Topics for Hackfest 24

Driving change in the community

Topics

Discussions & Proposals

Community process

Do we have enough people pushing code, should we really be stuck with one RM?

Should we really be supporting as many branches as we do, it's always hard to fill the RMaint roles?

- We need more QA people
- We need more SO people
- Small clear refactorings take months or even years to work through the system.. it discourages any form of code maintenance and modernisation effort... these sorts of bugs should go through in days.

Can we use GitLab?

Hackfest Target

- 2 one-hour sessions to talk through the above and come up with some proposals (MR/PA)

Using content delivery

We should really be starting to take advantage of CDN's both from the standpoint of improved performance and from the perspective of it pushing us to keep our libraries up to date more in line with current security best practices.

Hackfest Target

- Discuss the reasons we're not already doing this and propose some solutions (MB/PA)

Using AI in development

What tools are other Devs in community using right now?

Can we use AI in any of our pipelines to further improve our code and testing

- Copilot
- Codium
- ChatGPT

Hackfest target

- 1 hour discussion resulting in a writeup of tools/practices used. (Pedro/Martin to promote discussion here)

Stricter naming in schema + code

A hold should be a hold and referred to as such, not reserve, not request, not reservation.. Patron vs borrower etc, and many many others. This is confusing, adds unnecessary complexity and are traps for new and veteran developers. Using **to_api_mapping** in REST classes is adding more code to fix a

problem on the wrong end of it, i.e. it should be fixed at the schema level.
If updating the database schema is resisted because of compatibility with existing SQL reports, then we should first think/handle how to update SQL reports (or just notify users on upgrade?). Existing SQL reports in Koha systems should not be considered a good enough reason to prevent us from evolving/updating the database schema.

Hackfest Target

- Discussion - Why oh why do we keep piling pain on top of pain (PA)
- Technical debt is growing
    - It affects developer performance
    - It affects developer onboarding
    - It affects code performance
    - It affects reliability of code
- What's the solution to the reports?
    - Can 'versioned reports' in Mana work better to facilitate fixing of database fields
    - Should we migrate the Wiki reports into Mana and drop the Wiki
    - Does Mana need a better UI for all this.
- What's the solution to notice templates?
    - Can we lean on Mana for this too?

HEA Improvements

HEA is an opt-in community statistics aggregator for Koha usage. It's a little dated now and not all that well used.

It would be nice to give it a refresh, encouraging wider adoption from the wider user base.

- Resync which system preferences we track
- Add ability to 'phone home' to get updates on current installations, versions etc
- Add plugin usage tracking
- Database (MySQL or MariaDB), maybe the major/minor version?

Hackfest Target

Nice idea, but nothing to do right now. It would be nice to use this to start removing underused system preferences. Also, local data vs centralised.

- Maybe a quick discussion, but no real target for improvement here right now. (MR - Dropped)
- Why have this if it's not really being adopted or used?
- Discuss early in hackfest to see if this is worthwhile pursuing enhancements, dropping support or just leaving as is?

Next steps for Vue!

Any new pages from this point forward should be Vue based - there should be a Koha level Vue app that can handle any page assigned to the router. Basically I just don't like template toolkit. Pedro's http client can easily be expanded to handle wider usage in Koha. Unless we have some sort of break-point / hard cut-off date then people will just keep adding more .tt and .inc files because its what they're comfortable with. A Koha level Vue app could also fix the issue of requiring updates to the Apache config files in order to enable new Vue routes. We would just need one redirect rule pointing at the Koha level Vue app and the vue-router would do the rest

We need re-usable vue components that can be used from various places in the UI rather than the current WET copy/paste of components and distinct Apps for each module/area. (resurrect the Page component?)

We need to solve the /config endpoint problem properly rather than build more workarounds on top of the workaround.

Proposal for discussion: The main landing page should include a data structure with the different configurations (and maybe permissions, though those could be baked into a JWT on a cookie). In the event of a 'permission denied' API response, there could be a header returned along, with the new configurations.

https://github.com/PTFS-Europe/koha/tree/vue_monolith_prototype

Hackfest Target

- Discuss the road ahead (MB/TC/JD)
    - How do we generate/inject left menu options?
    - How do we use Vue components in non-vue application pages?

Hackfest Feedback

- Lazyloading Vue
- Attempt to have an example submission for a vue component outside of the App
- Submit the Apache changes bug required to have one Vue router instead of one per 'App'
- We need to solve the configuration/permission challenge
- Trigger session invalidation for a users session when their permissions change

Merge additional fields and custom patron attributes

These are functionally the same thing and should not be implemented and maintained separately, as they are currently. Example:
Bug 32610- Add ability to specify patron attribute as a date
This enhancement would benefit patron attributes but would have to be reimplemented to additional_fields when it's needed there.
Ideally, a single **Koha additional fields (or other naming)** section would benefit all adopting tables, starting with borrowers and all currently additional_fields supporting ones. All future enhancements would benefit all.

Hackfest Target

- Discussion around why this is a sensible approach, what caveats there might be holding us back. (PA/MR/JM)

Task Scheduler

The job queue is a FIFO queue right now; It would be nice to also be able to set delays on tasks to allow for scheduling tasks to run in the future (and maybe even set it as repeating). This would be especially helpful for the live holds queue work to allow for scheduling a recheck for branch availability after hold is found to not have any suitable open branches for assignment during the normal live assignment process.

Hackfest Target

- Discuss next steps and where our priorities lay.
- Discussion near the beginning of Hackfest to see if there's anything we should work on as a group.

Hackfest Output

- No-one is using RabbitMQ (ByWater, PTFS, Biblibre, HKS)
- **Bug 35655 - Make it possible to switch off RabbitMQ without any warns in logs/about page**
- **Bug 35920 - Centralize code from workers**

Tasks

Developer documentation

The community POD renderer is in dire need of a refresh and update. It would also be good to have a central place for onboarding new developers and a place to point plugin developers to when they're asking for information around the modules within Koha.

Hackfest Targets

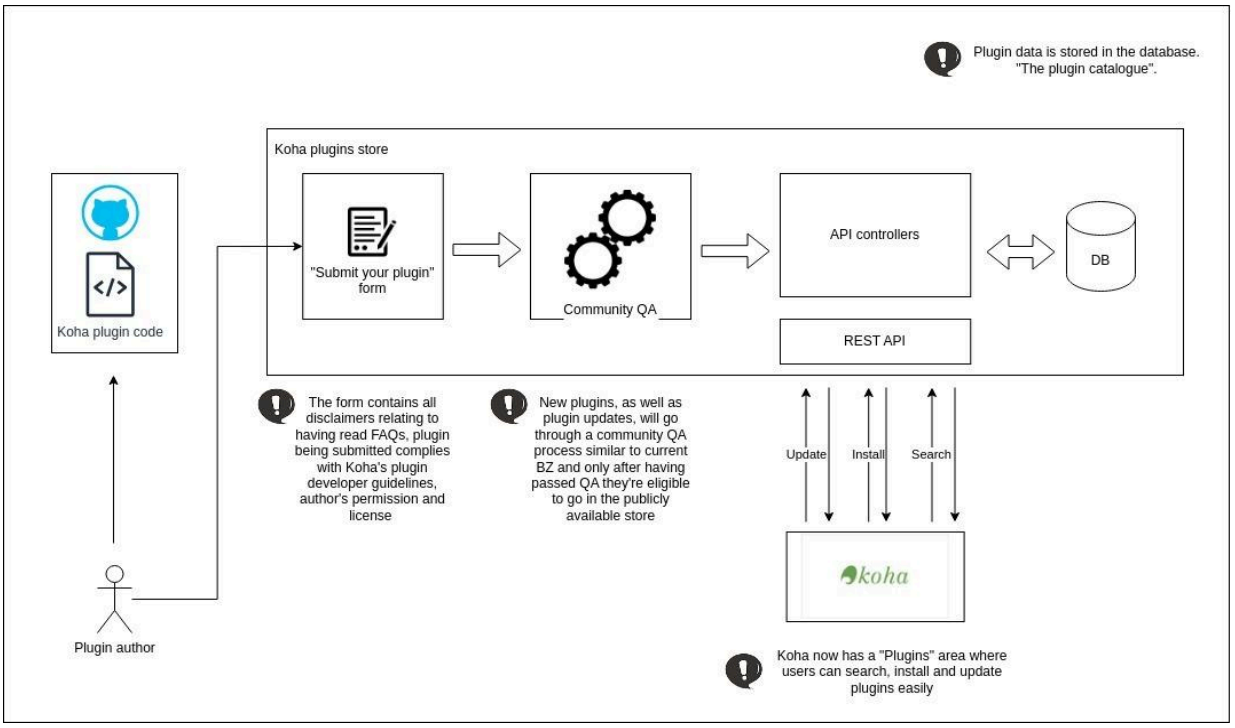
- POD Rendering site should be up and running by the end of Hackfest (MR/TC)
- Plan for improved PODs in Koha
- Discus needs for documentation of our Vue/JS components and how we might publish that documentation too.
- Dev manual/Cookbook for 'dev patterns', stop re-inventing the wheel
- gitBook perhaps
- Are there some existing patterns that are not widely adopted enough yet that we could DRY out and fix missing cases for at Hackfest (removing bad/old cases)

Official plugin store for Koha

Allowed plugins would be visible and searchable through a plugin store. Plugins who want to be part of the store would have to go through a QA process similar to what currently happens in the community.

- Including plugin authenticity (signed/verified/quality assured plugins)

Early draft:



The plugin store could also be used to complement HEA/Mana (or replace, if HEA/Mana adoption is non-existent) as it would store and provide info on downloads/adoption for each plugin.

Hackfest Target

- Solve the plack restart problem
- Consider how plugins can integrate more deeply with core in the following ways:
    - User permissions
    - Exposing themselves as a module
    - Integrating into system preferences management
    - Not putting static content under /api/
- Build a proof of concept for a centralised plugin store as a replacement/alternative to the existing ‘install from github/gitlab’ approach.

Hackfest output

- Need to re-think the koha-plack wrapper to “watch” the plugin directory
- We need eyes on **Bug 31074 - Cached plugin code is used in Koha even after changes to plugin (install/upgrade/uninstall)**
- **We need a process for what makes a plugin recognised by community and what should be highlighted by the plugin store (Translations, Security Audit, Test coverage).**

-

Rethink system preferences

Are they all really needed ? 800+ of them. Could some of them be moved into plugins? Should we better determine what should be a system preference and what shouldn't? Could HEA be used here to determine some system preferences that should be converted to plugins and removed from code? Adding every little niche functionality required by one library from some country to core with a toggle switch is not maintainable, should instead be a plugin only utilised by those who actually need it.

Hackfest Target

Agree a direction for system preferences

- The process for adding system preferences should be refined, and more questions asked - Do we really need this new preference?
- Look into Tomas's bug to split Preferences
    - Bug 26129- Add a new 'configurations' table
- If a whole module can be enabled/disabled, all its accompanying options should not be visible

Hackfest Output

- There is a bad pattern.. add a syspref...
    - then someone asks to make it per library
    - then someone asks to make it per item type
    - then someone asks to make it per patron category
- Tomas's 'configurations' table was designed around this to be much more aligned with circulation rules.
- Inheritance is important.. i.e. global default you can override at various levels
- Some modules deserve their own "Configuration page"
- We want a first-class way of adding preferences at plugin install time (and removing/hiding them at plugin removal time)
- We really do need to work on reducing complexity in the out of box experience.
- Preferences should have dependency trees to allow the grouping and hiding of dependent preferences

Rethink user permissions

User permissions are somewhat of a mess and have been for a long time. It would be helpful to split permissions by the objects/data they affect rather than module and then introduce a clearer roles system to allow permissions to be assigned to roles and then roles assigned to users. This would more clearly align to 'OAuth scopes' and the way the API expects permissions to be defined.

We could really do with simple 'CRUD' level permissions on all our api endpoints, but if we do that with the current permissions system which is a mix of role style permissions and subpermissions at a more granular level we'll end up in a mess.. We should split permissions down to their smallest unit and then define groups of permissions as roles that can be assigned to users. This would also open the door to proper 'scopes' in the OAuth world.

Bug 20813- Revamp user permissions system might be a starting point.. or it might not..

Bug 29509- GET /patrons* routes permissions excessive

- **Context level for CRUD permissions (Create Biblio in the context of Acq but not Catalogeur)**
- **Some permissions don't fit the Object-level permissions context.. they need.**
- **What you can do (CRUD), Where you can do it (From library)**
- **Allow implicit permission to trickle down.**
- **How do we manage scenarios where permissions are "cross module" i.e. needing vendor-related permissions to create/update ERM licenses?**
- **Interested parties (KIT for Testing, Mark? from HKS3 for reviewing code and refining concept, Paul from LMS Cloud)**
- **uses cases to keep in mind**
    - **acquisition: create a record without the general cataloguing module permission**
        - **maybe a dedicated api route in acq route for creating records instead of the general one**
        - **maybe something with embedded objects**
    - **limited items edition: intern cataloguer should only be able to edit item**
    - **self checkout???**
    - **leverage library groups to preserve consortia use cases**
- **have a view of existing roles assigned: to spot a student that had roles due to a summer internship but dont need then anymore**

Hackfest Target

Agree a direction for user permissions

- Plugins should be able to easily add to the avaiiable user permissions system

API Masquerading

Currently, the API's report back as being actioned by the API user. It would be helpful to add masquerading support to the API to allow for the distinction between the 'API Client' and the 'End user' who is using that API Client service (this is particularly important for tools like Aspen).

Hackfest Target

- Bug 30109- Wrong interface in action logs for some staff actions
- OAuth2 - Authorization Code Grant (allowing a 'client' to act on behalf of the 'user')
- Alexander look into this alongside a Koha dev (from the perspective of Aspen)

Improve patron edit form + patron quick add

The patron form is very hard-coded and too loaded. The patron quick add form is too reliant on the main form, making it difficult to be reused in other places of Koha.
Patron quick add should be easily reusable in other parts of Koha. We (developers) should be able to add the **patron quick add** form to the circulation UI, ILL UI, ERM UI, and so on, through plugins or otherwise.

Rewriting the patron form may require deeper thinking on how we handle patron core attributes versus borrower_attribute_types: Borrowers table currently sits on 83 columns, this is clearly a symptom of "one library needs a specific field, it gets added to core". 80 something fields later, here we are.

Hackfest Target

- Add a new "Quick add" using the API as step one
    - Come up with minimum requirements (i.e. what sysprefs are involved)
- Remove the "Quick add" from the normal form
- Migrate the current form to API driven flow above?
- Bug 36179 - "Quick add" patron form should be a modal

Template toolkit notices

It would be good to form a working party to start dealing with the horrible mix of syntaxes in the Notices tool. We really should finish off the migration to template toolkit and remove support for <<>> style templates; To achieve this however there are a few steps that need to take place:

Hackfest target (MR/ByWater)

- Build a proof of concept replacement for the notice editor that supports TT and drops <<>>
- Continue migrating default templates from <<>> to [% %] equivalents
- Submit the bugs to remove support for <<>> syntax

Elasticsearch

I (domm) and/or we (HKS3) would like to work on Elasticsearch:

- Bug 31652 GeoSearch, using Query DSL instead of query string query
- Fix Bugs in the ES-Mappings, update MARC mappings, ..

Hackfest Target

- Have a clear plan on how to use more advanced ES features in Koha

LMS Cloud

- Grab their cash management summary stuff, it looks nicer
- Grab display of parent child relationships in Marc stuff (OPACDetailVolumeDisplay)
- Mobile library support, rooting etc
- RFID Stacks support in staff client, KIT have a plugin version
- Room reservation plugin
- Event management plugin
- ILL
- Notice charges (postal notices for example)

Community communications

- Our tools are old
    - Katrin/Brendan are going to compile our 'needs for a bug tracking tool' - 3 weeks from Hackfest
    - We need to do a similar piece of work to list our needs for overall communication methods (Instant messaging, Larger discussions - Email/Forum/Q&A, Task tracking, Code review/Bug feedback, Documentation)
    - Suggestions
        - Mattermost - AA/BG investigating ($250 USD for us)
        - Discourse - PD is going to continue
        - SMTP server investigation - David HKS3

Sandboxes Data

- Goes into misc4dev
- Data we want
    - Checkouts between 'today' and X weeks ago (including some overdue, some due) - Martin/Jonathan
    - Subscriptions - Mikaela
    - Holds in various states
    - Acquisitions data
        - Funds
        - Orders
        - Vendors

        - Baskets
        - Standing orders
    - ERM data - Matt/Jonathan
    - Cash registers - Martin
    - Patron restrictions -
    - Fines - Martin
    - Courses - Mikaela
    - Stock rotation rota - Martin
    - Pre-linked biblio records with volume links, serial links and analytic links.. title and control number type linked. - Mikaela
    - SMS Providers
    -
- Use Data::Faker in Test::Builder to create more meaningful test data