

Please use comments for discussion points. Lets keep these requests clean and easy to read for the benefit of everyone involved. THERE IS A DOC with the old comments. Ask TnT or H31IX if you want access.

High Priority

- **#1 Priority:**
- **BukkitDev requires notifications before total conversion from the current forum based system can occur. Our forums are not meant to host projects, and is very cumbersome to maintain on top of dev.bukkit.org (DBO). This is required to be able to kill off our forum sections and move entirely to DBO.**
- Unapproved files can still be linked directly. While this leaves a very small opportunity for malicious plugins to get released into the wild, it is still a concern.
- Due to receiving lots of files that maliciously download a file onto the server which contains a backdoor, we have locked down auto-updaters and enforced that they only download files from BukkitDev. Because of this, we need tools for the authors to get the latest version and a static link to the latest file easily if they are to only use BukkitDev for file downloading.
- API required. For example, one to allow staff to more easily work on the approval queue. Specs to follow. Curse requires these specs to proceed with any sort of API for us.
- Add progress bar for uploading (huge texture packs take forever, and people think the site is broken). **Broken for IE and Firefox users. Chrome works fine.**
- [New - Sept 19th, 2012] Changes required for files as well as projects. Many files have minor issues and currently require PM'ing the dev to make small changes, instead of the more direct approach of redirecting the file back to them to make the changes, thus removing them from the queue until the changes are completed.

Low Priority

- Ability to tag people (@H31IX: hi)
- Some email alerts starting with the text 'Go'
- Forum moderator rank / position per project. (Author, Manager, Artist etc etc etc)
- Replying to something yourself should not trigger an alert (email) for you
- Bukkit Tools section? There are TONS of good tools that we have to turn down every day because we don't support them. (Ex: things like web interfaces and scripts)
- Add the ability to "soft-delete" files
- The ability to clearly mark builds as DEV, BETA, RB, BROKEN would be desirable as it aligns with Bukkit's use of Dev/Beta and RB builds designations. (Colour coding?)
- Possibly integrate with dl.bukkit.org, so when a new version is promoted (beta, RB, dev) it gets a new entry in dev.bukkit.org.
- A "version number" box should be added when uploading files and then appended to the file's name, or just kept separately. We require version numbers in the file's titles on DBO, but nobody does this because they don't know that it's required.
- IE Poll formatting - <http://postimage.org/image/4petgowtn/>

- Replies to projects should be more geared towards BukkitDev, for instance nobody on DBO uses lua, so sending messages about luadoc are irrelevant.
- When a forum section is deleted, it only gets 'marked as deleted' instead of actually being removed from the public. A bit of a weird feature. Should probably be hidden from all but DBO staff.
- Staff's messages should not trigger the spam filter
- When looking at another player's avatar, there should be something to tell if the user is currently online or not
- Add a message to let users know that their file will be approved shortly, we get tons of PM's asking if they did something wrong because their download button isn't appearing
- Added June 28th - Youtube player doesn't properly display 16:9 videos.
<http://forums.bukkit.org/threads/youtube-embed-fix.83587/>
LukeGB note: sort-of fixed in the Bukkit SASS. Please fix in templates!
- July 5th - add a "You are now leaving BukkitDev" warning to any off-site link for security.
- According to the "specification" a linebreak in the markdown should not create a linebreak in the presentation. a simple linebreak in md is introduced by two whitespaces at the end of the line followed by a single linebreak. a paragraph is introduced by two linebreaks in md. but every single linebreak in the editor will be turned into a linebreak in the presentation.
 - Example: this <return>should <return>be <return>a <return>single <return>line. these are<space><space><return>two lines. these are<return><return>two paragraphs.
- Markdown <<reply>> does not work, creates blank "" parameter. See <https://bukkit.atlassian.net/browse/SITE-88> for more info. Example: <<reply 981132="">>. Same as SITE-61

Known bugs/flaws:

- Comment reply doesn't always tag the name)
<http://forums.bukkit.org/threads/reply-to-comments-problem.79447/>
<http://forums.bukkit.org/threads/comment-replies-and-broken-support-for-markup-types.95457/#post-1299567>
- Users do not have a way to block or ignore someone themselves if they are getting PM spammed, they have to contact staff to ban them

**Please Scroll down below for complete
API spec.**

Complete v1.0 Api Spec (md_5 & lukegb):

The API shall revolve around several key points and is presently read-only. At the start there will be three access levels for the APIs.

No Authentication - Does not require any authentication or special data. Will provide static listings of publicly accessible information, eg a lists / information regarding projects and files.

User Authentication - available to everyone with a registered BukkitDev / Curseforge account. In order to access this section the requester will need to specify an api key via either the url parameter '?apikey=... ' or the 'X-API-Key' header. APIs under this category can view user specific pages and perform restricted actions such as modify projects or upload files.

Staff Authentication - Restricted to staff with the moderator rank. This api will provide the ability to view staff only information such as current reports, the project queue and the file queue. Users authenticating with a staff API key should be able to also perform moderating actions such as edit and upload files to any project (in the future).

In terms of API structure, it should be RESTful and based around a fluid url system. An example url layout is: <https://dev.bukkit.org/api/user/alerts> . No url parameters necessary. Authentication will be via the aforementioned methods, with the potential to expand to OAuth later. Ideally a normal API request will create an entry in the user's ip address history, while an OAuth request will not. Most of the API will revolve around different ways to access certain data types. The data types implemented will be as follows, with examples provided later.

- Project
- File
- Alert
- Conversation
- User
- Category

These are the primary data structures and we be available through the different access filters discussed below.

1. <https://dev.bukkit.org/api/1.0/staff/file-queue> (list of files including any additional file queue info)
2. <https://dev.bukkit.org/api/1.0/staff/project-queue> (list of projects including any additional project queue info)
3. <https://dev.bukkit.org/api/1.0/staff/reports> (list of reports)
4. <https://dev.bukkit.org/api/1.0/user/me> (single user)
5. <https://dev.bukkit.org/api/1.0/user/subscription-alerts> (list of notifications)
6. <https://dev.bukkit.org/api/1.0/user/private-messages> (list of private messages)

7. <https://dev.bukkit.org/api/1.0/user/private-messages/<message spec>> (single private message)
8. <https://dev.bukkit.org/api/1.0/<project-type>> (list of projects)
9. <https://dev.bukkit.org/api/1.0/<project type>/<category>> (list of projects)
10. <https://dev.bukkit.org/api/1.0/<project type>/<project slug>> (single project)
11. <https://dev.bukkit.org/api/1.0/<project type>/<project slug>/files> (list of files)
12. <https://dev.bukkit.org/api/1.0/<project type>/<project slug>/files/<file spec>> (single file)
13. <https://dev.bukkit.org/api/1.0/users/<name>> (single user)

Before jumping into examples of how the API should look I must mention a few things. What is outline below is a guide based on the information I have scraped in the past, or can currently see visible on the BukkitDev pages. I anticipate all fields below are useful and will be included in the final product, however that being said there may be more or less information present in the Curse databases that can be used in this API.

I also hope to maintain an 'official' up to spec binding of this API written in Java as that is of course the primary development language of the Bukkit community.

Data Types:

- Time - All dates and times must be in Unix time.
- Thumbnail - Split into pre: url and post: url (The base url and then the actual image with size in the middle. Refer to:
http://curseassoc.lukegb.com/api/users_have_projects/?user_list=90571076)
- Project - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/projects.json>
- File - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/files.json>
- Alert - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/alerts.json>
- Conversation - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/conversation.json>
- Category - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/category.json>
- User - <https://dl.dropbox.com/u/36903976/Curse/Api-Spec/user.json>

These are the base data types which we hope to get implemented initially. All API methods will revolve around the properties of these data types. While some API's will only contain arrays or single instances of these types, others will be a bit more complex, so I will outline what the expected content for each of them is.

Pagination

Pagination should work as follows:

- There is a default results-per-page limit of:
 - For projects, 20
 - For files and other objects, 30
- When querying, you can specify a start number. This is a count, and not a page number.
- When querying, you can specify a results-per-page limit. There MAY be a limit on how high this may be set, so you can't query for 6102 projects at once and kill the database

and site all in one go.

Example URLs:

- <https://dev.bukkit.org/api/1.0/server-mods/mechanics/>
 - 20 results returned
 - Start from top
 - Default sorting
- <https://dev.bukkit.org/api/1.0/server-mods/mechanics/?start=32>
 - 20 results returned
 - Start from 32th record (0-indexed - *skip* 32 records, start=0 is same as omitting)
 - Default sorting
- https://dev.bukkit.org/api/1.0/server-mods/mechanics/?per_page=30
 - 30 results returned
 - Start from top
 - Default sorting
- <https://dev.bukkit.org/api/1.0/server-mods/mechanics/?sort=created,desc>
 - 20 results returned
 - Start from top
 - Sort by initial creation date, descending

Example result format can be found at <https://gist.github.com/b4670897e7f0b3c1e0f2>

API Usages:

Detailed below in each of the API endpoint usages, but a quick background on each project.

- Fill - The snarky #bukkitdev-staff IRC bot. IRC bot and BukkitDev statistician. Fill provides alerts regarding the current queue sizes, and active reports. He also generates stats on how long files / projects remain in the queue and stats on the current queue.
- BROFIST - Bukkit Rapid Online File Investigation and (something) Tool. Aims to be a self contained web app to allow staff to decompile, inspect and approve files without leaving from their browser. Will also analyse for malicious files.
- plugin.bukkit.org (v2) - Plugin search interface rewrite. Luke's project, a rewrite of the existing plugin search page to use the new official API and provide extra features.

API endpoints:

1. File Queue
 - a. BROFIST - here meaning Bukkit Rapid Online File Investigation and (...something) Tool - file decompilation/analysis engine
 - b. Fill - #bukkitdev-staff IRC bot
2. Project Queue
 - a. BROFIST
 - b. Fill
3. Reports - An array of all open reports

- a. Fill
- 4. The logged in users profile page, following generic user spec
 - a. Bukkit forums account association
 - i. Ideally we'd be able to get this with a combination of two API keys: the target user's, and a moderator's key - this is for stuff like where we have a listing of projects owned by users on the Bukkit forums - no IP address history for the target user would be recorded (because that would defeat the point)
 - b. BROFIST, again
- 5. Subscription Alerts - An array of the last subscription alerts, as seen on <http://dev.bukkit.org/home/>
 - a. Bukkit forums account stuff
- 6. Private Messages - An paginated array of private messages
 - a. Bukkit forums stuff - showing PM's to forum users
- 7. Private Messages - An individual private message
 - a. Bukkit forums stuff
- 8. Projects - All projects in type <x> eg all server-mods
 - a. plugins.bukkit.org v2
 - i. For pbo I also need the ability to filter last-updated since, since I'm going to be feeding these documents into elasticsearch
- 9. Projects - All projects in type <x> and category <y>
 - a. plugins.bukkit.org v2
- 10. Project - A specific project
 - a. plugins.bukkit.org v2
- 11. Files - All files in a specific project
 - a. plugins.bukkit.org v2
- 12. File - A specific file in a project
 - a. plugins.bukkit.org v2
- 13. User - A single user matching <name>
 - a. BROFIST
 - b. pbov2 - plugins.bukkit.org/curseforge search.
 - c. other moderation tools?
 - d. Bukkit forums account stuff
 - i. This needs a list of projects that the user is involved in, as per spec.
 - ii. Minimal embedded project data is fine - project name, project slug, thumbnail and summary

Thanks for reading to the bottom of this document. A couple of final implementation details. As this API will be used cross domain, it is a must that JSONP be supported. A JSONP callback should be specified via the ?callback=.... url parameter.

Once again we thank the Curse employee(s) reading this and hope to get a shiny new API in the near future :)

Future
Oauth