Making Pull Requests

Anyone is welcome to and encouraged to propose changes to the Aspen code. Aspen gets stronger the more folks contribute! This page has guidelines on the best practices for submitting code for inclusion in Aspen.

If you have a development you'd like to submit a code change for, it is a good idea to check in with the Aspen Developer Community. Checking with the community will ensure that if someone else is working on a similar idea or project, you can discuss how the functionality will work to make sure we don't have redundant or conflicting development happening at the same time. You will also be able to get tips or suggestions on how to architect your development before you get started.

Where to get help and advice

You can check in with the community by attending an Aspen Developer Community meeting or by asking in the #developers channel in the Aspen-Discovery slack space. Reach out to jordan.fields@groveforlibraries.com or your support vendor for an invite to the slack space.

Steps to Making the Pull Request

These steps are provided as a broad suggestion on how to work with pull requests and keep things clear.

1. First time (or first-time-in-a-long-time) only: "Fork" https://github.com/Aspen-Discovery/aspen-discovery to create your own copy of the repository.
2. Create a new branch based on the release in which your code should debut. Generally this will be the default branch (*see note below) and give it a descriptive name (e.g., "Use natural sort for selecting items when placing a hold"). If you are working ahead and the branch for the next release is not available yet, please ask in the developers channel for the new branch to be created. Or alternatively if you are working on a bug fix for an older version and a point release is not available, please ask for that branch to be created. If you are unsure which release to target, please ask the developer community.
3. If your change is based on a public ticket or issue, include the issue number in the name of the branch to make review easier, e.g., *DIS-123 Implement a thing in Aspen*.
4. Make your changes in this new branch. In addition to your new features and bug fixes, you must also include descriptive release notes.
5. If your changes have taken longer than intended and the release you originally intended your code for is no longer open for changes, rebase your code to the new desired release. Test all of your changes locally to ensure your use cases are working properly. Consider the needs of consortial setups, and both new and existing installations. The

earlier code is tested the better! If you are unsure about how your change might affect other areas of the code, please ask the community for additional brainstorming.
6. Commit and push to your repository in GitHub.
7. Go to GitHub and start a new pull request through the web interface.
8. Enter a description about what has been done and why. It's often useful to provide a short use-case when it's not obvious from the change. Testing notes or a link to a more comprehensive description should also be added to the pull request description for non-trivial pull requests.
9. Review the changes on this screen once more so that you can catch any inadvertent changes. Also make sure that any new files were properly added.
10. Submit the pull request.

*Note on the default branch: Aspen has a regular monthly release schedule. The github.com default branch for the project is set to the current development branch. Learn more about the release schedule and release naming convention here.

What Happens to a Pull Request

When a pull request is submitted, some automated tests are run against the submitted code. As of November 2024, these automated tests are in active development; please feel free to discuss the details of failed tests for your pull requests as well as the need to modify, add, or delete tests in the Aspen slack #developers channel. Eventually, these tests should ensure the submission meets code style requirements. They might eventually include regression testing to ensure certain functions are not lost or disrupted by new code, but they currently do not.

The code will also be reviewed by Mark Noble or another experienced developer who has contributed substantial development in multiple areas of Aspen in 10 of the past 12 months. Code is reviewed as quickly as possible in an order that is most efficient for the reviewers. Longer, more complex code requiring more review time could take longer than smaller, more trivial changes.

This review considers several aspects of the change, including the following:

- General architecture, including the impact of the change on future developments
- Code quality, including comments
- Security
- Backward-compatibility
- Any effect the change might have on current users, with special emphasis on the following:
    - compatibility with consortia architecture
    - compatibility with Aspen themes and styling
    - the ability to turn the feature off for libraries that don’t want it
- Any new requirements or dependencies

- Performance implications
- Accessibility

You may receive a request for further information or clarification, suggestions for refinement, and/or other constructive comments that aim to maintain and improve quality. Reviewers take the review process very seriously as a means of both helping to onboard and upskill developers as well as ensuring Aspen code is stable and secure, and they frequently spend a significant amount of time on their comments and reviews. Please be respectful of the time reviewers donate to reviewing your pull request and make sure that you are addressing all comments before submitting your code for review again. Reviewers, nonetheless, are fallible, so feel free to question the comments if you don't find them reasonable. However, if you feel like you're getting shut down or locked out, please do your part to try something different or explain the problem you're trying to solve in a different way.

Once you have completed the changes requested in a review within your local branch, please re-submit your pull request by submitting an additional commit to your forked branch. If you know your work will consist of more than one commit, you should convert the pull request to a draft pull request. On the pull request page, in the Conversation tab, click on the "Still in progress? Convert to draft" link located on the right side of the page in the Reviewers section. When you have completed the work, click the "Ready for Review" button in the pull request page Conversation tab in the merge bubble toward the bottom of the conversation.

When everything checks out and all requirements are fulfilled, the pull request will get merged. After it has been merged, you can delete your own branch. It's a good idea to clean up obsolete branches regularly.

After the code is merged, the code is available for all libraries to test and review at any time between code merger and the full release to production.

Note that an accepted and merged pull request may need to be revisited before the full release to production servers on the second Wednesday following the first Thursday of each month. In general, the original developer of a pull request is expected to make any needed updates found during the testing process. If you are unable to make changes during the testing period, please let Mark Noble or another experienced developer know so they can help with any needed fixes.

This document was prepared by James Staub, Jordan Fields, and Mark Noble. It was shared for comments to all Aspen developers with more than 500 commits to the main Aspen repository including Kirstien Kroeger and Kodi Lein.