

## API

Since [Tiki24](#) an API has been available, leveraging [swagger-api/swagger-ui](#)

See it in action here: <https://doc.tiki.org/api/>

First commit: [https://gitlab.com/tikiwiki/tiki/-/merge\\_requests/1028](https://gitlab.com/tikiwiki/tiki/-/merge_requests/1028)

### Tiki 27+

Significant updates made in [Tiki27](#), particularly to support Internet of Things (IoT) deployments, enhancing and adding support for:

- Trackers
- File galleries

More detail on the use of the API for IoT deployment can be found [here](#).

### Tiki 24+

A self-documented REST API is available since Tiki 24. This new feature is exposing the most commonly used elements of the system, notably:

- Categories
- Comments
- Groups
- Search
- Trackers
- Translation
- Users
- and Wiki

To start using Tiki API, you may need to refer to this [documentation](#) which details its endpoints.

### Requirements

`.htaccess` file must be enabled to make the `/api/` URL work, which is standard practice to have [SEFURLs](#).

### Getting Started

Enable the preference `auth_api_tokens` via the admin page.

### Documentation

The Documentation is embedded in Tiki. See `/api/` on your target Tiki 24+ installation for an [OpenAPI 3.0 documentation](#) of the API.

See [GET/version reference](#) in documentation.

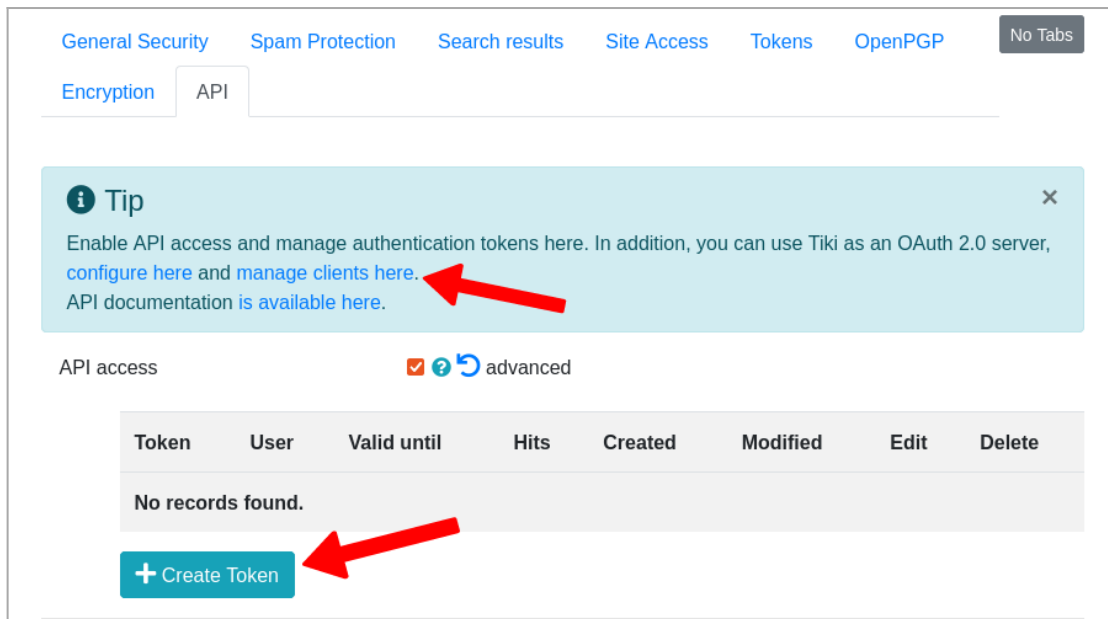
### Authorization

API requests should be authenticated with a token created by Tiki admin (via Admin -> Security tab). Each token gives their owner access with one and only one Tiki user. [Permissions](#) configuration is then based on that Tiki [user's groups](#).

Tokens can be created in two ways:

1. Using Tiki OAuth 2.0 server. The documentation contains endpoints and parameters for different grant types.

2. Manually, in the Control Panel via Admin -> Security tab. Each token is associated with a user. Any API call using the token will act as the user observing all user's permissions.



Using Tiki as OAuth 2.0 server or Create a user token

## OAuth 2.0 Server

[OAuth 2](#) provides authorization flows for third-party applications.

Authorization flow can be:

1. Machine-to-machine - use client authorization grant type. Send your credentials directly to `access_token` endpoint to retrieve the access token.
2. End-user-to-machine - use auth flow grant type. Start by sending the user to authorize endpoint. This allows Tiki to ask target user for permission to grant access token with their user privileges. Once agreed, user is redirected back to your app/web app/machine where you do a machine-to-machine request to `access_token` endpoint to get the actual access token.

Access tokens generated by Tiki OAuth server are JWT encoded.

## Tiki Restful API Coverage

[CRUD operations](#) (Create, Read, Update and Delete) are available for Category, Comments, Groups, Trackers/Fields/Items, Users and Wiki pages.

The endpoints include:

1. Authorization flow.
2. API version.
3. Category: Object categorization and and CRUD.
4. Comments: Thread locking, moderation and CRUD.
5. Groups: User association and CRUD.
6. Search index rebuild and lookup.
7. Trackers/Fields/Items: Special features like dump/export, clone, duplicate, clear and CRUD.
8. Manage object translations.
9. User registration and CRUD operations, messaging and emailing wiki pages.
10. Wiki pages: Locking and parsing/display and CRUD.

Major items in wishlist for next versions of the API:

1. Files and file galleries (added in Tiki27)

2. Articles, blogs, other wiki-related elements.
3. Calendars.

See all the references in the [documentation](#).

Example Tracker API usage with JavaScript here <https://dev.tiki.org/API-Access-Example>.

Pre-Tiki 24 notes

[+]

Aliases

- [service URL](#)
- [URL arguments](#)