

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

Tiki 27+

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

- Trackers
- File galleries

A short summary of the use of the API for IoT deployment can be found on the next page with more detailed information available [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.

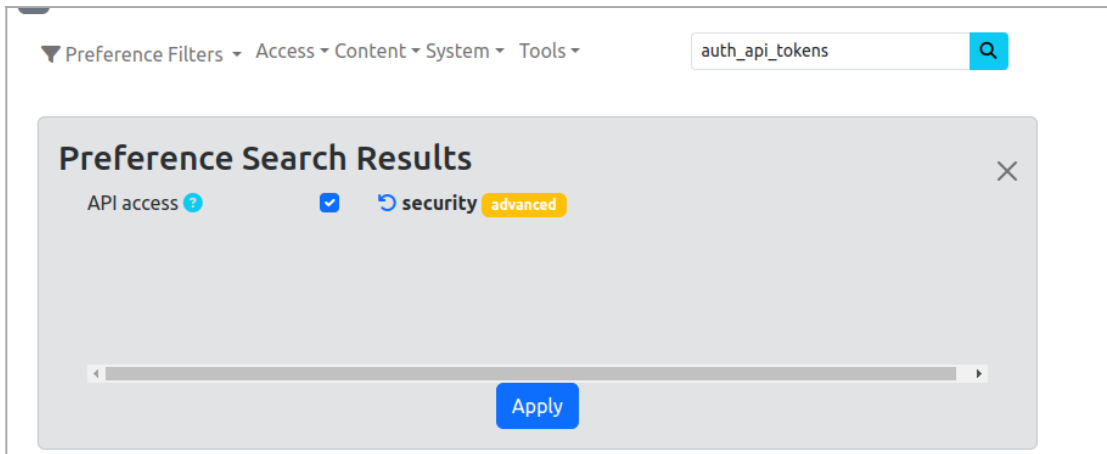
Getting Started

Enable the preference `auth_api_tokens` via the admin page.

Enable API access

In the Control Panel, the screenshot below shows how to enable `auth_api_tokens` labeled as API access:

- Check the box "API access"
- Click "Apply" to save your choice.



Enable API access

Documentation

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

i Documentation page

For example, let's assume that your Tiki 24+ instance is installed on `https://example.org`, then the page under `https://example.org/api/` should look alike:



API Documentation page

Example: Let's `GET` the API version

GET/version Request

Assuming `example.org` has API feature enabled, this code snippet,

```
$ curl --request GET 'http://example.org/api/version'
```

Should return the following output as reponse:

```
{ "version": "24.1vcs" }
```

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

Authorization token

Bearer token authorization header in format:

```
Authorization: Bearer TOKEN
```

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.

General Security Spam Protection Search results Site Access Tokens OpenPGP No Tabs

Encryption API

Tip ×

Enable API access and manage authentication tokens here. In addition, you can use Tiki as an OAuth 2.0 server, [configure here](#) and [manage clients here](#). API documentation is available [here](#).

API access   advanced

Token	User	Valid until	Hits	Created	Modified	Edit	Delete
No records found.							

[+ Create Token](#)

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.

i Tiki can act as an OAuth server.

In the Control Panel, Admin -> Security tab has a link to manage authenticated clients. This section creates client IDs and secrets for web, desktop or mobile applications using Tiki API.

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

[+]

Tiki can support wiki page updates (or data for most features in Tiki) by setting up data channels at this time. Adding a web service call to do it more directly would not be very hard so you can join in and get your commit access!

From tiki-admin.php, you can activate: "HTTP Basic Authentication: Check credentials from HTTP Basic Authentication" useful to allow webservices to use credentials." Related: Token Access

Use of Tiki services

Tiki's services live in `lib/core/Services/`. One can extrapolate the service URL from the file names and the names of the classes in the `Controller.php` files.

This only works if you have activated SEFURL feature.

Example:

For accessing information which is also available from searches in the Tiki site, the class is `action_lookup()` in file `lib/core/Services/Search/Controller.php`.

The path on tiki.org is: <https://tiki.org/tiki-search-lookup>

To refine the search, the arguments are the same as for <https://doc.tiki.org/PluginList-filter-control-block>

For example, in order to access the 45 first items from tracker 22, the syntax would be:

https://tiki.org/tiki-search-lookup?filter~type=trackeritem&filter~tracker_id=22&maxRecords=45

This works fine if called as ajax services from a page on the same Tiki.

If done from outside Tiki from another online server, only data visible for *anonymous user* (not logged in) will be shown.

In order to access data which is not visible to *anonymous user*, you may want to have a look at <https://doc.tiki.org/Token-Access>.

Controller pages

Where you can find more specific information and samples for the different controllers

- API Tracker

Aliases

- service URL
- URL arguments