Tiki API Knowledge Sharing 2021

Recording:
https://recordings.rna1.blindsidenetworks.com/citadelrock/d559965849921585c1849af03b7a51638700d979-1637863177279/presentation/

Time confirmed: 1800 UTC 25-Nov-2021
https://www.timeanddate.com/worldclock/fixedtime.html?iso=20211125T18&ah=2 on https://tiki.org/live

On the mailing list, there is a discussion on Tiki API (please login to access the mirror forum) developments. First, I am really happy that some progress is being made on this.

About 2 months ago, we (i.e. the company I'm with) also started developing a REST API for our SaaS platform which uses Tiki. Right now, we are developing a mobile app which uses this API via OAuth2 login. I'd like to share our experience so far, and also the generic parts of our code so that it can be helpful to the people who are developing the Tiki API.

So I have decided to organize 2 webinar sessions to walk through what we have done and the lessons learned so far. There will be time for questions and answers so come prepared with all your questions (I have seen quite a number in the mailing list already). After the technical session, I will arrange to hand over code snippets to the developers (e.g. Victor) who are working on the Tiki API.

Please indicate the times you are available using the Convene plugins below and I will choose the times that most people can attend, with some judgement involved, for example Victor and other key people would need to be there.

During the sessions, I will try to touch on all of the points raised by Victor's original post to the mailing list (copied below for reference), and also questions raised by others in response to his original post. The spirit of these sessions is simply to share knowledge and experience, as I am confident in the community's combined wisdom in the eventual path of the Tiki API. As we merge in new versions of Tiki, I'm confident it will interoperate fine with our own platform API.

Some may be wondering why we need our own platform API and can't just use the Tiki API. The answer to that is our API endpoints need to have request/response values that are very specific to our use case. As a SaaS platform, we wouldn't want to offer an API that allows everything that is possible in standard Tiki for a particular object type, but rather it needs to be highly restrictive. Moreover,the terminology is totally higher-level and different, e.g. we don't want to offer an API for a user to add an item to tracker ID x but instead we want an API for adding a which is what that tracker stores. There are also other complexities, for example, we have integrated a non-open source filtering and moderation system

. Having said that, it is always preferable to extend generic functionality than to be totally custom, so the aim is to share anything that is generic.

- Nelson

Time confirmed: 1800 UTC 25-Nov-2021
https://www.timeanddate.com/worldclock/fixedtime.html?iso=20211125T18&ah=2 on https://tiki.org/live

Combined session (both technical/non-technical) - 1 hour

	25 Nov 2021 18:00 GMT-0000	25 Nov 2021 19:00 GMT-0000	25 Nov 2021 21:00 GMT-0000	26 Nov 2021 15:00 GMT-0000	26 Nov 2021 16:00 GMT-0000	26 Nov 2021 17:00 GMT-0000	26 Nov 2021 18:00 GMT-0000	26 Nov 2021 21:00 GMT-0000
Nelson Ko								

Jonny Bradley								
luciash d' being ▯								
Victor Emanouilov								
Torsten Fabricius								
	5	3	1	1	1	3	2	0

Time confirmed: We'll try and cover everything in the above combined session and then we can continue (e.g. Victor and Nelson) as needed in the time immediately after.

Technical session - 1 hour

	25 Nov 2021 18:00 GMT-0000	25 Nov 2021 19:00 GMT-0000	25 Nov 2021 21:00 GMT-0000	26 Nov 2021 15:00 GMT-0000	26 Nov 2021 16:00 GMT-0000	26 Nov 2021 17:00 GMT-0000	26 Nov 2021 18:00 GMT-0000	26 Nov 2021 21:00 GMT-0000
Nelson Ko								
Jonny Bradley								
Victor Emanouilov								
Torsten Fabricius								
Bernard Sfez / Tiki Specialist								
	5	3	0	1	1	2	2	0

Victor's original post to the mailing list (for reference)

> *From: Victor Emanouilov - 2021-11-17 12:34:07*
>
> *Dear dev community,*
>
> *We are adding an API to Tiki 24. Your input will be highly valuable.*
>
> *As discussed previously with Jonny, Marc and others, first step was exploring the opportunity of exposing our ajax-related services as an API. This makes sense but also has some drawbacks discussed below. Main benefits:*
> *- extremely quick start - we have many services accessible via ajax for internal Tiki operations that can be exposed as an API for external systems to retrieve data and manipulate data*
> *- having one and the same code used by internal components and external services will ensure interoperability in the future and same behavior inside and outside Tiki*
> *- avoid designing and especially coding a whole new level of services for the API*
> *- opportunity to enhance existing ajax services with missing pieces like permission enforcement and documentation*

> *Main drawbacks:*
> - *lack the possibility of designing an API around an idea (resources/restful or graphql)*
> - *lack versioning support - changing an ajax service in Tiki automatically changes the API which possibly breaks existing integrations (though we can try to be as much backward-compatible as possible and also provide a list of breaking changes when upgrading Tiki)*
> - *existing services are not restful, neither graph-based, so the resulting API is non-standard*
> - *some of the existing services are not designed to be exposed via API - e.g. Cypht ajax controllers*
>
> *Having these points in mind, I started a MR with the quick-win method of exposing Tiki ajax services here:*
> *https://gitlab.com/tikiwiki/tiki/-/merge_requests/1028*
>
> *Solved some of the problems with authentication (currently bypassing all standard Tiki auth methods and allowing bearer token authentication only for now), CSRF (turning it off for API but also disabling session cookies, so CSRF are not possible via the API), ensure requests are stateless and no JS or other funky stuff is going on. Now we have the remaining TODO items like documenting the services, enforcing permissions, consider versioning and restful resources. I consider 3 possible ways to go forward but needed your input before investing more time:*
>
> *1. Leave the API urls like now 100% based on the ajax services. Typical URL is tiki.org/api/controller/action?param1=val1&param2=val2, e.g. tiki.org/api/tracker/view?id=12. Enhance services with permissions (where missing), document endpoints, input parameters and output. Versioning seems impossible in this case. At least the standard versioning we are used to see. Maybe we can do versioning based on Tiki version - API version 24 is one and the same for all Tiki 24 releases, then we have version 25, etc...*
>
> *2. Add an adapter between ajax services and API. Make adapter versionable (so we have API versions - any time an ajax service is changed, we keep old behavior and add a new adapter for the new version). Adapter will also help us come up with more standard restful API endpoints and input/output. It can skip certain ajax services that doesn't make sense to be exposed. Permissions should still be in the services themselves as currently, it is possible to get any object attribute without even logging in Tiki with a URL like this*
> *https://tiki.org/tiki-ajax_services.php?controller=attribute&action=get&type=trackeritem&object=2856&attribute=tiki.geo.lat.*
> *Documentation will be based on the newly written adapter rather than existing ajax services. Will take more time but solve most of the drawbacks. This is my preferred option.*
>
> *3. Design, write and document an API from scratch. We can make use of existing ajax services in terms of code but don't depend on their controller/action architecture - here we have the utmost flexibility to design a graphql or restful API but also the most time-consuming. Without a specific use-case, a project or a subset of Tiki-stored data*

> *and functionality to expose, I think it will be a waste of time for now. Also, one more bit here that's pretty important is the fact that Tiki as a web application is stateful in so many ways - it has extensive use of the PHP session that is not available in an API. Adding a proper stateless API requires refactoring all those interal code elements to depend on incoming params or database rather than the session which is a daunting task to even think about. Tiki will certainly benefit from this refactor but at a price of a huge time commitment.*
>
> *I also have a question about API documentation. We have a bunch of options here with most prominent ones being Swagger (OpenAPI), Raml and API blueprint - all of which require a separate set of documentation sources (in yaml, json or markdown format). We will need to go through existing services and document endpoints, required params, input, output, etc. Definitely a big task to do. I think there is also an option to use inline documentation - PHP class and method annotation documentation that can be exported to a visually appealing html doc. Not as robust as the first set of tools I mentioned but having the docs and the code in one place has its benefits (easier updates for example). Anyone has any preferences here?*
>
> *Thanks!*
> *Victor*