
todoist-python Documentation

Release 1.0

Doist

Aug 16, 2018

Contents

1	Modules	3
1.1	todoist	3
2	Indices and tables	13
	Python Module Index	15

The official Todoist Python API library.

1.1 todoist

1.1.1 todoist.api

exception `todoist.api.SyncError`

Bases: `exceptions.Exception`

class `todoist.api.TODOISTAPI` (*token=""*, *api_endpoint='https://todoist.com'*, *session=None*,
cache='~/todoist-sync/')

Bases: `object`

Implements the API that makes it possible to interact with a Todoist user account and its data.

classmethod `deserialize` (*data*)

reset_state ()

serialize ()

get_api_url ()

generate_uuid ()

Generates a uuid.

sync (*commands=None*)

Sends to the server the changes that were made locally, and also fetches the latest updated data from the server.

commit (*raise_on_error=True*)

Commits all requests that are queued. Note that, without calling this method none of the changes that are made to the objects are actually synchronized to the server, unless one of the aforementioned Sync API calls are called directly.

query (*queries, **kwargs*)

DEPRECATED: query endpoint is deprecated for a long time and this method will be removed in the next major version of todoist-python

add_item (*content*, ***kwargs*)
Adds a new task.

`todoist.api.state_default` (*obj*)

`todoist.api.json_default` (*obj*)

1.1.2 todoist.models

class `todoist.models.Model` (*data*, *api*)
Bases: `object`

Implements a generic object.

class `todoist.models.Collaborator` (*data*, *api*)
Bases: `todoist.models.Model`

Implements a collaborator.

delete (*project_id*)
Deletes a collaborator from a shared project.

class `todoist.models.CollaboratorState` (*data*, *api*)
Bases: `todoist.models.Model`

Implements a collaborator state.

class `todoist.models.Filter` (*data*, *api*)
Bases: `todoist.models.Model`

Implements a filter.

update (***kwargs*)
Updates filter.

delete ()
Deletes filter.

class `todoist.models.Item` (*data*, *api*)
Bases: `todoist.models.Model`

Implements an item.

update (***kwargs*)
Updates item.

delete ()
Deletes item.

move (*to_project*)
Moves item to another project.

close ()
Marks item as closed

complete (*force_history=0*)
Marks item as completed.

uncomplete (*update_item_orders=1*, *restore_state=None*)
Marks item as not completed.

update_date_complete (*new_date_utc=None*, *date_string=None*, *is_forward=None*)
Completes a recurring task.


```

class todoist.models.Label (data, api)
    Bases: todoist.models.Model

    Implements a label.

    update (**kwargs)
        Updates label.

    delete ()
        Deletes label.

class todoist.models.LiveNotification (data, api)
    Bases: todoist.models.Model

    Implements a live notification.

class todoist.models.GenericNote (data, api)
    Bases: todoist.models.Model

    Implements a note.

    local_manager = None
        has to be defined in subclasses

    update (**kwargs)
        Updates note.

    delete ()
        Deletes note.

class todoist.models.Note (data, api)
    Bases: todoist.models.GenericNote

    Implement an item note.

class todoist.models.ProjectNote (data, api)
    Bases: todoist.models.GenericNote

    Implement a project note.

class todoist.models.Project (data, api)
    Bases: todoist.models.Model

    Implements a project.

    update (**kwargs)
        Updates project.

    delete ()
        Deletes project.

    archive ()
        Marks project as archived.

    unarchive ()
        Marks project as not archived.

    share (email, message="")
        Shares projects with a user.

    take_ownership ()
        Takes ownership of a shared project.

class todoist.models.Reminder (data, api)
    Bases: todoist.models.Model

```

Implements a reminder.

update (***kwargs*)
 Updates reminder.

delete ()
 Deletes reminder.

1.1.3 todoist.managers

```
class todoist.managers.projects.ProjectsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.AllMixin,
todoist.managers.generic.GetByIdMixin, todoist.managers.generic.SyncMixin

    state_name = 'projects'
    object_type = 'project'

    add (name, **kwargs)
        Creates a local project object.

    update (project_id, **kwargs)
        Updates a project remotely.

    delete (project_ids)
        Deletes a project remotely.

    archive (project_id)
        Marks project as archived remotely.

    unarchive (project_id)
        Marks project as not archived remotely.

    update_orders_indents (ids_to_orders_indents)
        Updates the orders and indents of multiple projects remotely.

    share (project_id, email, message="")
        Shares a project with a user.

    get_archived ()
        Returns archived projects.

    get_data (project_id)
        Returns a project's uncompleted items.

    get (project_id)
        Gets an existing project.

class todoist.managers.items.ItemsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.AllMixin,
todoist.managers.generic.GetByIdMixin, todoist.managers.generic.SyncMixin

    state_name = 'items'
    object_type = 'item'

    add (content, project_id, **kwargs)
        Creates a local item object.

    update (item_id, **kwargs)
        Updates an item remotely.
```

delete (*item_ids*)
 Deletes items remotely.

move (*project_items, to_project*)
 Moves items to another project remotely.

close (*item_id*)
 Marks item as done

complete (*item_ids, force_history=0*)
 Marks items as completed remotely.

uncomplete (*item_ids, update_item_orders=1, restore_state=None*)
 Marks items as not completed remotely.

update_date_complete (*item_id, new_date_utc=None, date_string=None, is_forward=None*)
 Completes a recurring task remotely.

update_orders_indents (*ids_to_orders_indents*)
 Updates the order and indents of multiple items remotely.

update_day_orders (*ids_to_orders*)
 Updates in the local state the day orders of multiple items remotely.

get_completed (*project_id, **kwargs*)
 Returns a project's completed items.

get (*item_id*)
 Gets an existing item.

class `todoist.managers.labels.LabelsManager` (*api*)
 Bases: `todoist.managers.generic.Manager`, `todoist.managers.generic.AllMixin`,
`todoist.managers.generic.GetByIdMixin`, `todoist.managers.generic.SyncMixin`

state_name = 'labels'

object_type = 'label'

add (*name, **kwargs*)
 Creates a local label object.

update (*label_id, **kwargs*)
 Updates a label remotely.

delete (*label_id*)
 Deletes a label remotely.

update_orders (*id_order_mapping*)
 Updates the orders of multiple labels remotely.

get (*label_id*)
 Gets an existing label.

class `todoist.managers.notes.GenericNotesManager` (*api*)
 Bases: `todoist.managers.generic.Manager`, `todoist.managers.generic.AllMixin`,
`todoist.managers.generic.GetByIdMixin`, `todoist.managers.generic.SyncMixin`

object_type = 'note'

update (*note_id, **kwargs*)
 Updates an note remotely.

delete (*note_id*)
 Deletes an note remotely.

```
class todoist.managers.notes.NotesManager (api)
    Bases: todoist.managers.notes.GenericNotesManager

    state_name = 'notes'

    add (item_id, content, **kwargs)
        Creates a local item note object.

    get (note_id)
        Gets an existing note.

class todoist.managers.notes.ProjectNotesManager (api)
    Bases: todoist.managers.notes.GenericNotesManager

    state_name = 'project_notes'

    add (project_id, content, **kwargs)
        Creates a local project note object.

class todoist.managers.filters.FiltersManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.AllMixin,
todoist.managers.generic.GetByIdMixin, todoist.managers.generic.SyncMixin

    state_name = 'filters'

    object_type = 'filter'

    add (name, query, **kwargs)
        Creates a local filter object.

    update (filter_id, **kwargs)
        Updates a filter remotely.

    delete (filter_id)
        Deletes a filter remotely.

    update_orders (id_order_mapping)
        Updates the orders of multiple filters remotely.

    get (filter_id)
        Gets an existing filter.

class todoist.managers.reminders.RemindersManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.AllMixin,
todoist.managers.generic.GetByIdMixin, todoist.managers.generic.SyncMixin

    state_name = 'reminders'

    object_type = 'reminder'

    add (item_id, **kwargs)
        Creates a local reminder object.

    update (reminder_id, **kwargs)
        Updates a reminder remotely.

    delete (reminder_id)
        Deletes a reminder remotely.

    get (reminder_id)
        Gets an existing reminder.

class todoist.managers.locations.LocationsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.AllMixin,
todoist.managers.generic.SyncMixin
```

```

state_name = 'locations'

object_type = None

clear()
    Clears the locations.

class todoist.managers.user.UserManager (api)
    Bases: todoist.managers.generic.Manager

    update (**kwargs)
        Updates the user data.

    update_goals (**kwargs)
        Updates the user's karma goals.

    sync ()

    get (key=None, default=None)

    get_id ()

    login (email, password)
        Logins user, and returns the response received by the server.

    login_with_google (email, oauth2_token, **kwargs)
        Logins user with Google account, and returns the response received by the server.

    register (email, full_name, password, **kwargs)
        Registers a new user.

    delete (current_password, **kwargs)
        Deletes an existing user.

    update_notification_setting (notification_type, service, dont_notify)
        Updates the user's notification settings.

class todoist.managers.invitations.InvitationsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.SyncMixin

    state_name = None

    object_type = 'share_invitation'

    accept (invitation_id, invitation_secret)
        Accepts an invitation to share a project.

    reject (invitation_id, invitation_secret)
        Rejects an invitation to share a project.

    delete (invitation_id)
        Delete an invitation to share a project.

class todoist.managers.collaborators.CollaboratorsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.
    GetByIdMixin, todoist.managers.generic.SyncMixin

    state_name = 'collaborators'

    object_type = None

    delete (project_id, email)
        Deletes a collaborator from a shared project.

class todoist.managers.collaborator_states.CollaboratorStatesManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.SyncMixin

```

```

    state_name = 'collaborator_states'

    object_type = None

    get_by_ids (project_id, user_id)
        Finds and returns the collaborator state based on the project and user ids.

class todoist.managers.biz_invitations.BizInvitationsManager (api)
    Bases: todoist.managers.generic.Manager

    state_name = None

    object_type = None

    accept (invitation_id, invitation_secret)
        Accepts a business invitation to share a project.

    reject (invitation_id, invitation_secret)
        Rejects a business invitation to share a project.

class todoist.managers.live_notifications.LiveNotificationsManager (api)
    Bases: todoist.managers.generic.Manager, todoist.managers.generic.
    GetByIdMixin, todoist.managers.generic.AllMixin, todoist.managers.generic.
    SyncMixin

    state_name = 'live_notifications'

    object_type = None

    set_last_read (id)
        Sets the last known notification.

    mark_read (id)
        Marks notification as read.

    mark_read_all ()
        Marks all notifications as read.

    mark_unread (id)
        Marks notification as unread.

class todoist.managers.generic.Manager (api)
    Bases: object

    state_name = None

    object_type = None

    state

    queue

    token

class todoist.managers.generic.AllMixin
    Bases: object

    all (filt=None)

class todoist.managers.generic.GetByIdMixin
    Bases: object

    get_by_id (obj_id, only_local=False)
        Finds and returns the object based on its id.

```

class todoist.managers.generic.**SyncMixin**

Bases: object

Syncs this specific type of objects.

sync ()

CHAPTER 2

Indices and tables

- genindex
- modindex

t

- `todoist.api`, 3
- `todoist.managers.biz_invitations`, 10
- `todoist.managers.collaborator_states`, 9
- `todoist.managers.collaborators`, 9
- `todoist.managers.filters`, 8
- `todoist.managers.generic`, 10
- `todoist.managers.invitations`, 9
- `todoist.managers.items`, 6
- `todoist.managers.labels`, 7
- `todoist.managers.live_notifications`, 10
- `todoist.managers.locations`, 8
- `todoist.managers.notes`, 7
- `todoist.managers.projects`, 6
- `todoist.managers.reminders`, 8
- `todoist.managers.user`, 9
- `todoist.models`, 4

A

accept() (todoist.managers.biz_invitations.BizInvitationsManager method), 10
 accept() (todoist.managers.invitations.InvitationsManager method), 9
 add() (todoist.managers.filters.FiltersManager method), 8
 add() (todoist.managers.items.ItemsManager method), 6
 add() (todoist.managers.labels.LabelsManager method), 7
 add() (todoist.managers.notes.NotesManager method), 8
 add() (todoist.managers.notes.ProjectNotesManager method), 8
 add() (todoist.managers.projects.ProjectsManager method), 6
 add() (todoist.managers.reminders.RemindersManager method), 8
 add_item() (todoist.api.TODOISTAPI method), 3
 all() (todoist.managers.generic.AllMixin method), 10
 AllMixin (class in todoist.managers.generic), 10
 archive() (todoist.managers.projects.ProjectsManager method), 6
 archive() (todoist.models.Project method), 5

B

BizInvitationsManager (class in todoist.managers.biz_invitations), 10

C

clear() (todoist.managers.locations.LocationsManager method), 9
 close() (todoist.managers.items.ItemsManager method), 7
 close() (todoist.models.Item method), 4
 Collaborator (class in todoist.models), 4
 CollaboratorsManager (class in todoist.managers.collaborators), 9
 CollaboratorState (class in todoist.models), 4
 CollaboratorStatesManager (class in todoist.managers.collaborator_states), 9
 commit() (todoist.api.TODOISTAPI method), 3

complete() (todoist.managers.items.ItemsManager method), 7
 complete() (todoist.models.Item method), 4

D

delete() (todoist.managers.collaborators.CollaboratorsManager method), 9
 delete() (todoist.managers.filters.FiltersManager method), 8
 delete() (todoist.managers.invitations.InvitationsManager method), 9
 delete() (todoist.managers.items.ItemsManager method), 6
 delete() (todoist.managers.labels.LabelsManager method), 7
 delete() (todoist.managers.notes.GenericNotesManager method), 7
 delete() (todoist.managers.projects.ProjectsManager method), 6
 delete() (todoist.managers.reminders.RemindersManager method), 8
 delete() (todoist.managers.user.UserManager method), 9
 delete() (todoist.models.Collaborator method), 4
 delete() (todoist.models.Filter method), 4
 delete() (todoist.models.GenericNote method), 5
 delete() (todoist.models.Item method), 4
 delete() (todoist.models.Label method), 5
 delete() (todoist.models.Project method), 5
 delete() (todoist.models.Reminder method), 6
 deserialize() (todoist.api.TODOISTAPI class method), 3

F

Filter (class in todoist.models), 4
 FiltersManager (class in todoist.managers.filters), 8

G

generate_uuid() (todoist.api.TODOISTAPI method), 3
 GenericNote (class in todoist.models), 5
 GenericNotesManager (class in todoist.managers.notes), 7

get() (todoist.managers.filters.FiltersManager method), 8
 get() (todoist.managers.items.ItemsManager method), 7
 get() (todoist.managers.labels.LabelsManager method), 7
 get() (todoist.managers.notes.NotesManager method), 8
 get() (todoist.managers.projects.ProjectsManager method), 6
 get() (todoist.managers.reminders.RemindersManager method), 8
 get() (todoist.managers.user.UserManager method), 9
 get_api_url() (todoist.api.TODOISTAPI method), 3
 get_archived() (todoist.managers.projects.ProjectsManager method), 6
 get_by_id() (todoist.managers.generic.GetByIdMixin method), 10
 get_by_ids() (todoist.managers.collaborator_states.CollaboratorStatesManager method), 10
 get_completed() (todoist.managers.items.ItemsManager method), 7
 get_data() (todoist.managers.projects.ProjectsManager method), 6
 get_id() (todoist.managers.user.UserManager method), 9
 GetByIdMixin (class in todoist.managers.generic), 10

I

InvitationsManager (class in todoist.managers.invitations), 9
 Item (class in todoist.models), 4
 ItemsManager (class in todoist.managers.items), 6

J

json_default() (in module todoist.api), 4

L

Label (class in todoist.models), 4
 LabelsManager (class in todoist.managers.labels), 7
 LiveNotification (class in todoist.models), 5
 LiveNotificationsManager (class in todoist.managers.live_notifications), 10
 local_manager (todoist.models.GenericNote attribute), 5
 LocationsManager (class in todoist.managers.locations), 8
 login() (todoist.managers.user.UserManager method), 9
 login_with_google() (todoist.managers.user.UserManager method), 9

M

Manager (class in todoist.managers.generic), 10
 mark_read() (todoist.managers.live_notifications.LiveNotificationsManager method), 10
 mark_read_all() (todoist.managers.live_notifications.LiveNotificationsManager method), 10
 mark_unread() (todoist.managers.live_notifications.LiveNotificationsManager method), 10

Model (class in todoist.models), 4
 move() (todoist.managers.items.ItemsManager method), 7
 move() (todoist.models.Item method), 4

N

Note (class in todoist.models), 5
 NotesManager (class in todoist.managers.notes), 7

O

object_type (todoist.managers.biz_invitations.BizInvitationsManager attribute), 10
 object_type (todoist.managers.collaborator_states.CollaboratorStatesManager attribute), 10
 object_type (todoist.managers.collaborators.CollaboratorsManager attribute), 9
 object_type (todoist.managers.filters.FiltersManager attribute), 8
 object_type (todoist.managers.generic.Manager attribute), 10
 object_type (todoist.managers.invitations.InvitationsManager attribute), 9
 object_type (todoist.managers.items.ItemsManager attribute), 6
 object_type (todoist.managers.labels.LabelsManager attribute), 7
 object_type (todoist.managers.live_notifications.LiveNotificationsManager attribute), 10
 object_type (todoist.managers.locations.LocationsManager attribute), 9
 object_type (todoist.managers.notes.GenericNotesManager attribute), 7
 object_type (todoist.managers.projects.ProjectsManager attribute), 6
 object_type (todoist.managers.reminders.RemindersManager attribute), 8

P

Project (class in todoist.models), 5
 ProjectNote (class in todoist.models), 5
 ProjectNotesManager (class in todoist.managers.notes), 8
 ProjectsManager (class in todoist.managers.projects), 6

Q

query() (todoist.api.TODOISTAPI method), 3
 queue (todoist.managers.generic.Manager attribute), 10

R

register() (todoist.managers.user.UserManager method), 9
 reject() (todoist.managers.biz_invitations.BizInvitationsManager method), 10
 reject() (todoist.managers.invitations.InvitationsManager method), 9

Reminder (class in `todoist.models`), 5
 RemindersManager (class in `todoist.managers.reminders`), 8
 reset_state() (`todoist.api.TODOISTAPI` method), 3

S

serialize() (`todoist.api.TODOISTAPI` method), 3
 set_last_read() (`todoist.managers.live_notifications.LiveNotificationsManager` method), 10
 share() (`todoist.managers.projects.ProjectsManager` method), 6
 share() (`todoist.models.Project` method), 5
 state (`todoist.managers.generic.Manager` attribute), 10
 state_default() (in module `todoist.api`), 4
 state_name (`todoist.managers.biz_invitations.BizInvitationsManager` attribute), 10
 state_name (`todoist.managers.collaborator_states.CollaboratorStatesManager` attribute), 9
 state_name (`todoist.managers.collaborators.CollaboratorsManager` attribute), 9
 state_name (`todoist.managers.filters.FiltersManager` attribute), 8
 state_name (`todoist.managers.generic.Manager` attribute), 10
 state_name (`todoist.managers.invitations.InvitationsManager` attribute), 9
 state_name (`todoist.managers.items.ItemsManager` attribute), 6
 state_name (`todoist.managers.labels.LabelsManager` attribute), 7
 state_name (`todoist.managers.live_notifications.LiveNotificationsManager` attribute), 10
 state_name (`todoist.managers.locations.LocationsManager` attribute), 8
 state_name (`todoist.managers.notes.NotesManager` attribute), 8
 state_name (`todoist.managers.notes.ProjectNotesManager` attribute), 8
 state_name (`todoist.managers.projects.ProjectsManager` attribute), 6
 state_name (`todoist.managers.reminders.RemindersManager` attribute), 8
 sync() (`todoist.api.TODOISTAPI` method), 3
 sync() (`todoist.managers.generic.SyncMixin` method), 11
 sync() (`todoist.managers.user.UserManager` method), 9
 SyncError, 3
 SyncMixin (class in `todoist.managers.generic`), 10

T

take_ownership() (`todoist.models.Project` method), 5
`todoist.api` (module), 3
`todoist.managers.biz_invitations` (module), 10
`todoist.managers.collaborator_states` (module), 9
`todoist.managers.collaborators` (module), 9

`todoist.managers.filters` (module), 8
`todoist.managers.generic` (module), 10
`todoist.managers.invitations` (module), 9
`todoist.managers.items` (module), 6
`todoist.managers.labels` (module), 7
`todoist.managers.live_notifications` (module), 10
`todoist.managers.locations` (module), 8
`todoist.managers.notes` (module), 7
`todoist.managers.projects` (module), 6
`todoist.managers.reminders` (module), 8
`todoist.managers.user` (module), 9
`todoist.models` (module), 4
`TODOISTAPI` (class in `todoist.api`), 3
 token (`todoist.managers.generic.Manager` attribute), 10

U

unarchive() (`todoist.models.Project` method), 5
 uncomplete() (`todoist.managers.items.ItemsManager` method), 7
 uncomplete() (`todoist.models.Item` method), 4
 update() (`todoist.managers.filters.FiltersManager` method), 8
 update() (`todoist.managers.items.ItemsManager` method), 6
 update() (`todoist.managers.labels.LabelsManager` method), 7
 update() (`todoist.managers.notes.GenericNotesManager` method), 7
 update() (`todoist.managers.projects.ProjectsManager` method), 6
 update() (`todoist.managers.reminders.RemindersManager` method), 8
 update() (`todoist.managers.user.UserManager` method), 9
 update() (`todoist.models.Filter` method), 4
 update() (`todoist.models.GenericNote` method), 5
 update() (`todoist.models.Item` method), 4
 update() (`todoist.models.Label` method), 5
 update() (`todoist.models.Project` method), 5
 update() (`todoist.models.Reminder` method), 6
 update_date_complete() (`todoist.managers.items.ItemsManager` method), 7
 update_date_complete() (`todoist.models.Item` method), 4
 update_day_orders() (`todoist.managers.items.ItemsManager` method), 7
 update_goals() (`todoist.managers.user.UserManager` method), 9
 update_notification_setting() (`todoist.managers.user.UserManager` method), 9
 update_orders() (`todoist.managers.filters.FiltersManager` method), 8

`update_orders()` (`todoist.managers.labels.LabelsManager`
method), 7

`update_orders_indents()` (`todoist.managers.items.ItemsManager`
method), 7

`update_orders_indents()` (`todoist.managers.projects.ProjectsManager`
method), 6

`UserManager` (class in `todoist.managers.user`), 9