of *Milestone*
- **estimated_start** – est. start time of *Milestone*
- **estimated_finish** – est. finish time of *Milestone*
- **attrs** – optional attributes of *Milestone*

instance

alias of *Milestone*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.MoveOnDestroyMixinList

Bases: object

Mixin that define a delete method with moveTo parameter

class taiga.models.models.MoveOnDestroyMixinObject

Bases: object

Mixin that define a delete method with moveTo parameter

class taiga.models.models.Point(*requester, **params*)

Bases: *MoveOnDestroyMixinObject, InstanceResource*

Taiga Point model

Parameters

- **color** – the color of *Point*
- **value** – value of *Point*
- **name** – name of *Point*
- **order** – the order of *Point*
- **project** – the Taiga project of *Point*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**Points**(*requester*)

Bases: *MoveOnDestroyMixinList*, *ListResource*

Points factory

create(*project*, *name*, *value*, ***attrs*)

Create a new *UserStoryStatus*.

Parameters

- **project** – *Project* id
- **name** – name of *Point*
- **value** – value of *Point*
- **attrs** – optional attributes of *Point*

instance

alias of *Point*

list(*pagination=True*, *page_size=None*, *page=None*, ***queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod **parse**(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**Priorities**(*requester*)

Bases: *MoveOnDestroyMixinList*, *ListResource*

Priorities factory class

create(*project*, *name*, ****attrs**)

Create a new *Priority*.

Parameters

- **project** – *Project* id
- **name** – email of the priority
- **attrs** – optional attributes of the priority

instance

alias of *Priority*

list(*pagination=True*, *page_size=None*, *page=None*, ****queryparams**)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod **parse**(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**Priority**(*requester*, ****params**)

Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Priority model

Parameters

- **name** – name of *Priority*
- **color** – color of the class:*Priority*
- **order** – order of the class:*Priority*
- **project** – project of the class:*Priority*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**Project**(*requester, **params*)

Bases: [InstanceResource](#)

Taiga project model

Parameters

- **requester** – Requester instance
- **name** – name of the project
- **description** – description of the project (not available in the [Projects.list\(\)](#) response)
- **creation_template** – base template for the project
- **is_backlog_activated** – name of the project
- **is_issues_activated** – name of the project
- **is_kanban_activated** – name of the project
- **is_wiki_activated** – determines if the project is private or not
- **is_private** – determines if the project is private or not
- **videoconferences** – appear-in or talky
- **videoconferences_salt** – for videoconference chat url generation
- **total_milestones** – missing
- **total_story_points** – missing

add_epic(*subject, **attrs*)

Adds a [UserStory](#) and returns a [UserStory](#) resource.

Parameters

- **subject** – subject of [UserStory](#)
- **attrs** – other [UserStory](#) attributes

add_issue(*subject, priority, status, issue_type, severity, **attrs*)

Adds a Issue and returns a [Issue](#) resource.

Parameters

- **subject** – subject of [Issue](#)
- **priority** – priority of [Issue](#)
- **priority** – status of [Issue](#)

- **issue_type** – type of *Issue*
- **severity** – severity of *Issue*
- **attrs** – other *Issue* attributes

add_issue_attribute(*name*, ****attrs**)

Add a new Issue attribute and return a *IssueAttribute* object.

Parameters

- **name** – name of *IssueAttribute*
- **attrs** – optional attributes for *IssueAttribute*

add_issue_status(*name*, ****attrs**)

Add a Issue status to the project and returns a *IssueStatus* object.

Parameters

- **name** – name of *IssueStatus*
- **attrs** – optional attributes for *IssueStatus*

add_issue_type(*name*, ****attrs**)

Add a Issue type to the project and returns a *IssueType* object.

Parameters

- **name** – name of *IssueType*
- **attrs** – optional attributes for *IssueType*

add_membership(*email*, *role*, ****attrs**)

Add a Membership to the project and returns a *Membership* resource.

Parameters

- **email** – email for *Membership*
- **role** – role for *Membership*
- **attrs** – role for *Membership*
- **attrs** – optional *Membership* attributes

add_milestone(*name*, *estimated_start*, *estimated_finish*, ****attrs**)

Add a Milestone to the project and returns a *Milestone* object.

Parameters

- **name** – name of *Milestone*
- **estimated_start** – estimated start time of the *Milestone*
- **estimated_finish** – estimated finish time of the *Milestone*
- **attrs** – optional attributes for *Milestone*

add_point(*name*, *value*, ****attrs**)

Add a Point to the project and returns a *Point* object.

Parameters

- **name** – name of *Point*
- **value** – value of *Point*

- **attrs** – optional attributes for *Point*

add_priority(*name*, ****attrs**)

Add a Priority to the project and returns a *Priority* object.

Parameters

- **name** – name of *Priority*
- **attrs** – optional attributes for *Priority*

add_role(*name*, ****attrs**)

Add a Role to the project and returns a *Role* object.

Parameters

- **name** – name of *Role*
- **attrs** – optional attributes for *Role*

add_severity(*name*, ****attrs**)

Add a Severity to the project and returns a *Severity* object.

Parameters

- **name** – name of *Severity*
- **attrs** – optional attributes for *Severity*

add_swimlane(*name*, ****attrs**)

Adds a *SwimLane* and returns a *SwimLane* resource.

Parameters

- **name** – name of *SwimLane*
- **attrs** – other *SwimLane* attributes

add_tag(*tag*, *color=None*)

Add a new tag and return a response object.

Parameters

- **tag** – name of the tag
- **color** – optional color of the tag

add_task_attribute(*name*, ****attrs**)

Add a new Task attribute and return a *TaskAttribute* object.

Parameters

- **name** – name of *TaskAttribute*
- **attrs** – optional attributes for *TaskAttribute*

add_task_status(*name*, ****attrs**)

Add a Task status to the project and returns a *TaskStatus* object.

Parameters

- **name** – name of *TaskStatus*
- **attrs** – optional attributes for *TaskStatus*

add_user_story(*subject*, ***attrs*)

Adds a *UserStory* and returns a *UserStory* resource.

Parameters

- **subject** – subject of *UserStory*
- **attrs** – other *UserStory* attributes

add_user_story_attribute(*name*, ***attrs*)

Add a new User Story attribute and return a *UserStoryAttribute* object.

Parameters

- **name** – name of *UserStoryAttribute*
- **attrs** – optional attributes for *UserStoryAttribute*

add_user_story_status(*name*, ***attrs*)

Add a UserStory status to the project and returns a *UserStoryStatus* object.

Parameters

- **name** – name of *UserStoryStatus*
- **attrs** – optional attributes for *UserStoryStatus*

add_webhook(*name*, *url*, *key*, ***attrs*)

Add a new Webhook and return a *Webhook* object.

Parameters

- **name** – name of *Webhook*
- **url** – payload url of *Webhook*
- **key** – secret key of *Webhook*
- **attrs** – optional attributes for *Webhook*

add_wikilink(*title*, *href*, ***attrs*)

Add a Wiki link to the project and returns a *WikiLink* object.

Parameters

- **title** – title of *WikiLink*
- **href** – href of *WikiLink*
- **attrs** – optional attributes for *WikiLink*

add_wikipage(*slug*, *content*, ***attrs*)

Add a Wiki page to the project and returns a *WikiPage* object.

Parameters

- **name** – name of *WikiPage*
- **attrs** – optional attributes for *WikiPage*

delete(*query=None*)

Delete the current InstanceResource

duplicate(*name, description, is_private=False, users=[], **attrs*)

Duplicate a *Project*

Parameters

- **name** – name of new *Project*
- **description** – description of new *Project*
- **is_private** – determines if the project is private or not
- **users** – users of the new *Project*
- **attrs** – optional attributes for the new *Project*

get_epic_by_ref(*ref*)

Get a *Epic* by ref.

Parameters

ref – *Epic* reference

get_issue_by_ref(*ref*)

Get a *Issue* by ref.

Parameters

ref – *Issue* reference

get_task_by_ref(*ref*)

Get a *Task* by ref.

Parameters

ref – *Task* reference

get_userstory_by_ref(*ref*)

Get a *UserStory* by ref.

Parameters

ref – *UserStory* reference

import_issue(*subject, priority, status, issue_type, severity, **attrs*)

Import and issue and returns a *Issue* resource.

Parameters

- **subject** – subject of *Issue*
- **priority** – priority of *Issue*
- **status** – status of *Issue*
- **issue_type** – issue type of *Issue*
- **severity** – severity of *Issue*
- **attrs** – optional *Issue* attributes

import_milestone(*name, estimated_start, estimated_finish, **attrs*)

Import a Milestone and returns a *Milestone* object.

Parameters

- **name** – name of *Milestone*
- **estimated_start** – estimated start time of the *Milestone*
- **estimated_finish** – estimated finish time of the *Milestone*

- **attrs** – optional attributes for *Milestone*

import_task(*subject*, *status*, ****attrs**)

Import a Task and return a *Task* object.

Parameters

- **subject** – subject of *Task*
- **status** – status of *Task*
- **attrs** – optional attributes for *Task*

import_user_story(*subject*, *status*, ****attrs**)

Import an user story and returns a *UserStory* resource.

Parameters

- **subject** – subject of *UserStory*
- **status** – status of *UserStory*
- **attrs** – optional *UserStory* attributes

import_wikilink(*title*, *href*, ****attrs**)

Import a Wiki link and return a *WikiLink* object.

Parameters

- **title** – title of *WikiLink*
- **href** – href of *WikiLink*
- **attrs** – optional attributes for *WikiLink*

import_wikiplane(*slug*, *content*, ****attrs**)

Import a Wiki page and return a *WikiPage* object.

Parameters

- **slug** – slug of *WikiPage*
- **content** – content of *WikiPage*
- **attrs** – optional attributes for *Task*

issues_stats()

Get stats for issues of the project

like()

Like the project

list_epic_attributes()

Get the list of *EpicAttribute* resources for the project.

list_epics()

Get the list of *Epic* resources for the project.

list_issue_attributes()

Get the list of *IssueAttribute* resources for the project.

list_issue_statuses()

Get the list of *IssueStatus* resources for the project.

list_issue_types()

Get the list of *IssueType* resources for the project.

list_issues()

Returns the *Issue* list of the project.

list_memberships()

Get the list of *Membership* resources for the project.

list_milestones(queryparams)**

Get the list of *Milestone* resources for the project.

list_points()

Get the list of *Point* resources for the project.

list_priorities()

Get the list of *Priority* resources for the project.

list_roles()

Get the list of *Role* resources for the project.

list_severities()

Get the list of *Severity* resources for the project.

list_swimlanes(queryparams)**

Returns the *SwimLane* list of the project.

list_tags()

Get the list of tags for the project.

list_task_attributes()

Get the list of *TaskAttribute* resources for the project.

list_task_statuses()

Get the list of *Task* resources for the project.

list_user_stories(queryparams)**

Returns the *UserStory* list of the project.

list_user_story_attributes()

Get the list of *UserStoryAttribute* resources for the project.

list_user_story_statuses()

Get the list of *UserStoryStatus* resources for the project.

list_webhooks()

Get the list of *Webhook* resources for the project.

list_wikilinks()

Get the list of *WikiLink* resources for the project.

list_wikipages()

Get the list of *WikiPage* resources for the project.

classmethod parse(requester, entry)

Turns a JSON object into a model instance.

patch(fields, **args)

Patch the current InstanceResource

star()

Stars the project

Deprecated since version 0.8.5: Update Taiga and use like instead

stats()

Get the stats of the project

to_dict()

Get a dictionary representation of InstanceResource

unlike()

Unlike the project

unstar()

Unstars the project

Deprecated since version 0.8.5: Update Taiga and use unlike instead

update(args)**

Update the current InstanceResource

class taiga.models.models.**Projects**(requester)

Bases: *ListResource*

Projects factory

create(name, description, **attrs)

Create a new *Project*

Parameters

- **name** – name of *Project*
- **description** – description of *Project*
- **attrs** – optional attributes for *Project*

get_by_slug(slug)

Get a *Project* by slug

Parameters

slug – the slug of *Project*

instance

alias of *Project*

list(pagination=True, page_size=None, page=None, **queryparams)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value

- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(entries)

Parse a JSON array into a list of model instances.

class `taiga.models.models.Role(requester, **params)`Bases: *InstanceResource*

Role model

Parameters

- **requester** – Requester instance
- **name** – name of *Role*
- **slug** – slug of *Role*
- **order** – order of *Role*
- **computable** – choose if *Role* is computable or not

delete(query=None)Delete the current *InstanceResource***classmethod** `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(fields, **args)Patch the current *InstanceResource***to_dict()**Get a dictionary representation of *InstanceResource***update(**args)**Update the current *InstanceResource***class** `taiga.models.models.Roles(requester)`Bases: *ListResource*

Roles factory

create(project, name, **attrs)Create a new *Role***Parameters**

- **project** – *Project* id
- **name** – name of *Role*
- **attrs** – optional attributes for *Role*

instancealias of *Role*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.Severities(*requester*)

Bases: *MoveOnDestroyMixinList, ListResource*

Severities factory

create(*project, name, **attrs*)

Create a new *Severity*

Parameters

- **project** – *Project* id
- **name** – name of *Severity*
- **attrs** – optional attributes for *Role*

instance

alias of *Severity*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)

- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class `taiga.models.models.Severity(requester, **params)`Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Severity model

Parameters

- **requester** – Requester instance
- **name** – name of *Severity*
- **order** – order of *Severity*
- **project** – *Project* id

delete(*move_to_id*)Delete the current *InstanceResource***classmethod** `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields*, ***args*)Patch the current *InstanceResource***to_dict**()Get a dictionary representation of *InstanceResource***update**(***args*)Update the current *InstanceResource***class** `taiga.models.models.SwimLane(requester, **params)`Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Taiga Swimlane model

Parameters

- **name** – The name of *SwimLane*
- **order** – the order of *SwimLane*
- **project** – the project of *SwimLane*
- **statuses** – the statuses of *SwimLane*

delete(*move_to_id*)Delete the current *InstanceResource*

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.SwimLanes(requester)`

Bases: [MoveOnDestroyMixinList](#), [ListResource](#)

create(*project*, *name*, ***attrs*)

Create a new [SwimLane](#).

Parameters

- **project** – [Project](#) id
- **name** – name of [SwimLane](#)
- **attrs** – optional attributes of [SwimLane](#)

instance

alias of [SwimLane](#)

list(*pagination=True*, *page_size=None*, *page=None*, ***queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**Task**(*requester*, ***params*)

Bases: *CustomAttributeResource*, *CommentableResource*

Task model

Parameters

- **assigned_to** – assigned to property of *TaskStatus*
- **blocked_note** – blocked note of *TaskStatus*
- **description** – description of of *TaskStatus* (not available in the *Tasks.list()* response)
- **version** – version of *TaskStatus*
- **is_blocked** – is blocked property of *TaskStatus*
- **milestone** – milestone property of *TaskStatus*
- **project** – the project of *TaskStatus*
- **user_story** – the user story of *TaskStatus*
- **status** – status of *TaskStatus*
- **subject** – subject of *TaskStatus*
- **tags** – tags of *TaskStatus*
- **us_order** – the use order of *TaskStatus*
- **taskboard_order** – the taskboard order of *TaskStatus*
- **is_iocaine** – the is iocaine of *TaskStatus*
- **external_reference** – external reference of *TaskStatus*
- **watchers** – watchers of *TaskStatus*
- **due_date** – *Task* due date

add_comment(*comment*)

Add a comment to the current element

Parameters

comment – the comment you want to insert

attach(*attached_file*, ***attrs*)

Attach a file to the *Task*

Parameters

- **attached_file** – file path to attach
- **attrs** – optional attributes for the attached file

delete(*query=None*)

Delete the current InstanceResource

get_attributes()

Get all the attributes of the current object

list_attachments()

Get a list of *TaskAttachment*.

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

set_attribute(*id, value, version=1*)

Set attribute to a specific value

Parameters

- **id** – id of the attribute
- **value** – value of the attribute
- **version** – version of the attribute (default = 1)

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.TaskAttachment(requester, **params)`

Bases: [Attachment](#)

TaskAttachment model

delete(*query=None*)

Delete the current InstanceResource

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.TaskAttachments(requester)`

Bases: [Attachments](#)

TaskAttachments factory

create(*project, object_id, attached_file, **attrs*)

Create a new [Attachment](#).

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the [Attachment](#)

instance

alias of *TaskAttachment*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.TaskAttribute(*requester, **params*)

Bases: *CustomAttribute*

TaskAttribute model

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.TaskAttributes(*requester*)

Bases: *CustomAttributes*

TaskAttributes factory

create(*project, name, **attrs*)

Create a new *CustomAttribute*.

Parameters

- **project** – *Project* id
- **name** – name of the custom attribute
- **attrs** – optional attributes of the custom attributes

instance

alias of *TaskAttribute*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.TaskStatus(*requester, **params*)

Bases: *MoveOnDestroyMixinObject, InstanceResource*

Task Status model

Parameters

- **name** – the name of *TaskStatus*
- **color** – the color of *TaskStatus*
- **order** – the order of *TaskStatus*
- **project** – the project of *TaskStatus*
- **is_closed** – the is closed property of *TaskStatus*

delete(*move_to_id*)

Delete the current *InstanceResource*

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.TaskStatuses(requester)`

Bases: [MoveOnDestroyMixinList](#), [ListResource](#)

create(*project*, *name*, ***attrs*)

Create a new [TaskStatus](#).

Parameters

- **project** – [Project](#) id
- **name** – name of [TaskStatus](#)
- **attrs** – optional attributes of [TaskStatus](#)

instance

alias of [TaskStatus](#)

list(*pagination=True*, *page_size=None*, *page=None*, ***queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.Tasks(*requester*)

Bases: *ListResource*

Tasks factory

create(*project*, *subject*, *status*, ****attrs**)

Create a new *Task*.

Parameters

- **project** – *Project* id
- **subject** – subject of *Task*
- **status** – status of *Task*
- **attrs** – optional attributes of *Task*

instance

alias of *Task*

list(*pagination=True*, *page_size=None*, *page=None*, ****queryparams**)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.User(*requester*, ****params**)

Bases: *InstanceResource*

User model

delete(*query=None*)

Delete the current *InstanceResource*

classmethod parse(*requester*, *entry*)

Turns a JSON object into a model instance.

patch(*fields*, ***args*)

Patch the current InstanceResource

starred_projects()

Get a list of starred *Project*.

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**UserStories**(*requester*)

Bases: *ListResource*

UserStories factory class

create(*project*, *subject*, ***attrs*)

Create a new *UserStory*.

Parameters

- **project** – *Project* id
- **subject** – subject of *UserStory*
- **attrs** – optional attributes of *UserStory*

instance

alias of *UserStory*

list(*pagination=True*, *page_size=None*, *page=None*, ***queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**UserStory**(*requester*, ***params*)

Bases: *CustomAttributeResource*, *CommentableResource*

User Story model

Parameters

- **assigned_to** – assigned to of *UserStory*
- **assigned_users** – additional users assigned to of *UserStory*
- **backlog_order** – backlog order of *UserStory*
- **blocked_note** – blocked note of *UserStory*
- **version** – version of *UserStory*
- **client_requirement** – client requirement of *UserStory*
- **description** – description of *UserStory* (not available in the *UserStories.list()* response)
- **is_blocked** – is blocked of *UserStory*
- **kanban_order** – kanban order of *UserStory*
- **milestone** – milestone of *UserStory*
- **points** – points of *UserStory*
- **project** – project of *UserStory*
- **sprint_order** – sprint order of *UserStory*
- **status** – status of *UserStory*
- **subject** – subject of *UserStory*
- **tags** – tags of *UserStory*
- **team_requirement** – team requirement of *UserStory*
- **watchers** – watchers of *UserStory*
- **due_date** – *UserStory* due date
- **generated_from_issue** – *UserStory* parent issue
- **generated_from_task** – *UserStory* parent task

add_comment(*comment*)

Add a comment to the current element

Parameters

comment – the comment you want to insert

add_task(*subject*, *status*, ***attrs*)

Add a *Task* to the current *UserStory* and return it. :param subject: subject of *Task* :param status: status of *Task* :param attrs: optional attributes for *Task*

attach(*attached_file*, ***attrs*)

Attach a file to the *UserStory*

Parameters

- **attached_file** – file path to attach
- **attrs** – optional attributes for the attached file

delete(*query=None*)

Delete the current InstanceResource

get_attributes()

Get all the attributes of the current object

list_attachments()

Get a list of *UserStoryAttachment*.

list_tasks()

Get a list of *Task* in the current *UserStory*.

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

set_attribute(*id, value, version=1*)

Set attribute to a specific value

Parameters

- **id** – id of the attribute
- **value** – value of the attribute
- **version** – version of the attribute (default = 1)

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**UserStoryAttachment**(*requester, **params*)

Bases: *Attachment*

UserStoryAttachment class

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**UserStoryAttachments**(*requester*)

Bases: *Attachments*

UserStoryAttachments factory class

create(*project, object_id, attached_file, **attrs*)

Create a new *Attachment*.

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the *Attachment*

instance

alias of *UserStoryAttachment*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.UserStoryAttribute(*requester, **params*)

Bases: *CustomAttribute*

UserStoryAttribute model

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(args)**

Update the current InstanceResource

class taiga.models.models.**UserStoryAttributes**(*requester*)

Bases: *CustomAttributes*

UserStoryAttributes factory

create(*project*, *name*, ***attrs*)

Create a new *CustomAttribute*.

Parameters

- **project** – *Project* id
- **name** – name of the custom attribute
- **attrs** – optional attributes of the custom attributes

instance

alias of *UserStoryAttribute*

list(*pagination=True*, *page_size=None*, *page=None*, ***queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester*, *entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**UserStoryStatus**(*requester*, ***params*)

Bases: *MoveOnDestroyMixinObject*, *InstanceResource*

Taiga User Story Status model

Parameters

- **color** – the color of *UserStoryStatus*

- **is_closed** – closed property of *UserStoryStatus*
- **name** – The name of *UserStoryStatus*
- **order** – order of *UserStoryStatus*
- **project** – the Taiga project of *UserStoryStatus*
- **wip_limit** – wip limit of *UserStoryStatus*

delete(*move_to_id*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**UserStoryStatuses**(*requester*)

Bases: *MoveOnDestroyMixinList, ListResource*

create(*project, name, **attrs*)

Create a new *UserStoryStatus*.

Parameters

- **project** – *Project* id
- **name** – name of *UserStoryStatus*
- **attrs** – optional attributes of *UserStoryStatus*

instance

alias of *UserStoryStatus*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(entries)

Parse a JSON array into a list of model instances.

class `taiga.models.models.Users(requester)`Bases: [ListResource](#)

Users factory class

instancealias of [User](#)**list**(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(entries)

Parse a JSON array into a list of model instances.

class `taiga.models.models.Webhook(requester, **params)`Bases: [InstanceResource](#)

Webhook model

Parameters

- **requester** – Requester instance
- **name** – name of [Webhook](#)
- **url** – payload url of [Webhook](#)
- **key** – secret key of [Webhook](#)

delete(*query=None*)

Delete the current InstanceResource

classmethod parse(*requester, entry*)

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class taiga.models.models.**Webhooks**(*requester*)

Bases: *ListResource*

Webhooks factory

create(*project, name, url, key, **attrs*)

Create a new *Webhook*

Parameters

- **project** – *Project* id
- **name** – name of *Webhook*
- **url** – payload url of *Webhook*
- **key** – secret key of *Webhook*
- **attrs** – optional attributes for *Webhook*

instance

alias of *Webhook*

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class `taiga.models.models.WikiAttachment(requester, **params)`

Bases: [Attachment](#)

WikiAttachment model

delete(*query=None*)

Delete the current InstanceResource

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(***args*)

Update the current InstanceResource

class `taiga.models.models.WikiAttachments(requester)`

Bases: [Attachments](#)

WikiAttachments factory

create(*project, object_id, attached_file, **attrs*)

Create a new [Attachment](#).

Parameters

- **project** – *Project* id
- **object_id** – id of the current object
- **ref** – *Task* reference
- **attached_file** – file path that you want to upload
- **attrs** – optional attributes for the [Attachment](#)

instance

alias of [WikiAttachment](#)

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)

- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(entries)

Parse a JSON array into a list of model instances.

class `taiga.models.models.WikiLink(requester, **params)`

Bases: *InstanceResource*

WikiLink model

Parameters

- **project** – *Project* id
- **title** – title of the wiki link
- **href** – href for the wiki link
- **order** – order of the wiki link

delete(query=None)

Delete the current *InstanceResource*

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(fields, **args)

Patch the current *InstanceResource*

to_dict()

Get a dictionary representation of *InstanceResource*

update(args)**

Update the current *InstanceResource*

class `taiga.models.models.WikiLinks(requester)`

Bases: *ListResource*

WikiLinks factory

create(project, title, href, **attrs)

Create a new *WikiLink*

Parameters

- **project** – *Project* id
- **title** – title of the wiki link
- **href** – href for the wiki link
- **attrs** – optional attributes for the *WikiLink*

instance

alias of [WikiLink](#)

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(*requester, entries*)

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.models.**WikiPage**(*requester, **params*)

Bases: [InstanceResource](#)

WikiPage model

Parameters

- **project** – [Project](#) id
- **slug** – slug of the wiki page
- **content** – content of the wiki page
- **watchers** – list of watchers id

attach(*attached_file, **attrs*)

Attach a file to the [WikiPage](#)

Parameters

- **attached_file** – file path to attach
- **attrs** – optional attributes for the attached file

delete(*query=None*)

Delete the current InstanceResource

list_attachments()

Get a list of [WikiAttachment](#).

classmethod `parse(requester, entry)`

Turns a JSON object into a model instance.

patch(*fields, **args*)

Patch the current InstanceResource

to_dict()

Get a dictionary representation of InstanceResource

update(**args)

Update the current InstanceResource

class `taiga.models.models.WikiPages(requester)`

Bases: [ListResource](#)

WikiPages factory

create(*project, slug, content, **attrs*)

create a new [WikiPage](#)

Parameters

- **project** – [Project](#) id
- **slug** – slug of the wiki page
- **content** – content of the wiki page
- **attrs** – optional attributes for the [WikiPage](#)

instance

alias of [WikiPage](#)

list(*pagination=True, page_size=None, page=None, **queryparams*)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod `parse(requester, entries)`

Parse a JSON array into a list of model instances.

parse_list(*entries*)

Parse a JSON array into a list of model instances.

class taiga.models.base.InstanceResource(requester, **params)

InstanceResource model

Base class for methods that operates on a single resource (*update()*, *patch()*, *delete()*).

Parameters

- **requester** – Requester instance
- **params** – :various parameters

delete(query=None)

Delete the current *InstanceResource*

classmethod parse(requester, entry)

Turns a JSON object into a model instance.

patch(fields, **args)

Patch the current *InstanceResource*

to_dict()

Get a dictionary representation of *InstanceResource*

update(**args)

Update the current *InstanceResource*

class taiga.models.base.ListResource(requester)

ListResource model

Base class for methods that operates on a list of resources (*list()*, *get()*, *delete()*).

Parameters

requester – Requester instance

list(pagination=True, page_size=None, page=None, **queryparams)

Retrieves a list of objects.

By default uses local cache and remote pagination

If pagination is used and no page is requested (the default), all the remote objects are retrieved and appended in a single list.

If pagination is disabled, all the objects are fetched from the endpoint and returned. This may trigger some parsing error if the result set is very large.

Parameters

- **pagination** – Use pagination (default: *True*)
- **page_size** – Size of the pagination page (default: *100*). Any non numeric value will be casted to the default value
- **page** – Page number to retrieve (default: *None*). Ignored if *pagination* is *False*
- **queryparams** – Additional filter parameters as accepted by the remote API

Returns

<SearchableList>

classmethod parse(requester, entries)

Parse a JSON array into a list of model instances.

`parse_list(entries)`

Parse a JSON array into a list of model instances.

`class taiga.models.base.SearchableList(iterable=(), /)`

1.5 Development & community

python-taiga is an open-source project.

You don't need to be an expert developer to make a valuable contribution - all you need is a little knowledge, and a willingness to follow the contribution guidelines.

Refer to [Nephila contribution guidelines](#) for the general contribution guidelines and code of conduct.

1.6 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

Refer to [Nephila contribution guidelines](#) for the general contribution guidelines and code of conduct.

You can contribute in many ways:

1.6.1 Types of Contributions

Report Bugs

Report bugs at <https://github.com/nephila/python-taiga/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

Write Documentation

python-taiga could always use more documentation, whether as part of the official python-taiga docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/nephila/python-taiga/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Get Started!

Ready to contribute? Here's how to set up `python-taiga` for local development.

1. Fork the `python-taiga` repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/python-taiga.git
```

3. Install your local copy into a virtualenv. Assuming you have `virtualenvwrapper` installed, this is how you set up your fork for local development:

```
$ mkvirtualenv taiga
$ cd taiga/
$ pip install -e .
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass `flake8` and the tests, including testing other Python versions with `tox`:

```
$ tox
```

To get `tox`, just `pip install` them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

1.6.2 Development tips

This project allows you to use `pre-commit` to ensure an easy compliance to the project code styles.

If you want to use it, install it globally (for example with `pip3 install --user precommit`, but check [installation instruction](#)). When first cloning the project ensure you install the git hooks by running `pre-commit install`.

From now on every commit will be checked against our code style.

Check also the available `tox` environments with `tox -l`: the ones not marked with a python version number are tools to help you work on the project buy checking / formatting code style, running docs etc.

1.6.3 Testing tips

You can test your project using any specific version of python.

For example `tox -epy37` runs the tests on python 3.7.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. Pull request must be named with the following naming scheme:
`<type>/(<optional-task-type>-)<number>-description`
See below for available types.
2. The pull request should include tests.
3. If the pull request adds functionality, the docs should be updated. Documentation must be added in `README.rst` file, and must include usage information for the end user. In case of public API method, add extended docstrings with full parameters description and usage example.
4. Add a changes file in `changes` directory describing the contribution in one line. It will be added automatically to the history file upon release. File must be named as `<issue-number>.<type>` with type being:
 - `.feature`: For new features.
 - `.bugfix`: For bug fixes.
 - `.doc`: For documentation improvement.
 - `.removal`: For deprecation or removal of public API.
 - `.misc`: For general issues.Check [towncrier](#) documentation for more details.
5. The pull request should work for all python versions declared in `tox.ini`. Check the CI and make sure that the tests pass for all supported versions.

Release a version

1. Update authors file
2. Merge `develop` on `master` branch
3. Bump release via task: `inv tag-release --level=(major|minor|patch)`
4. Update changelog via towncrier: `towncrier --yes`
5. Commit changelog with `git commit --amend` to merge with `bump-my-version` commit
6. Create tag `git tag <version>`
7. Push tag to github
8. Publish the release from the tags page
9. If pipeline succeeds, push `master`
10. Merge `master` back on `develop`
11. Bump development version via task: `inv tag-dev --level=release`
12. Push `develop`

To increment dev version use `inv tag-dev --level=relver` (e.g. to pass from 1.2.0.dev1 to 1.2.0.dev2)`

1.7 History

1.7.1 1.3.3 (2025-08-25)

Features

- Add proxies parameter to taiga client (#55)

1.7.2 1.3.2 (2025-01-22)

Bugfixes

- Add missing due_date to Issue allowed_params (#196)

1.7.3 1.3.1 (2025-01-13)

Bugfixes

- Fix parser when entries is None (#169)
- Add missing swimlane to user story allowed_params (#185)
- Use pytest for tests (#190)

1.7.4 1.3.0 (2024-01-02)

Features

- Switch to bump-my-version (#140)
- Switch to Coveralls Github action (#155)
- Add a duplicate() method to Project. (#161)
- Adds SwimLane/SwimLanes models and support to add/list in Project. (#162)
- Adds the ability to read/write the default_swimlane attribute in Project. (#166)

Bugfixes

- Add the version parameter to the alloed parameter so the requester can access it. (#149)

1.7.5 1.2.0 (2023-08-23)

Features

- Add list_attachments on WikiPage (#100)
- Add refresh_token API call (#131)

Bugfixes

- Add count to SearchResult object, fix wikipages attribute name in SearchResult object (#111)
- Add moveTo parameter in delete methods when needed by Taiga API (#130)
- Fix ruff linting error (#134)

1.7.6 1.1.0 (2023-04-23)

Features

- Update tooling, drop Python 2 (#59)
- Implement list_epic_attributes() (#103)
- Update packaging - python versions (#124)

Bugfixes

- Update HistoryItem methods signatures (#97)
- Improve models documentation (#105)
- Add refresh token support to tests/resources/auth_users_success.json (#114)
- Fix pagination (#116)
- Update linting tools and fix code style (#117)

Improved Documentation

- Improve documentation (#58)

1.7.7 1.0.0 (2019-08-08)

- Add support for python 3.7
- Add support for epics
- Add support for additional attributes from Taiga 3.2/3.3
- Improve models
- Improve pagination support
- Fix and enforce code style

1.7.8 0.9.0 (2018-01-31)

- Add support for multiple authentication backends
- Add support for self-signed and not verified SSL certificates
- Add support for webhooks
- Add pagination support for lists
- Add debian packaging
- Avoid installing tests as a separate module
- Move documentation to readthedocs
- Add support for Python 3.5/3.6

1.7.9 0.8.6 (2016-08-26)

- Fix header values to be strings instead of boolean
- Fix requests compatibility issues

1.8 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

PYTHON MODULE INDEX

t

`taiga.client`, 5

`taiga.models.base`, 60

`taiga.models.models`, 6

A

add_comment() (*taiga.models.models.CommentableResource* method), 7
 add_comment() (*taiga.models.models.Epic* method), 10
 add_comment() (*taiga.models.models.Issue* method), 19
 add_comment() (*taiga.models.models.Task* method), 43
 add_comment() (*taiga.models.models.UserStory* method), 50
 add_epic() (*taiga.models.models.Project* method), 31
 add_issue() (*taiga.models.models.Project* method), 31
 add_issue_attribute() (*taiga.models.models.Project* method), 32
 add_issue_status() (*taiga.models.models.Project* method), 32
 add_issue_type() (*taiga.models.models.Project* method), 32
 add_membership() (*taiga.models.models.Project* method), 32
 add_milestone() (*taiga.models.models.Project* method), 32
 add_point() (*taiga.models.models.Project* method), 32
 add_priority() (*taiga.models.models.Project* method), 33
 add_role() (*taiga.models.models.Project* method), 33
 add_severity() (*taiga.models.models.Project* method), 33
 add_swimlane() (*taiga.models.models.Project* method), 33
 add_tag() (*taiga.models.models.Project* method), 33
 add_task() (*taiga.models.models.UserStory* method), 50
 add_task_attribute() (*taiga.models.models.Project* method), 33
 add_task_status() (*taiga.models.models.Project* method), 33
 add_user_story() (*taiga.models.models.Project* method), 33
 add_user_story_attribute() (*taiga.models.models.Project* method), 34
 add_user_story_status() (*taiga.models.models.Project* method), 34
 add_webhook() (*taiga.models.models.Project* method),

34
 add_wikilink() (*taiga.models.models.Project* method), 34
 add_wikipage() (*taiga.models.models.Project* method), 34
 attach() (*taiga.models.models.Epic* method), 10
 attach() (*taiga.models.models.Issue* method), 19
 attach() (*taiga.models.models.Task* method), 43
 attach() (*taiga.models.models.UserStory* method), 50
 attach() (*taiga.models.models.WikiPage* method), 59
 Attachment (class in *taiga.models.models*), 6
 Attachments (class in *taiga.models.models*), 7
 auth() (*taiga.client.TaigaAPI* method), 5
 auth_app() (*taiga.client.TaigaAPI* method), 6

C

CommentableResource (class in *taiga.models.models*), 7
 create() (*taiga.models.models.Attachments* method), 7
 create() (*taiga.models.models.CustomAttributes* method), 9
 create() (*taiga.models.models.EpicAttachments* method), 11
 create() (*taiga.models.models.EpicAttributes* method), 12
 create() (*taiga.models.models.Epics* method), 15
 create() (*taiga.models.models.EpicStatuses* method), 14
 create() (*taiga.models.models.IssueAttachments* method), 20
 create() (*taiga.models.models.IssueAttributes* method), 22
 create() (*taiga.models.models.Issues* method), 24
 create() (*taiga.models.models.Memberships* method), 26
 create() (*taiga.models.models.Milestones* method), 27
 create() (*taiga.models.models.Points* method), 29
 create() (*taiga.models.models.Priorities* method), 30
 create() (*taiga.models.models.Projects* method), 38
 create() (*taiga.models.models.Roles* method), 39
 create() (*taiga.models.models.Severities* method), 40
 create() (*taiga.models.models.SwimLanes* method), 42

- create() (*taiga.models.models.TaskAttachments method*), 44
 create() (*taiga.models.models.TaskAttributes method*), 45
 create() (*taiga.models.models.Tasks method*), 48
 create() (*taiga.models.models.TaskStatuses method*), 47
 create() (*taiga.models.models.UserStories method*), 49
 create() (*taiga.models.models.UserStoryAttachments method*), 51
 create() (*taiga.models.models.UserStoryAttributes method*), 53
 create() (*taiga.models.models.UserStoryStatuses method*), 54
 create() (*taiga.models.models.Webhooks method*), 56
 create() (*taiga.models.models.WikiAttachments method*), 57
 create() (*taiga.models.models.WikiLinks method*), 58
 create() (*taiga.models.models.WikiPages method*), 60
 CustomAttribute (*class in taiga.models.models*), 8
 CustomAttributeResource (*class in taiga.models.models*), 8
 CustomAttributes (*class in taiga.models.models*), 9
- ## D
- delete() (*taiga.models.base.InstanceResource method*), 61
 delete() (*taiga.models.models.Attachment method*), 6
 delete() (*taiga.models.models.CommentableResource method*), 8
 delete() (*taiga.models.models.CustomAttribute method*), 8
 delete() (*taiga.models.models.CustomAttributeResource method*), 8
 delete() (*taiga.models.models.Epic method*), 10
 delete() (*taiga.models.models.EpicAttachment method*), 11
 delete() (*taiga.models.models.EpicAttribute method*), 12
 delete() (*taiga.models.models.EpicStatus method*), 13
 delete() (*taiga.models.models.History method*), 15
 delete() (*taiga.models.models.Issue method*), 19
 delete() (*taiga.models.models.IssueAttachment method*), 20
 delete() (*taiga.models.models.IssueAttribute method*), 21
 delete() (*taiga.models.models.IssueStatus method*), 22
 delete() (*taiga.models.models.IssueType method*), 24
 delete() (*taiga.models.models.Membership method*), 25
 delete() (*taiga.models.models.Milestone method*), 27
 delete() (*taiga.models.models.Point method*), 28
 delete() (*taiga.models.models.Priority method*), 30
 delete() (*taiga.models.models.Project method*), 34
 delete() (*taiga.models.models.Role method*), 39
 delete() (*taiga.models.models.Severity method*), 41
 delete() (*taiga.models.models.SwimLane method*), 41
 delete() (*taiga.models.models.Task method*), 43
 delete() (*taiga.models.models.TaskAttachment method*), 44
 delete() (*taiga.models.models.TaskAttribute method*), 45
 delete() (*taiga.models.models.TaskStatus method*), 46
 delete() (*taiga.models.models.User method*), 48
 delete() (*taiga.models.models.UserStory method*), 50
 delete() (*taiga.models.models.UserStoryAttachment method*), 51
 delete() (*taiga.models.models.UserStoryAttribute method*), 52
 delete() (*taiga.models.models.UserStoryStatus method*), 54
 delete() (*taiga.models.models.Webhook method*), 55
 delete() (*taiga.models.models.WikiAttachment method*), 57
 delete() (*taiga.models.models.WikiLink method*), 58
 delete() (*taiga.models.models.WikiPage method*), 59
 delete_comment() (*taiga.models.models.HistoryEntity method*), 16
 delete_comment() (*taiga.models.models.HistoryEpic method*), 16
 delete_comment() (*taiga.models.models.HistoryIssue method*), 17
 delete_comment() (*taiga.models.models.HistoryTask method*), 17
 delete_comment() (*taiga.models.models.HistoryUserStory method*), 18
 delete_comment() (*taiga.models.models.HistoryWiki method*), 18
 downvote() (*taiga.models.models.Issue method*), 19
 duplicate() (*taiga.models.models.Project method*), 34
- ## E
- Epic (*class in taiga.models.models*), 10
 EpicAttachment (*class in taiga.models.models*), 11
 EpicAttachments (*class in taiga.models.models*), 11
 EpicAttribute (*class in taiga.models.models*), 12
 EpicAttributes (*class in taiga.models.models*), 12
 Epics (*class in taiga.models.models*), 14
 EpicStatus (*class in taiga.models.models*), 13
 EpicStatuses (*class in taiga.models.models*), 14
- ## G
- get() (*taiga.models.models.HistoryEntity method*), 16
 get() (*taiga.models.models.HistoryEpic method*), 16
 get() (*taiga.models.models.HistoryIssue method*), 17
 get() (*taiga.models.models.HistoryTask method*), 17
 get() (*taiga.models.models.HistoryUserStory method*), 18

[get\(\)](#) (*taiga.models.models.HistoryWiki method*), 18
[get_attributes\(\)](#) (*taiga.models.models.CustomAttributeResource method*), 8
[get_attributes\(\)](#) (*taiga.models.models.Epic method*), 10
[get_attributes\(\)](#) (*taiga.models.models.Issue method*), 19
[get_attributes\(\)](#) (*taiga.models.models.Task method*), 43
[get_attributes\(\)](#) (*taiga.models.models.UserStory method*), 51
[get_by_slug\(\)](#) (*taiga.models.models.Projects method*), 38
[get_epic_by_ref\(\)](#) (*taiga.models.models.Project method*), 35
[get_issue_by_ref\(\)](#) (*taiga.models.models.Project method*), 35
[get_task_by_ref\(\)](#) (*taiga.models.models.Project method*), 35
[get_userstory_by_ref\(\)](#) (*taiga.models.models.Project method*), 35

H

[History](#) (*class in taiga.models.models*), 15
[HistoryEntity](#) (*class in taiga.models.models*), 16
[HistoryEpic](#) (*class in taiga.models.models*), 16
[HistoryIssue](#) (*class in taiga.models.models*), 17
[HistoryTask](#) (*class in taiga.models.models*), 17
[HistoryUserStory](#) (*class in taiga.models.models*), 18
[HistoryWiki](#) (*class in taiga.models.models*), 18

I

[import_issue\(\)](#) (*taiga.models.models.Project method*), 35
[import_milestone\(\)](#) (*taiga.models.models.Project method*), 35
[import_task\(\)](#) (*taiga.models.models.Project method*), 36
[import_user_story\(\)](#) (*taiga.models.models.Project method*), 36
[import_wikilink\(\)](#) (*taiga.models.models.Project method*), 36
[import_wikipedia\(\)](#) (*taiga.models.models.Project method*), 36
[instance](#) (*taiga.models.models.EpicAttachments attribute*), 11
[instance](#) (*taiga.models.models.EpicAttributes attribute*), 13
[instance](#) (*taiga.models.models.Epics attribute*), 15
[instance](#) (*taiga.models.models.EpicStatuses attribute*), 14
[instance](#) (*taiga.models.models.IssueAttachments attribute*), 21
[instance](#) (*taiga.models.models.IssueAttributes attribute*), 22
[instance](#) (*taiga.models.models.Issues attribute*), 25
[instance](#) (*taiga.models.models.IssueStatuses attribute*), 23
[instance](#) (*taiga.models.models.IssueTypes attribute*), 24
[instance](#) (*taiga.models.models.Memberships attribute*), 26
[instance](#) (*taiga.models.models.Milestones attribute*), 27
[instance](#) (*taiga.models.models.Points attribute*), 29
[instance](#) (*taiga.models.models.Priorities attribute*), 30
[instance](#) (*taiga.models.models.Projects attribute*), 38
[instance](#) (*taiga.models.models.Roles attribute*), 39
[instance](#) (*taiga.models.models.Severities attribute*), 40
[instance](#) (*taiga.models.models.SwimLanes attribute*), 42
[instance](#) (*taiga.models.models.TaskAttachments attribute*), 44
[instance](#) (*taiga.models.models.TaskAttributes attribute*), 46
[instance](#) (*taiga.models.models.Tasks attribute*), 48
[instance](#) (*taiga.models.models.TaskStatuses attribute*), 47
[instance](#) (*taiga.models.models.Users attribute*), 55
[instance](#) (*taiga.models.models.UserStories attribute*), 49
[instance](#) (*taiga.models.models.UserStoryAttachments attribute*), 52
[instance](#) (*taiga.models.models.UserStoryAttributes attribute*), 53
[instance](#) (*taiga.models.models.UserStoryStatuses attribute*), 54
[instance](#) (*taiga.models.models.Webhooks attribute*), 56
[instance](#) (*taiga.models.models.WikiAttachments attribute*), 57
[instance](#) (*taiga.models.models.WikiLinks attribute*), 58
[instance](#) (*taiga.models.models.WikiPages attribute*), 60
[InstanceResource](#) (*class in taiga.models.base*), 60
[Issue](#) (*class in taiga.models.models*), 19
[IssueAttachment](#) (*class in taiga.models.models*), 20
[IssueAttachments](#) (*class in taiga.models.models*), 20
[IssueAttribute](#) (*class in taiga.models.models*), 21
[IssueAttributes](#) (*class in taiga.models.models*), 21
[Issues](#) (*class in taiga.models.models*), 24
[issues_stats\(\)](#) (*taiga.models.models.Project method*), 36
[IssueStatus](#) (*class in taiga.models.models*), 22
[IssueStatuses](#) (*class in taiga.models.models*), 23
[IssueType](#) (*class in taiga.models.models*), 23
[IssueTypes](#) (*class in taiga.models.models*), 24

L

[like\(\)](#) (*taiga.models.models.Project method*), 36
[list\(\)](#) (*taiga.models.base.ListResource method*), 61

- `list()` (*taiga.models.models.Attachments method*), 7
 - `list()` (*taiga.models.models.CustomAttributes method*), 9
 - `list()` (*taiga.models.models.EpicAttachments method*), 12
 - `list()` (*taiga.models.models.EpicAttributes method*), 13
 - `list()` (*taiga.models.models.Epics method*), 15
 - `list()` (*taiga.models.models.EpicStatuses method*), 14
 - `list()` (*taiga.models.models.IssueAttachments method*), 21
 - `list()` (*taiga.models.models.IssueAttributes method*), 22
 - `list()` (*taiga.models.models.Issues method*), 25
 - `list()` (*taiga.models.models.IssueStatuses method*), 23
 - `list()` (*taiga.models.models.IssueTypes method*), 24
 - `list()` (*taiga.models.models.Memberships method*), 26
 - `list()` (*taiga.models.models.Milestones method*), 27
 - `list()` (*taiga.models.models.Points method*), 29
 - `list()` (*taiga.models.models.Priorities method*), 30
 - `list()` (*taiga.models.models.Projects method*), 38
 - `list()` (*taiga.models.models.Roles method*), 39
 - `list()` (*taiga.models.models.Severities method*), 40
 - `list()` (*taiga.models.models.SwimLanes method*), 42
 - `list()` (*taiga.models.models.TaskAttachments method*), 45
 - `list()` (*taiga.models.models.TaskAttributes method*), 46
 - `list()` (*taiga.models.models.Tasks method*), 48
 - `list()` (*taiga.models.models.TaskStatuses method*), 47
 - `list()` (*taiga.models.models.Users method*), 55
 - `list()` (*taiga.models.models.UserStories method*), 49
 - `list()` (*taiga.models.models.UserStoryAttachments method*), 52
 - `list()` (*taiga.models.models.UserStoryAttributes method*), 53
 - `list()` (*taiga.models.models.UserStoryStatuses method*), 54
 - `list()` (*taiga.models.models.Webhooks method*), 56
 - `list()` (*taiga.models.models.WikiAttachments method*), 57
 - `list()` (*taiga.models.models.WikiLinks method*), 59
 - `list()` (*taiga.models.models.WikiPages method*), 60
 - `list_attachments()` (*taiga.models.models.Epic method*), 10
 - `list_attachments()` (*taiga.models.models.Issue method*), 20
 - `list_attachments()` (*taiga.models.models.Task method*), 43
 - `list_attachments()` (*taiga.models.models.UserStory method*), 51
 - `list_attachments()` (*taiga.models.models.WikiPage method*), 59
 - `list_epic_attributes()` (*taiga.models.models.Project method*), 36
 - `list_epics()` (*taiga.models.models.Project method*), 36
 - `list_issue_attributes()` (*taiga.models.models.Project method*), 36
 - `list_issue_statuses()` (*taiga.models.models.Project method*), 36
 - `list_issue_types()` (*taiga.models.models.Project method*), 36
 - `list_issues()` (*taiga.models.models.Project method*), 37
 - `list_memberships()` (*taiga.models.models.Project method*), 37
 - `list_milestones()` (*taiga.models.models.Project method*), 37
 - `list_points()` (*taiga.models.models.Project method*), 37
 - `list_priorities()` (*taiga.models.models.Project method*), 37
 - `list_roles()` (*taiga.models.models.Project method*), 37
 - `list_severities()` (*taiga.models.models.Project method*), 37
 - `list_swimlanes()` (*taiga.models.models.Project method*), 37
 - `list_tags()` (*taiga.models.models.Project method*), 37
 - `list_task_attributes()` (*taiga.models.models.Project method*), 37
 - `list_task_statuses()` (*taiga.models.models.Project method*), 37
 - `list_tasks()` (*taiga.models.models.UserStory method*), 51
 - `list_user_stories()` (*taiga.models.models.Epic method*), 10
 - `list_user_stories()` (*taiga.models.models.Project method*), 37
 - `list_user_story_attributes()` (*taiga.models.models.Project method*), 37
 - `list_user_story_statuses()` (*taiga.models.models.Project method*), 37
 - `list_webhooks()` (*taiga.models.models.Project method*), 37
 - `list_wikilinks()` (*taiga.models.models.Project method*), 37
 - `list_wikipages()` (*taiga.models.models.Project method*), 37
 - `ListResource` (*class in taiga.models.base*), 61
- ## M
- `me()` (*taiga.client.TaigaAPI method*), 6
 - `Membership` (*class in taiga.models.models*), 25
 - `Memberships` (*class in taiga.models.models*), 26
 - `Milestone` (*class in taiga.models.models*), 27
 - `Milestones` (*class in taiga.models.models*), 27
 - `module`
 - `taiga.client`, 5
 - `taiga.models.base`, 60

- taiga.models.models, 6
 MoveOnDestroyMixinList (class in taiga.models.models), 28
 MoveOnDestroyMixinObject (class in taiga.models.models), 28
- ## P
- parse() (taiga.models.base.InstanceResource class method), 61
 parse() (taiga.models.base.ListResource class method), 61
 parse() (taiga.models.models.Attachment class method), 6
 parse() (taiga.models.models.Attachments class method), 7
 parse() (taiga.models.models.CommentableResource class method), 8
 parse() (taiga.models.models.CustomAttribute class method), 8
 parse() (taiga.models.models.CustomAttributeResource class method), 9
 parse() (taiga.models.models.CustomAttributes class method), 10
 parse() (taiga.models.models.Epic class method), 10
 parse() (taiga.models.models.EpicAttachment class method), 11
 parse() (taiga.models.models.EpicAttachments class method), 12
 parse() (taiga.models.models.EpicAttribute class method), 12
 parse() (taiga.models.models.EpicAttributes class method), 13
 parse() (taiga.models.models.Epics class method), 15
 parse() (taiga.models.models.EpicStatus class method), 14
 parse() (taiga.models.models.EpicStatuses class method), 14
 parse() (taiga.models.models.History class method), 15
 parse() (taiga.models.models.Issue class method), 20
 parse() (taiga.models.models.IssueAttachment class method), 20
 parse() (taiga.models.models.IssueAttachments class method), 21
 parse() (taiga.models.models.IssueAttribute class method), 21
 parse() (taiga.models.models.IssueAttributes class method), 22
 parse() (taiga.models.models.Issues class method), 25
 parse() (taiga.models.models.IssueStatus class method), 23
 parse() (taiga.models.models.IssueStatuses class method), 23
 parse() (taiga.models.models.IssueType class method), 24
 parse() (taiga.models.models.IssueTypes class method), 24
 parse() (taiga.models.models.Membership class method), 26
 parse() (taiga.models.models.Memberships class method), 26
 parse() (taiga.models.models.Milestone class method), 27
 parse() (taiga.models.models.Milestones class method), 28
 parse() (taiga.models.models.Point class method), 28
 parse() (taiga.models.models.Points class method), 29
 parse() (taiga.models.models.Priorities class method), 30
 parse() (taiga.models.models.Priority class method), 31
 parse() (taiga.models.models.Project class method), 37
 parse() (taiga.models.models.Projects class method), 39
 parse() (taiga.models.models.Role class method), 39
 parse() (taiga.models.models.Roles class method), 40
 parse() (taiga.models.models.Severities class method), 41
 parse() (taiga.models.models.Severity class method), 41
 parse() (taiga.models.models.SwimLane class method), 41
 parse() (taiga.models.models.SwimLanes class method), 42
 parse() (taiga.models.models.Task class method), 43
 parse() (taiga.models.models.TaskAttachment class method), 44
 parse() (taiga.models.models.TaskAttachments class method), 45
 parse() (taiga.models.models.TaskAttribute class method), 45
 parse() (taiga.models.models.TaskAttributes class method), 46
 parse() (taiga.models.models.Tasks class method), 48
 parse() (taiga.models.models.TaskStatus class method), 46
 parse() (taiga.models.models.TaskStatuses class method), 47
 parse() (taiga.models.models.User class method), 48
 parse() (taiga.models.models.Users class method), 55
 parse() (taiga.models.models.UserStories class method), 49
 parse() (taiga.models.models.UserStory class method), 51
 parse() (taiga.models.models.UserStoryAttachment class method), 51
 parse() (taiga.models.models.UserStoryAttachments class method), 52
 parse() (taiga.models.models.UserStoryAttribute class method), 52
 parse() (taiga.models.models.UserStoryAttributes class

- `method`), 53
- `parse()` (`taiga.models.models.UserStoryStatus` class method), 54
- `parse()` (`taiga.models.models.UserStoryStatuses` class method), 55
- `parse()` (`taiga.models.models.Webhook` class method), 56
- `parse()` (`taiga.models.models.Webhooks` class method), 56
- `parse()` (`taiga.models.models.WikiAttachment` class method), 57
- `parse()` (`taiga.models.models.WikiAttachments` class method), 58
- `parse()` (`taiga.models.models.WikiLink` class method), 58
- `parse()` (`taiga.models.models.WikiLinks` class method), 59
- `parse()` (`taiga.models.models.WikiPage` class method), 59
- `parse()` (`taiga.models.models.WikiPages` class method), 60
- `parse_list()` (`taiga.models.base.ListResource` method), 61
- `parse_list()` (`taiga.models.models.Attachments` method), 7
- `parse_list()` (`taiga.models.models.CustomAttributes` method), 10
- `parse_list()` (`taiga.models.models.EpicAttachments` method), 12
- `parse_list()` (`taiga.models.models.EpicAttributes` method), 13
- `parse_list()` (`taiga.models.models.Epics` method), 15
- `parse_list()` (`taiga.models.models.EpicStatuses` method), 14
- `parse_list()` (`taiga.models.models.IssueAttachments` method), 21
- `parse_list()` (`taiga.models.models.IssueAttributes` method), 22
- `parse_list()` (`taiga.models.models.Issues` method), 25
- `parse_list()` (`taiga.models.models.IssueStatuses` method), 23
- `parse_list()` (`taiga.models.models.IssueTypes` method), 24
- `parse_list()` (`taiga.models.models.Memberships` method), 26
- `parse_list()` (`taiga.models.models.Milestones` method), 28
- `parse_list()` (`taiga.models.models.Points` method), 29
- `parse_list()` (`taiga.models.models.Priorities` method), 30
- `parse_list()` (`taiga.models.models.Projects` method), 39
- `parse_list()` (`taiga.models.models.Roles` method), 40
- `parse_list()` (`taiga.models.models.Severities` method), 41
- `parse_list()` (`taiga.models.models.SwimLanes` method), 42
- `parse_list()` (`taiga.models.models.TaskAttachments` method), 45
- `parse_list()` (`taiga.models.models.TaskAttributes` method), 46
- `parse_list()` (`taiga.models.models.Tasks` method), 48
- `parse_list()` (`taiga.models.models.TaskStatuses` method), 47
- `parse_list()` (`taiga.models.models.Users` method), 55
- `parse_list()` (`taiga.models.models.UserStories` method), 49
- `parse_list()` (`taiga.models.models.UserStoryAttachments` method), 52
- `parse_list()` (`taiga.models.models.UserStoryAttributes` method), 53
- `parse_list()` (`taiga.models.models.UserStoryStatuses` method), 55
- `parse_list()` (`taiga.models.models.Webhooks` method), 57
- `parse_list()` (`taiga.models.models.WikiAttachments` method), 58
- `parse_list()` (`taiga.models.models.WikiLinks` method), 59
- `parse_list()` (`taiga.models.models.WikiPages` method), 60
- `patch()` (`taiga.models.base.InstanceResource` method), 61
- `patch()` (`taiga.models.models.Attachment` method), 6
- `patch()` (`taiga.models.models.CommentableResource` method), 8
- `patch()` (`taiga.models.models.CustomAttribute` method), 8
- `patch()` (`taiga.models.models.CustomAttributeResource` method), 9
- `patch()` (`taiga.models.models.Epic` method), 11
- `patch()` (`taiga.models.models.EpicAttachment` method), 11
- `patch()` (`taiga.models.models.EpicAttribute` method), 12
- `patch()` (`taiga.models.models.EpicStatus` method), 14
- `patch()` (`taiga.models.models.History` method), 15
- `patch()` (`taiga.models.models.Issue` method), 20
- `patch()` (`taiga.models.models.IssueAttachment` method), 20
- `patch()` (`taiga.models.models.IssueAttribute` method), 21
- `patch()` (`taiga.models.models.IssueStatus` method), 23
- `patch()` (`taiga.models.models.IssueType` method), 24
- `patch()` (`taiga.models.models.Membership` method), 26
- `patch()` (`taiga.models.models.Milestone` method), 27
- `patch()` (`taiga.models.models.Point` method), 28
- `patch()` (`taiga.models.models.Priority` method), 31
- `patch()` (`taiga.models.models.Project` method), 37

patch() (*taiga.models.models.Role method*), 39
 patch() (*taiga.models.models.Severity method*), 41
 patch() (*taiga.models.models.SwimLane method*), 42
 patch() (*taiga.models.models.Task method*), 44
 patch() (*taiga.models.models.TaskAttachment method*), 44
 patch() (*taiga.models.models.TaskAttribute method*), 45
 patch() (*taiga.models.models.TaskStatus method*), 47
 patch() (*taiga.models.models.User method*), 48
 patch() (*taiga.models.models.UserStory method*), 51
 patch() (*taiga.models.models.UserStoryAttachment method*), 51
 patch() (*taiga.models.models.UserStoryAttribute method*), 52
 patch() (*taiga.models.models.UserStoryStatus method*), 54
 patch() (*taiga.models.models.Webhook method*), 56
 patch() (*taiga.models.models.WikiAttachment method*), 57
 patch() (*taiga.models.models.WikiLink method*), 58
 patch() (*taiga.models.models.WikiPage method*), 60
 Point (*class in taiga.models.models*), 28
 Points (*class in taiga.models.models*), 29
 Priorities (*class in taiga.models.models*), 29
 Priority (*class in taiga.models.models*), 30
 Project (*class in taiga.models.models*), 31
 Projects (*class in taiga.models.models*), 38

R

refresh_token() (*taiga.client.TaigaAPI method*), 6
 Role (*class in taiga.models.models*), 39
 Roles (*class in taiga.models.models*), 39

S

search() (*taiga.client.TaigaAPI method*), 6
 SearchableList (*class in taiga.models.base*), 62
 set_attribute() (*taiga.models.models.CustomAttributeResource method*), 9
 set_attribute() (*taiga.models.models.Epic method*), 11
 set_attribute() (*taiga.models.models.Issue method*), 20
 set_attribute() (*taiga.models.models.Task method*), 44
 set_attribute() (*taiga.models.models.UserStory method*), 51
 Severities (*class in taiga.models.models*), 40
 Severity (*class in taiga.models.models*), 41
 star() (*taiga.models.models.Project method*), 37
 starred_projects() (*taiga.models.models.User method*), 49
 stats() (*taiga.models.models.Milestone method*), 27
 stats() (*taiga.models.models.Project method*), 38
 SwimLane (*class in taiga.models.models*), 41

SwimLanes (*class in taiga.models.models*), 42

T

taiga.client
 module, 5
 taiga.models.base
 module, 60
 taiga.models.models
 module, 6
 TaigaAPI (*class in taiga.client*), 5
 Task (*class in taiga.models.models*), 42
 TaskAttachment (*class in taiga.models.models*), 44
 TaskAttachments (*class in taiga.models.models*), 44
 TaskAttribute (*class in taiga.models.models*), 45
 TaskAttributes (*class in taiga.models.models*), 45
 Tasks (*class in taiga.models.models*), 47
 TaskStatus (*class in taiga.models.models*), 46
 TaskStatuses (*class in taiga.models.models*), 47
 to_dict() (*taiga.models.base.InstanceResource method*), 61
 to_dict() (*taiga.models.models.Attachment method*), 6
 to_dict() (*taiga.models.models.CommentableResource method*), 8
 to_dict() (*taiga.models.models.CustomAttribute method*), 8
 to_dict() (*taiga.models.models.CustomAttributeResource method*), 9
 to_dict() (*taiga.models.models.Epic method*), 11
 to_dict() (*taiga.models.models.EpicAttachment method*), 11
 to_dict() (*taiga.models.models.EpicAttribute method*), 12
 to_dict() (*taiga.models.models.EpicStatus method*), 14
 to_dict() (*taiga.models.models.History method*), 15
 to_dict() (*taiga.models.models.Issue method*), 20
 to_dict() (*taiga.models.models.IssueAttachment method*), 20
 to_dict() (*taiga.models.models.IssueAttribute method*), 21
 to_dict() (*taiga.models.models.IssueStatus method*), 23
 to_dict() (*taiga.models.models.IssueType method*), 24
 to_dict() (*taiga.models.models.Membership method*), 26
 to_dict() (*taiga.models.models.Milestone method*), 27
 to_dict() (*taiga.models.models.Point method*), 29
 to_dict() (*taiga.models.models.Priority method*), 31
 to_dict() (*taiga.models.models.Project method*), 38
 to_dict() (*taiga.models.models.Role method*), 39
 to_dict() (*taiga.models.models.Severity method*), 41
 to_dict() (*taiga.models.models.SwimLane method*), 42
 to_dict() (*taiga.models.models.Task method*), 44
 to_dict() (*taiga.models.models.TaskAttachment method*), 44

- to_dict() (*taiga.models.models.TaskAttribute method*), 45
- to_dict() (*taiga.models.models.TaskStatus method*), 47
- to_dict() (*taiga.models.models.User method*), 49
- to_dict() (*taiga.models.models.UserStory method*), 51
- to_dict() (*taiga.models.models.UserStoryAttachment method*), 51
- to_dict() (*taiga.models.models.UserStoryAttribute method*), 52
- to_dict() (*taiga.models.models.UserStoryStatus method*), 54
- to_dict() (*taiga.models.models.Webhook method*), 56
- to_dict() (*taiga.models.models.WikiAttachment method*), 57
- to_dict() (*taiga.models.models.WikiLink method*), 58
- to_dict() (*taiga.models.models.WikiPage method*), 60
- ## U
- undelete_comment() (*taiga.models.models.HistoryEntity method*), 16
- undelete_comment() (*taiga.models.models.HistoryEpic method*), 16
- undelete_comment() (*taiga.models.models.HistoryIssue method*), 17
- undelete_comment() (*taiga.models.models.HistoryTask method*), 17
- undelete_comment() (*taiga.models.models.HistoryUserStory method*), 18
- undelete_comment() (*taiga.models.models.HistoryWiki method*), 18
- unlike() (*taiga.models.models.Project method*), 38
- unstar() (*taiga.models.models.Project method*), 38
- update() (*taiga.models.base.InstanceResource method*), 61
- update() (*taiga.models.models.Attachment method*), 7
- update() (*taiga.models.models.CommentableResource method*), 8
- update() (*taiga.models.models.CustomAttribute method*), 8
- update() (*taiga.models.models.CustomAttributeResource method*), 9
- update() (*taiga.models.models.Epic method*), 11
- update() (*taiga.models.models.EpicAttachment method*), 11
- update() (*taiga.models.models.EpicAttribute method*), 12
- update() (*taiga.models.models.EpicStatus method*), 14
- update() (*taiga.models.models.History method*), 16
- update() (*taiga.models.models.Issue method*), 20
- update() (*taiga.models.models.IssueAttachment method*), 20
- update() (*taiga.models.models.IssueAttribute method*), 21
- update() (*taiga.models.models.IssueStatus method*), 23
- update() (*taiga.models.models.IssueType method*), 24
- update() (*taiga.models.models.Membership method*), 26
- update() (*taiga.models.models.Milestone method*), 27
- update() (*taiga.models.models.Point method*), 29
- update() (*taiga.models.models.Priority method*), 31
- update() (*taiga.models.models.Project method*), 38
- update() (*taiga.models.models.Role method*), 39
- update() (*taiga.models.models.Severity method*), 41
- update() (*taiga.models.models.SwimLane method*), 42
- update() (*taiga.models.models.Task method*), 44
- update() (*taiga.models.models.TaskAttachment method*), 44
- update() (*taiga.models.models.TaskAttribute method*), 45
- update() (*taiga.models.models.TaskStatus method*), 47
- update() (*taiga.models.models.User method*), 49
- update() (*taiga.models.models.UserStory method*), 51
- update() (*taiga.models.models.UserStoryAttachment method*), 51
- update() (*taiga.models.models.UserStoryAttribute method*), 53
- update() (*taiga.models.models.UserStoryStatus method*), 54
- update() (*taiga.models.models.Webhook method*), 56
- update() (*taiga.models.models.WikiAttachment method*), 57
- update() (*taiga.models.models.WikiLink method*), 58
- update() (*taiga.models.models.WikiPage method*), 60
- upvote() (*taiga.models.models.Issue method*), 20
- User (*class in taiga.models.models*), 48
- Users (*class in taiga.models.models*), 55
- UserStories (*class in taiga.models.models*), 49
- UserStory (*class in taiga.models.models*), 49
- UserStoryAttachment (*class in taiga.models.models*), 51
- UserStoryAttachments (*class in taiga.models.models*), 51
- UserStoryAttribute (*class in taiga.models.models*), 52
- UserStoryAttributes (*class in taiga.models.models*), 53
- UserStoryStatus (*class in taiga.models.models*), 53
- UserStoryStatuses (*class in taiga.models.models*), 54
- ## W
- Webhook (*class in taiga.models.models*), 55
- Webhooks (*class in taiga.models.models*), 56
- WikiAttachment (*class in taiga.models.models*), 57
- WikiAttachments (*class in taiga.models.models*), 57
- WikiLink (*class in taiga.models.models*), 58
- WikiLinks (*class in taiga.models.models*), 58
- WikiPage (*class in taiga.models.models*), 59
- WikiPages (*class in taiga.models.models*), 60