YouTrack Python Client Documentation

version

Tom Mertens

July 08, 2025

Contents

YouTrack Python Client

Usage Guide

Installation

```
poetry add youtrack-client
```

Quickstart

```
from youtrack.client import YouTrackClient
client = YouTrackClient(base_url='https://your-youtrack-instance', token='perm:XXXX')
issues = client.get_issues(project_id='PROJECT')
print(issues)
```

See the API Reference for full API details and CLI Reference for CLI usage.

API Reference

YouTrack API client implementation.

This module provides the YouTrackClient class for interacting with the YouTrack REST API.

Features: - Authenticate with YouTrack using a personal access token - Create, update, search, and list issues - Add comments, transition issues, and attach files - Retrieve issue history and changes - List and manage workitems (time tracking) - Find IDs for projects, issues, users, custom fields, workflows, boards, user stories, and more - Manage agile boards, sprints, and user stories (epics) - Run reports and retrieve deadline calendars - Manage issue links and link types - Run queries and commands on issues

Usage:

> from youtrack.client import YouTrackClient client = YouTrackClient.from_config() issues = client.list_issues(project_id=”0-0”)

See each method’s docstring for details.

exception `youtrack.client.`**`YouTrackError`**

Bases: **`Exception`**

Custom exception for YouTrack API errors with meaningful messages.

class `youtrack.client.`**`YouTrackClient`** (`token`, `base_url`)

Bases: **`object`**

Parameters:

- **token** (*str*) –
- **base_url** (*str*) –

classmethod **`from_config`** **(**`config_path`=None**)**

Load YouTrack credentials from a .youtrack.toml file.

Parameters: **config_path** (*str* | *None*) –

`create_issue` **(**`project_id`, `summary`, `description`='', `custom_fields`=None, `story_points`=None**)**

Create a new issue in the specified project.

Parameters:

- **project_id** (*str*) – The ID of the project.
- **summary** (*str*) – The issue summary/title.
- **description** (*str, optional*) – The issue description.
- **custom_fields** (*dict, optional*) – Custom fields to set.
- **story_points** (*int, optional*) – Value for the ‘Story points’ custom field. Required in some workflows.

Returns: The created issue data.

Return type: dict

`list_issues` **(**`project_id`**,** `query=''`**,** `limit=20`**,** `skip=0`**)**

List issues in a project with optional query and pagination.

Parameters:

- **project_id** (*str*) – The ID of the project.
- **query** (*str, optional*) – YouTrack query string.
- **limit** (*int, optional*) – Max results to return.
- **skip** (*int, optional*) – Results to skip.

Returns: List of issues.

Return type: list

`update_issue` **(**`issue_id`**,** `summary=None`**,** `description=None`**,** `custom_fields=None`**)**

Update an existing issue with new information.

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **summary** (*str, optional*) – New summary.
- **description** (*str, optional*) – New description.
- **custom_fields** (*dict, optional*) – Custom fields to update.

Returns: The updated issue data.

Return type: dict

`search_issues` **(**`query`**,** `limit=20`**,** `skip=0`**)**

Search for issues using a YouTrack query.

Parameters:

- **query** (*str*) – YouTrack query string.
- **limit** (*int, optional*) – Max results to return.
- **skip** (*int, optional*) – Results to skip.

Returns: List of issues.

Return type: list

`add_comment` **(**`issue_id`**,** `text`**)**

Add a comment to an issue.

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **text** (*str*) – The comment text.

Returns: The created comment data.

Return type: dict

`transition_issue` **(**`issue_id`**,** `field_name`**,** `new_state`**)**

Transition an issue to a new workflow state by updating a custom field (e.g., State).

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **field_name** (*str*) – The custom field name (e.g., ‘State’).
- **new_state** (*str*) – The new state value.

Returns: The updated issue data.

Return type: dict

`attach_file` **(**`issue_id`**,** `file_path`**)**

Attach a file to an issue.

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **file_path** (*str*) – Path to the file to attach.

Returns: The attachment data.

Return type: dict

`get_issue_history` **(**`issue_id`**)**

Retrieve the history and changes of an issue.

Parameters: **issue_id** (*str*) – The ID of the issue.

Returns: List of activity records.

Return type: list

`list_workitems` **(**`project_id`**,** `limit=20`**,** `skip=0`**)**

List workitems (time tracking entries) in a project, with pagination support.

Parameters:

- **project_id** (*str*) – The ID of the project.
- **limit** (*int, optional*) – Max results to return.
- **skip** (*int, optional*) – Results to skip.

Returns: List of workitems.

Return type: list

`calculate_time_spent` **(**`issue_id`**)**

Calculate total time spent on an issue by summing its workitems’ durations.

Parameters: **issue_id** (*str*) – The ID of the issue.

Returns: Total time spent (minutes).

Return type: int

`list_workitem_types` **(**`project_id`**)**

List allowed workitem types for a project.

Parameters: **project_id** (*str*) – The ID of the project.

Returns: List of workitem types.

Return type: list

`add_spent_time` **(**`issue_id`**,** `duration`**,** `workitem_type_id`**,** `description=''`**)**

Add spent time (workitem) to an issue. Duration is in minutes. workitem_type_id is required.

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **duration** (*int*) – Time spent in minutes.
- **workitem_type_id** (*str*) – The workitem type ID.
- **description** (*str, optional*) – Description for the workitem.

Returns: The created workitem data.

Return type: dict

`list_projects` **()**

List all projects in the YouTrack instance.

Returns: List of projects, each as a dict with ‘id’, ‘name’, and ‘shortName’.

Return type: list

`get_issue` **(**`issue_id`**)**

Retrieve details for a specific issue by its ID.

Parameters: **issue_id** (*str*) – The ID of the issue.

Returns: Issue details including id, summary, description, and project info.

Return type: dict

`list_users` **(**`query=''`**,** `limit=20`**,** `skip=0`**)**

List users in the YouTrack instance, optionally filtered by a query string.

Parameters:

- **query** (*str, optional*) – Query string to filter users (e.g., by name or email).
- **limit** (*int, optional*) – Maximum number of users to return.
- **skip** (*int, optional*) – Number of users to skip (for pagination).

Returns: List of user dicts with id, login, name, and email.

Return type: list

`list_custom_fields` **(**`project_id`**)**

List custom fields for a given project.

Parameters: **project_id** (*str*) – The ID of the project.

Returns: List of custom fields with id, name, and field type info.

Return type: list

`list_workflows` **()**

List all workflows in the YouTrack instance.

Returns: List of workflows with id, name, and description.

Return type: list

`list_boards` **(**`project_id=None`**)**

List all agile boards. Optionally filter boards by project ID.

Parameters: **project_id** (*str, optional*) – If provided, only boards containing this project are returned.

Returns: List of boards with id, name, and associated projects.

Return type: list

`list_sprints` **(**`board_id`**)**

List all sprints for a given agile board.

Parameters: **board_id** (*str*) – The ID of the agile board.

Returns: List of sprints with id, name, start, finish, and isArchived status.

Return type: list

`list_user_stories` **(**`board_id`**,** `sprint_id=None`**)**

List user stories (epics) on a board, optionally for a specific sprint.

Parameters:

- **board_id** (*str*) – The ID of the agile board.
- **sprint_id** (*str, optional*) – The ID of the sprint. If provided, only user stories in this sprint are listed.

Returns: List of user stories with id, summary, and custom fields.

Return type: list

`add_issue_to_sprint` **(**`board_id`**,** `sprint_id`**,** `issue_id`**)**

Add an issue to a sprint on a specific agile board.

Parameters:

- **board_id** (*str*) – The ID of the agile board.
- **sprint_id** (*str*) – The ID of the sprint.
- **issue_id** (*str*) – The ID of the issue to add.

Returns: The response from the API (usually the updated sprint or issue info).

Return type: dict

`add_issue_to_user_story` **(**`board_id`**,** `user_story_id`**,** `issue_id`**)**

Add an issue as a subtask to a user story (epic) on a board.

Parameters:

- **board_id** (*str*) – The ID of the agile board.
- **user_story_id** (*str*) – The ID of the user story (epic).
- **issue_id** (*str*) – The ID of the issue to add as a subtask.

Returns: The response from the API (usually the updated user story or issue info).

Return type: dict

`add_user_story_to_sprint` **(**`board_id`**,** `sprint_id`**,** `user_story_id`**)**

Add a user story (epic) to a sprint on a board.

Parameters:

- **board_id** (*str*) – The ID of the agile board.
- **sprint_id** (*str*) – The ID of the sprint.
- **user_story_id** (*str*) – The ID of the user story (epic) to add.

Returns: The response from the API (usually the updated sprint or user story info).

Return type: dict

`run_report` **(**`report_id`**)**

Run a report by its ID and return the result.

Parameters: **report_id** (*str*) – The ID of the report to execute.

Returns: The report execution result.

Return type: dict

`authenticate` **()**

Placeholder for authentication logic. Not required for token-based auth.

`get_deadline_calendars` **()**

Retrieve all deadline calendars (holiday calendars) in the instance.

Returns: List of calendars with id, name, and holidays.

Return type: list

`get_issue_links` **(**`issue_id`**)**

Get all links for a specific issue.

Parameters: **issue_id** (*str*) – The ID of the issue.

Returns: List of issue links with id, direction, link type, and linked issues.

Return type: list

`list_issue_link_types` **()**

List all available issue link types in the instance.

Returns: List of link types with id, name, and direction info.

Return type: list

`list_issue_link_types_for_issue` (`issue_id`)
List link types available for a specific issue.

Parameters: **issue_id** (*str*) – The ID of the issue.

Returns: List of link types for the issue.

Return type: list

`list_issue_link_types_for_project` (`project_id`)
List link types available for a specific project.

Parameters: **project_id** (*str*) – The ID of the project.

Returns: List of link types for the project.

Return type: list

`add_issue_link` (`source_issue_id`, `target_issue_id`, `link_type_id`)
Add a link between two issues using a specific link type.

Parameters:

- **source_issue_id** (*str*) – The ID of the source issue.
- **target_issue_id** (*str*) – The ID of the target issue.
- **link_type_id** (*str*) – The ID of the link type to use.

Returns: The response from the API (usually the updated link info).

Return type: dict

`run_query` (`query`, `fields`='id,summary,description', `limit`=20, `skip`=0)
Run a search query on issues, returning selected fields.

Parameters:

- **query** (*str*) – YouTrack query string.
- **fields** (*str, optional*) – Comma-separated fields to return for each issue.
- **limit** (*int, optional*) – Max results to return.
- **skip** (*int, optional*) – Results to skip.

Returns: List of issues matching the query.

Return type: list

`run_command` (`issue_id`, `command`, `comment`=None)
Run a command on an issue (e.g., change state, assign, add comment, etc.).

Parameters:

- **issue_id** (*str*) – The ID of the issue.
- **command** (*str*) – The command string to execute (YouTrack command language).
- **comment** (*str, optional*) – Optional comment to add with the command.

Returns: The response from the API (usually the updated issue info).

Return type: dict

CLI Reference

```
youtrack --help
```

Note

The CLI exposes all public methods of the API client. See `youtrack <command> --help` for details.

# For advanced CLI docs, consider using sphinx-argparse or manual command listing.

User Documentation

Welcome to the YouTrack Python Client documentation!

- For installation and quickstart, see Usage Guide.
- For API reference, see API Reference.
- For CLI usage, see CLI Reference.

Indices and tables

- **`genindex`**
- **`modindex`**
- **`search`**

Index

Python Module Index