

2022-06-30 DSpace 7 Working Group Meeting

- [Date](#)
- [Agenda](#)
- [Attendees](#)
- [Current Work](#)
 - [Project Board](#)
 - [New Feature development process for 7.3](#)
 - [Issue Triage process for 7.3](#)
- [Notes](#)

Date


30 Jun 2022 from [14:00-15:00 UTC](#)

Location: <https://lyrasis.zoom.us/my/dspace> (Meeting ID: 502 527 3040). **Passcode:** dspace

- More connection options available at [DSpace Meeting Room](#)

Tim is on holiday from July 2-10, returning on Monday, July 11. Therefore, next week's meeting on July 7 is cancelled.

7.4 Release Plan

 Release Schedule (*rough timeline*):

- **TBD (PR Creation Deadline):** All new feature (or larger) PRs should be created by this date. (Smaller bug fixes are welcome anytime)
- **TBD (Initial Review/Test Deadline):** All code reviewers or testers should submit their feedback by this date. Code reviews must be constructive in nature, with resolution suggestions. Any code reviews submitted AFTER this date will be considered *non-blocking* reviews. This means feedback received after Jan 20 is *optional* to address (unless the team or PR developer decides it is required).
- **TBD (Initial Feedback/Revisions of PRs due):** PRs should be updated based on initial review/test.
- **TBD (Secondary Review/Test Deadline):** Re-review of all updated PRs.
- **Friday, September 23 (PR Merge Deadline):** All new feature PRs should be merged by this date. (Bug fixes can still get in, as long as they are small or important)
- **Week of September 26:** Internal / Early release goal. If possible, we'd like to release 7.3 this week.
- **Monday, October 3:** Public Release Deadline. 7.4 will be announced/released by this date.

Ongoing/Weekly tasks:

- Tackle/Claim issues on [7.4 board](#) (starting with "high priority")
- Review/test all PRs assigned to you for review/testing: <https://github.com/pulls/review-requested> (*Prioritize reviews of "high priority" PRs first*)

Agenda

- (55 mins) General Discussion Topics
 1. **Main topic: 7.4 Release (due Oct 3): Maintenance release only? Or prioritize maintenance first but also allow new features?** Which approach do we want to take?
 - a. Development priorities for 7.4, given the [v5 & 6 EOL announcement](#). These priorities have been approved by Steering
 - i. **HIGHEST PRIORITY:** *Stability fixes and major bug fixes.* Goal is to immediately fix any major issues we are aware of (or hear about) that might impact production sites.
 1. We will need to prioritize bug fixes based on severity, likely number of impacted users, whether workarounds exist, etc.
 - ii. **HIGH PRIORITY:** Porting of any missing 5.x/6.x features to 7.x from [tier listing](#)
 1. Several steering members noted that porting additional Admin Tools would be welcome, especially batch import/export tools.
 - iii. **LOWER PRIORITY:** Any other new features (which were not in 5.x/6.x)
 - b. During 7.4 planning, we may want to set earlier deadlines for large features. Existing model works best for small features / bug fixes.
 - i. Large features should be broken down into stages / steps. Schedule an earlier PR deadline(s) for those steps. Also schedule review deadlines for those steps
 - ii. Goal is to ensure Large features being implementation earlier & get feedback earlier. If they can be broken down into stages/steps, then the final PR & review will be much easier.
 2. **Maintenance tasks we feel should be included in 7.4** (*Below, please include links to tickets or brief descriptions of issues you feel need to be prioritized highly*)
 - a. Curation Tasks are not very usable from the Admin UI: their results are only sent to dspace.log <https://github.com/DSpace/dspace-angular/issues/1322> and they cannot restore deleted objects <https://github.com/DSpace/DSpace/issues/7956> and cannot run on items <https://github.com/DSpace/DSpace/issues/8384>
 - b. Resource Policy edit form is confusing & have usability issues. See for example <https://github.com/DSpace/dspace-angular/issues/1704> and <https://github.com/DSpace/dspace-angular/issues/644> (which is not policy specific but is reproducible when editing a policy)
 - c. Improve password requirements/management. See for example these tickets https://github.com/orgs/DSpace/projects/18?card_filter_query=label%3A%22authentication%3A+password%22
 - d. Provide a way to manage/cleanup Processes, e.g. <https://github.com/DSpace/DSpace/issues/7974> and <https://github.com/DSpace/dspace-angular/issues/1343> and <https://github.com/DSpace/dspace-angular/issues/1660>
 - e. Issues with file download from submission/workflow, see <https://github.com/DSpace/dspace-angular/issues/1644> and <https://github.com/DSpace/DSpace/issues/8378>

- f. General caching issues, e.g. <https://github.com/DSpace/dspace-angular/issues/644> and <https://github.com/DSpace/dspace-angular/issues/742> and https://github.com/orgs/DSpace/projects/18?card_filter_query=label%3A%22performance%2F+caching%22
 - g. Statistics display issues, e.g. https://github.com/orgs/DSpace/projects/18?card_filter_query=label%3A%22component%3A+statistics%22
 - h. (Possibly) Restrict system managed metadata fields from editing: <https://github.com/DSpace/DSpace/issues/8299>
3. **Feature PRs which missed 7.3**
- a. Recent Submissions Listing on Homepage (Tier 2, donated, smaller PR): <https://github.com/DSpace/dspace-angular/pull/1632>
 - b. SAF (Simple Archive Format) Import/Export from UI (Tier 3, highly-prioritized by Steering, medium PR): <https://github.com/DSpace/DSpace/pull/8318>
 - c. OpenAIRE Graph "quality assurance" (notification broker) feature (Tier 4, donated, large PRs but has had several reviews): <https://github.com/DSpace/dspace-angular/pull/1562> and <https://github.com/DSpace/DSpace/pull/8184>
 - d. OpenAIRE "suggestions" (publication claim) feature (Tier 4, donated, large PRs - no reviews yet): <https://github.com/DSpace/dspace-angular/pull/1638> and <https://github.com/DSpace/DSpace/pull/8280>
 - e. Flexible additional metadata for MyDSpace page (Donated, small PR): <https://github.com/DSpace/dspace-angular/pull/1621>
 - f. Amazon S3 support improvements (arguably not a new feature): <https://github.com/DSpace/DSpace/pull/8356>
 - g. Port + Refactor event consumer curation code (Queueing Curation Tasks) (donated): <https://github.com/DSpace/DSpace/pull/8151>
- (5 mins) Planning for next week
 - Review the [Backlog Board](#) - Are there any tickets here stuck in the "Triage" column? We'd like to keep this column as small as possible.
 - Review the [7.4 Project Board](#) - Assign tickets to developers & assign PRs to reviewers.
 - Paid (by DSpace project) developers must keep in mind priority. If new "high" or "medium" priority tickets come in, developers should move effort off of "low" priority tasks.
 - Volunteer developers are allowed to work on tickets regardless of priority, but ideally will review code in priority order.

Attendees

- Art Lowel (Atmire)
- Andrea Bollini (4Science)
- Tim Donohue
- Lieven Droogmans
- Giuseppe Digilio (4Science)
- Ben Bosman
- Paulo Graça
- Mark H. Wood
- Pascal-Nicolas Becker

Current Work

Project Board

DSpace 7.4 Project Board: <https://github.com/orgs/DSpace/projects/18>

To quickly find PRs assigned to you for review, visit <https://github.com/pulls/review-requested> (This is also available in the GitHub header under "Pull Requests Review Requests")

New Feature development process for 7.3

- **For brand new UI features**, at a minimum, the UI ticket should contain a description of how the feature will be implemented
 - If the UI feature involves entirely new User Interface interactions or components, we recommend *mockups or links to examples elsewhere on the web*. (If it's useful, you can create a Wiki page and use the [Balsamiq wireframes](#) plugin in our wiki)
 - Feature design should be made publicly known (i.e. in a meeting) to other Developers. **Comments/suggestions must be accepted for TWO WEEKS, or until consensus is achieved (whichever comes first)**. After that, silence is assumed to be consent to move forward with development as designed. (The team may decide to extend this two week deadline on a case by case basis, but only before the two week period has passed. After two weeks, the design will move forward as-is.)
 - This does mean that if a UI feature is later found to have design/usability flaws, those flaws will need to be noted in a bug ticket (to ensure we don't repeat them in other features) and fixed in follow-up work.
- **For brand new REST features (i.e. new endpoints or major changes to endpoints)**, at a minimum we need a REST Contract prior to development.
 - REST Contract should be made publicly known (i.e. in a meeting) to other Developers. **Comments/suggestions must be accepted for TWO WEEKS, or until consensus is achieved (whichever comes first)**. After that, silence is assumed to be consent to move forward with development. (The team may decide to extend this two week deadline on a case by case basis, but only before the two week period has passed. After two weeks, the design will move forward as-is.)
 - This does mean that some REST features may need future improvement if the initial design is found to later have RESTful design flaws. Such flaws will need to be noted in a bug ticket (to ensure we don't repeat them in other features) and fixed in follow-up work.
 - **REST API Backwards Compatibility support**
 - During 7.x development, we REQUIRE backwards compatibility in the REST API layer between any sequential 7.x releases. This means that the 7.1 REST API must be backwards compatible with 7.0, and 7.2 must be compatible with 7.1, etc.
 - However, deprecation of endpoints is allowed, and multi-step 7.x releases may involve breaking changes (but those breaking changes must be deprecated first & documented in Release Notes). This means that it's allowable for the 7.2 release to have changes which are incompatible with the 7.0 release, provided they were first deprecated in 7.1. Similarly, 7.3 might have breaking changes from either 7.1 or 7.0, provided they were deprecated first.
 - After 7.x development, no breaking changes are allowed in minor releases. They can only appear in major releases (e.g. 7.x8.0 or 8.x9.0 may include breaking changes).

Issue Triage process for 7.3

- **Overview of our Triage process:**

1. *Initial Analysis:* [Tim Donohue](#) will do a quick analysis of all issue tickets coming into our [Backlog Board](#) (this is where newly reported issues will automatically appear).
2. *Prioritization/Assignment:* If the ticket should be considered for this release, [Tim Donohue](#) will categorize/label it (high/medium/low priority) and immediately assign to a developer to further analysis. Assignment will be based on who worked on that feature in the past.
 - a. "high priority" label = A feature is badly broken or missing/not working. These tickets must be implemented first, as ideally they *should be resolved* in the next release. (Keep in mind however that priorities may change as the release date approaches. So, it is possible that a "high priority" ticket may be rescheduled if it is a new feature that cannot fit into release timelines.)
 - b. "medium priority" label = A feature is difficult to use, but mostly works.. These tickets *might* be resolved prior to the next release (but the release will not be delayed to fix these issues).
 - c. "low priority" label = A feature has usability issues or other smaller inconveniences or a non-required feature is not working as expected. These tickets are simply "nice to have" in the next release. We'll attempt to fix them as time allows, but no guarantees are made.
3. *Detailed Analysis:* Developers should immediately analyze assigned tickets and respond back within 1-2 days. The developer is expected to respond to [Tim Donohue](#) with the following:
 - a. Is the bug reproducible? (If the developer did not understand the bug report they may respond saying they need more information to proceed.)
 - b. Does the developer agree with the initial prioritization (high/medium/low), or do they recommend another priority?
 - c. Does the bug appear to be on the frontend/UI or backend/REST API?
 - d. Does the developer have an idea of how difficult it would be to fix? Either a rough estimate, or feel free to create an immediate PR (if the bug is tiny & you have time to do so).
 - e. Are you (or your team) interested in being assigned this work?
4. *Final Analysis:* [Tim Donohue](#) will look at the feedback from the developer, fix ticket labels & move it to the appropriate work Board. If it is moved to the [Project Board](#), then the ticket may be immediately assigned back to the developer (if they expressed an interest) to begin working on it.
 - a. If the ticket needs more info, [Tim Donohue](#) will send it back to the reporter and/or attempt to reproduce the bug himself. Once more info is provided, it may be sent back to the developer for a new "Detailed Analysis".

Notes

- Discussed release plan for 7.4. Decided together that it **MUST concentrate heavily on maintenance fixes**.
 - However we want to include donations
 - We also want to do our best to include PRs which missed 7.3 (see list in agenda)
- Donations: How do we include?
 - Concern about larger donates (e.g. Notify work, OpenAIRE)
 - **Set a donation notification deadline** – that way we can discuss the donation ****before**** it is added. Very large donations may need to be pushed to 7.5, if we cannot find reviewers
- Who are the reviewers these days? Mostly Atmire, 4Science & Tim. A few others help but our review team is very small
- Need clear messaging on 7.4. Need to make it clear core team is primarily doing maintenance, but we still welcome donations (of code or reviewers). Need to make it clear we also encourage others to review
- Bugs in system
 - Too many "high priority" bugs these days. Seems like a default tag. Need to reprioritize
- Discussions of maintenance tasks
 - Lots of feature have known minor issues or cleanup needed (e.g. ORCID has parts flagged as "beta", etc). Do we work on this?
 - Also many small, but annoying bugs popping up in v7 (reported by users)
 - Recommend concentrating more on **stability and annoying bugs**. While "cleanup" of existing features is nice, we need to concentrate 7.4 on fixing things that have the most impact (e.g. a bug that impacts all users is more important than one that only impacts some – so, it would be more important to fix a bug in authentication than to cleanup ORCID)
- LARGER PRs
 - Need individual attention/discussion. Need to assign reviewers & set deadlines specific to those PRs. Cannot be treated the same as all other PRs.