2022-06-09 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

09 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

7.3 Release Plan

Release Schedule *(tentative)*:

- *Thursday, Apr 28 (PR Creation Deadline):* All new feature (or larger) PRs should be created by this date. (Smaller bug fixes are welcome anytime)
- *Thursday, May 19 (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).
- *(NEW) Thursday, May 26 (Initial Feedback/Revisions of PRs due): PRs should be updated based on initial review/test.*
- ***(NEW) Thursday, June 9 (Secondary Review/Test Deadline): Re-review of all updated PRs.***
- *(REVISED) Friday, June 17 (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)
- *(REVISED) Week of June 20* Internal / Early release goal. If possible, we'd like to release 7.3 in late May or first week of June.
- *(REVISED) Monday, June 27:* Public Release Deadline. 7.3 *must be* announced/released by this date.

Ongoing/Weekly tasks:

- Tackle/Claim issues on 7.3 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

- (30 mins) General Discussion Topics
    1. **TODAY is the deadline for secondary reviews.** (However, I fully realize that some reviews are not possible & likely to miss this deadline... especially for interdependent PRs, like ORCID)
    2. Review 7.3 Board & status of all open tickets.
    3. (Notes for post-7.3) 7.4 Planning discussion
        a. Development priorities for 7.4, given the v5 & 6 EOL announcement.
            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.
            ii. HIGH PRIORITY: Porting of any missing 5.x/6.x features to 7.x from tier listing
            iii. LOWER PRIORITY: Any other new features (which were not in 5.x/6.x)
        b. Based on feedback, we will likely want to rethink our planning strategies for the 7.4 release. In 7.3, we've noticed the same pattern of too few reviews prior to the Review Deadline and too many PRs getting created at the last minute (just before PR creation deadline). Some brainstorms follow
        c. 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.
- (30 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 7.3 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 7.3 Project Board: https://github.com/orgs/DSpace/projects/16

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