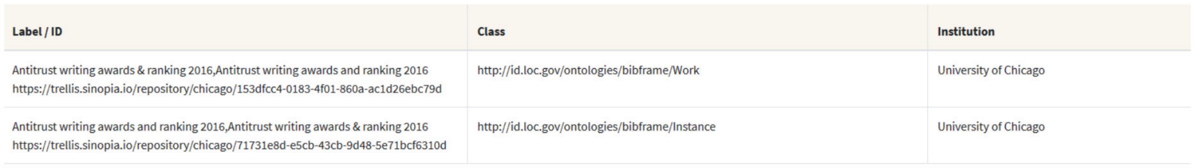
I should get a Work and an Instance back (since all the data is indexed in a keyword sense)...



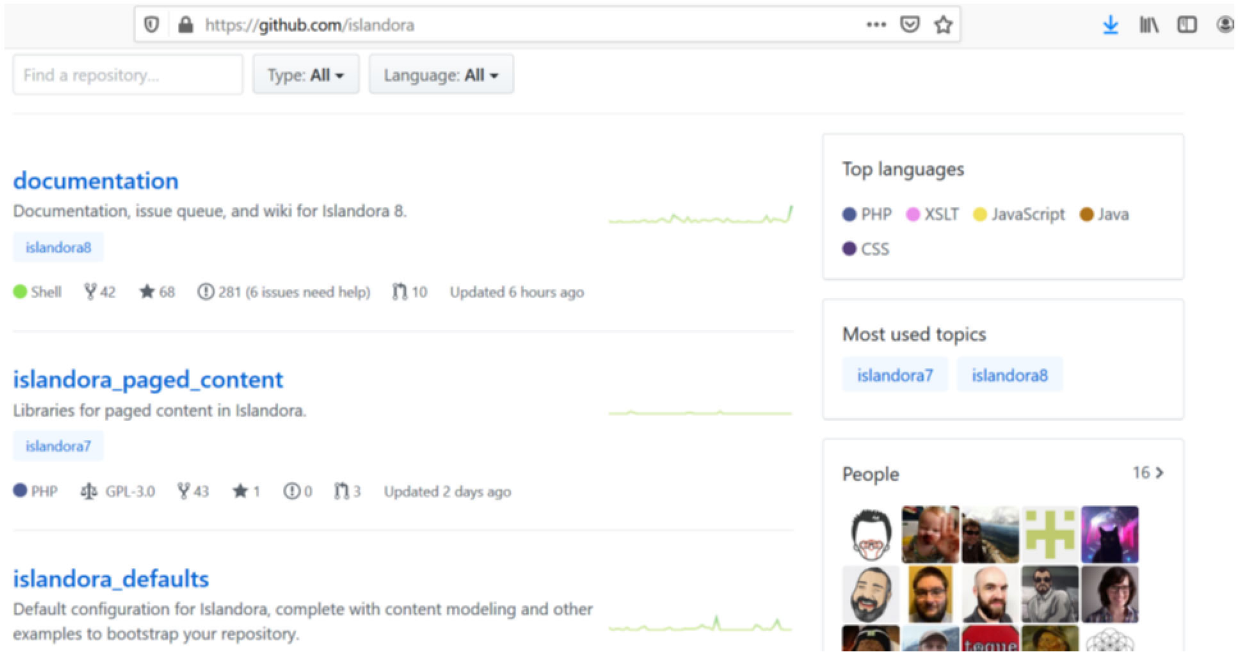
Label / ID	Class	Institution
Antitrust writing awards & ranking 2016,Antitrust writing awards and ranking 2016 https://trellis.sinopia.io/repository/chicago/153dfcc4-0183-4f01-860a-ac1d26ebc79d	http://id.loc.gov/ontologies/bibframe/Work	University of Chicago
Antitrust writing awards and ranking 2016,Antitrust writing awards & ranking 2016 https://trellis.sinopia.io/repository/chicago/71731e8d-e5cb-43cb-9d48-5e71bcf6310d	http://id.loc.gov/ontologies/bibframe/Instance	University of Chicago

And I do!

Navigating GitHub for meaningful communication about software platforms (with no intention of learning git or coding) – Part 2

Wilhelmina Randtke
Florida Academic Library Services Cooperative

Not-programmer things to do on GitHub



https://github.com/islandora

Find a repository... Type: All Language: All

documentation
Documentation, issue queue, and wiki for Islandora 8.
islandora8
Shell 42 68 281 (6 issues need help) 10 Updated 6 hours ago

islandora_paged_content
Libraries for paged content in Islandora.
islandora7
PHP GPL-3.0 43 1 0 3 Updated 2 days ago

islandora_defaults
Default configuration for Islandora, complete with content modeling and other examples to bootstrap your repository.

Top languages
PHP XSLT JavaScript Java CSS

Most used topics
islandora7 islandora8

People 16

Figure 1, above. Screenshot of the GitHub for the Islandora Foundation, showing multiple repositories. In the screenshot, the following repositories are visible “documentation:” “islandora_paged_content” and “islandora_defaults.” Scrolling down would reveal a total of 66 repositories. The repository “documentation” is instructions on using the software. Other repositories are pieces of code, each of which does something specific (for example, show a series of images in sequenced order to display all pages of a book or allow a menu in the web interface for configuring options when a search is run on the site). The key thing is that the authoritative version of what is conceptually one single software platform and one single documentation manual is spread across numerous repositories on GitHub. And supporting libraries, which are necessary to run the software platform, are not represented on the Islandora Foundation GitHub because the supporting libraries would be used in an out-of-the-box way when the platform is installed.

Stepping back: Remember not all activity is on GitHub

One key thing to remember is that an absence of activity on GitHub does not mean an absence of activity. Closed source software will likely have no public code or will have limited public code for the purpose of supporting interaction with application programming interfaces (APIs). Any organization managing or writing software is extremely likely to have some communication tools which are only available to members of the organization and so won't be a public GitHub. Having a public face involves a few extra steps, including checking over what is made publicly available for passwords, and is extra work. So, an absence of activity on GitHub for a person, organization, or software platform does not mean an absence of activity. Think of checking GitHub as similar to checking a single listserv archive when you are looking for information about a specific software platform.

Also, the amount of public activity from a user account on GitHub does not indicate how active the person is in coding. An absence of activity from a user account does not tell you anything.

Non-programmer activity: Find examples of projects using a software platform

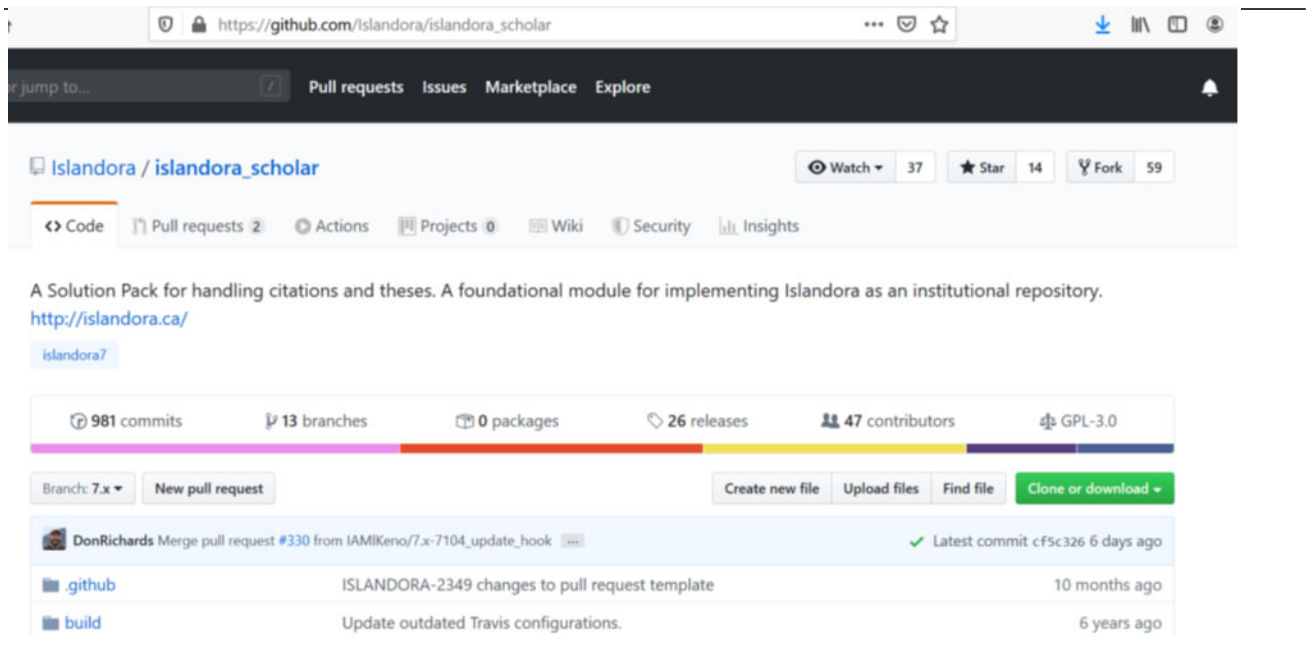
A common need for libraries is to do exploration of several possible software solutions for a need and determine which is best. One way of doing this is to identify who else is using a software platform and contact that organization for information about their experiences. Activity on GitHub is done through accounts which can belong to either an organization or a person. It's not necessary to know git or to understand GitHub well in order to be able to use GitHub to locate people and organizations using a software platform.

In git, one of the ways you can interact with code is to "fork" the code. At heart, a fork is a copy of a codebase that you keep. You can modify it, then you can also examine your own "commits" (i.e. saves) and quickly pull up and review any commits (i.e. saves) made to the source you forked (i.e. copied) from. So, you can easily combine your changes with any changes to the source as that original source gets updated over time. Doing this involves a deep dive, in order to understand what changes to make and to make them. So, anyone who has forked a repository and is regularly saving changes to the fork is likely taking a deep dive into that software platform and engaged. Those are the big installs and power users, likely the people who know where good examples of a software platform can be found and who is using the platform. These are people who are likely to know answers to questions you might have.

In GitHub, once you have located the source for a software platform, you can see forks of each repository.

Each individual repository can be forked. An organizational account cannot be forked. Going back to Figure 1, in order to comprehensively see fork activity, you would have to click into all 66 repositories on the Islandora Foundation GitHub account and view forks from there. Approach this like citation trails in research, where going back over things or comprehensively following citation trails will lead you to the same handful of most important papers. The repositories displayed near the top of the list are the most recently updated. Each repository should have both a descriptive name and when you click into it should display a neatly written README file (should meaning an aspirational goal or best practice and not necessarily the norm). A good approach is to think about what about the software is important to you, for example, zooming in on photos, displaying citations in Blue Book format, or any other need you have. Then, skim for repositories that seem as if they might be related and click on the name of a few repositories to view them. Once you are viewing a single repository, you will see links in the top right of the repository's page for “Watch,” “Star,” and “Fork.” Each has a number next to it, which is accounts that have performed the action. Clicking the number pulls up the list of GitHub accounts which have performed that action.

Figure 2, next page. Homepage of the Islandora Scholar institutional repository portions of the software platform showing links in the top right for “Watch,” “Star,” and “Fork.” Clicking on the list of forks gives a list of 59 user accounts, each of which has copied the code to their account and may or may not have made customizations or shared changes back. Of those 59 accounts, a quick skim shows a handful of accounts with acronyms of large universities. These are institutions which have likely assessed the software in-depth or which currently use the software. If you are considering software to meet a need at your libraries, it may be useful to contact them for advice and pointers.



https://github.com/Islandora/islandora_scholar

jump to… Pull requests Issues Marketplace Explore

Islandora / islandora_scholar

Watch 37 Star 14 Fork 59

Code Pull requests 2 Actions Projects 0 Wiki Security Insights

A Solution Pack for handling citations and theses. A foundational module for implementing Islandora as an institutional repository.
http://islandora.ca/

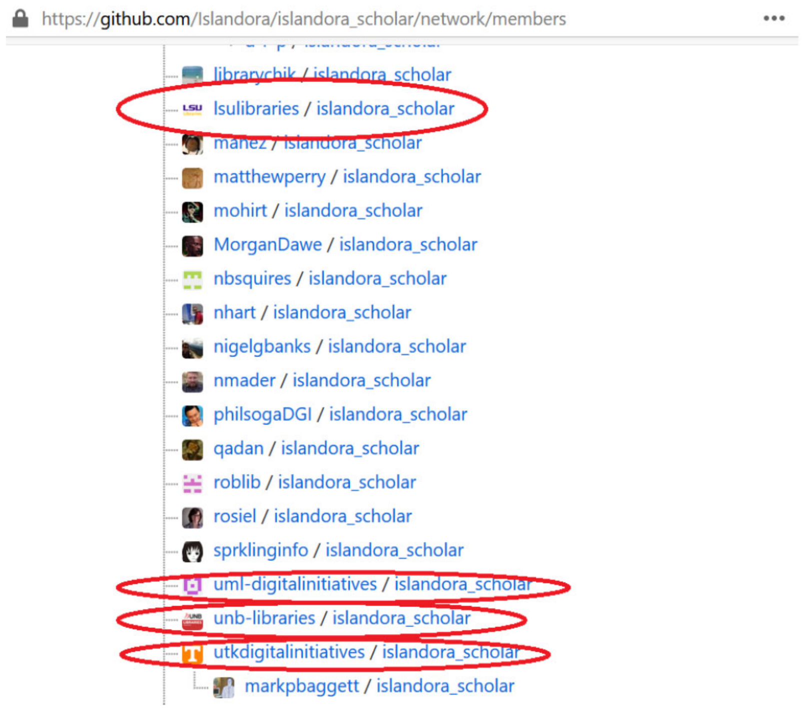
islandora7

981 commits 13 branches 0 packages 26 releases 47 contributors GPL-3.0

Branch: 7.x New pull request Create new file Upload files Find file Clone or download

DonRichards Merge pull request #330 from IAMlKeno/7.x-7104_update_hook … Latest commit cf5c326 6 days ago

.github	ISLANDORA-2349 changes to pull request template	10 months ago
build	Update outdated Travis configurations.	6 years ago



https://github.com/Islandora/islandora_scholar/network/members

librarychik / islandora_scholar
lsulibraries / islandora_scholar
manez / islandora_scholar
matthewperry / islandora_scholar
mohirt / islandora_scholar
MorganDawe / islandora_scholar
nbsquires / islandora_scholar
nhart / islandora_scholar
nigelgbanks / islandora_scholar
nmader / islandora_scholar
philsogaDGI / islandora_scholar
qadan / islandora_scholar
roblib / islandora_scholar
rosiel / islandora_scholar
sprklinginfo / islandora_scholar
uml-digitalinitiatives / islandora_scholar
unb-libraries / islandora_scholar
utkdigitalinitiatives / islandora_scholar
markpbaggett / islandora_scholar

Figure 3, above. A list of accounts that have forked the repository shown in Figure 2. There are numerous accounts. Skimming reveals some institutional accounts, which are circled in red in this screenshot. Those institutions have at some point interacted with this code, and so it's possible to do online searches or contact the institution to find examples of the software platform in action or to talk with someone who might be able to share experiences of running the software. While these are likely to be the power users and will slant towards more technically resourced institutions, the people there may be familiar with who else is using the software and may be able to refer to other users with different levels of resourcing.

<u>Non-programmer activity: Find specific people who have trusted access to the code</u>

One of the responsibilities of my current job is to locate bug fixes, and if they look good, then walk things through in my organization. I am with a large consortium, which runs multiple open source software platforms for digital publishing but which only contributes code and only takes a deep dive on a small fraction of those projects. This can sound well-resourced and technical, but at the same time, fixes generally need review by talented employees with the skill level to understand a change and approval from employees who would be affected in case anything goes wrong. Like in many libraries, I am nontechnical and working with technical employees in other departments. IT is housed separately from the library side of things. If a desired change is a sanity check, then that's a much easier ask than a deep dive is. If I can identify what is and isn't a sanity check, that helps to move some changes forward. For software that the consortium runs but does not contribute code to, often it's important to connect with an expert in the open source community in order to be able to trust a bug fix before implementing it. To do this, I can use GitHub to identify core developers on a software project and people who are able to change the authoritative versions of software made publicly available on GitHub.

What this is useful for is when I've asked a question on a listserv or a support forum and gotten some possible answers. The closer the person giving the advice is to the authoritative version of the software, the more likely that the recommended fix is a sanity check rather than a close look for technical people inside my organization.

To see user accounts on an institutional GitHub account, look along the right-hand side of the organizational GitHub page, and look under the heading labeled “People.” You can see this in Figure 1. Mousing over each person's profile image shows the username. Anyone added to the organizational GitHub account has trusted access to at least one repository, and advice on a listserv or support forum coming from that person is likely to be more authoritative. That may make the difference between being able to resource a review at all or not or may play into what needs to be done to review the recommended change (i.e. how many hours of a person's time to ask for).

<u>Non-programmer activity: Report bugs and "me too" on bugs that are an issue for your organization.</u>

In software development, bug reports and feature requests are generally tracked in some kind of ticketing system. Two common ticketing systems which allow members of the public to participate (with approval) are GitHub issues and the Jira ticketing system. Some software communities will use both.

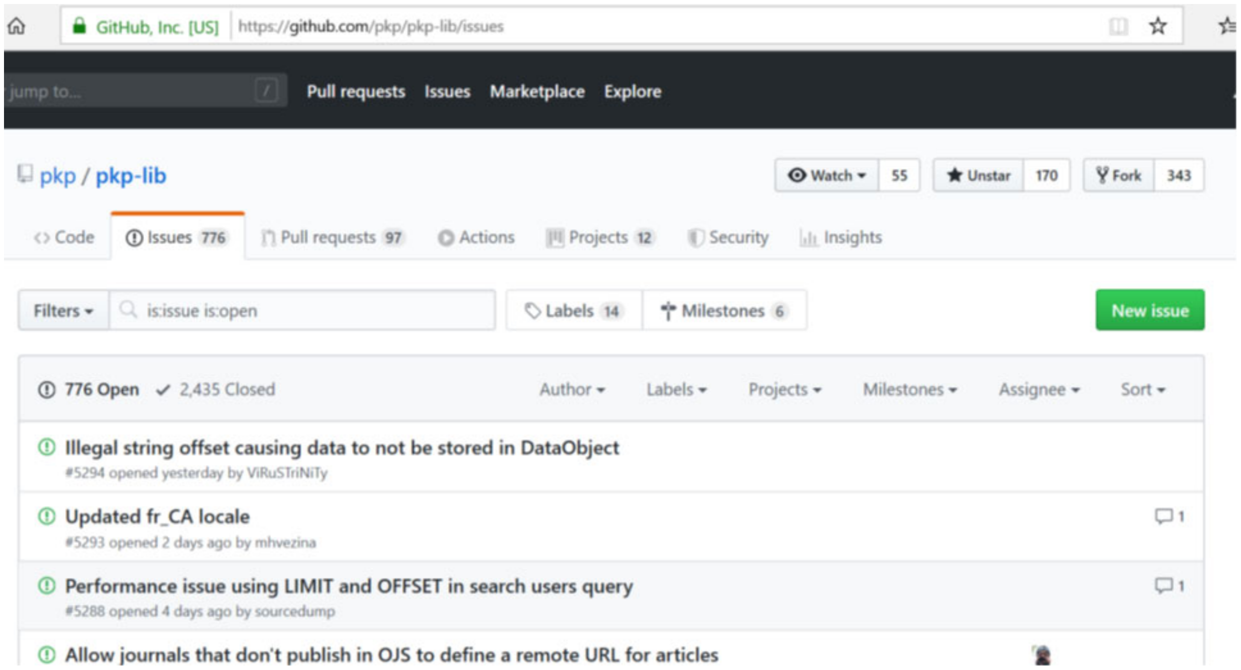
The way bug reports and feature requests are used is generally to keep a list and keep notes about items on that list. Most are not used to make forward progress on an item, so the fact that a ticket or issue exists doesn't mean there is or will be any action on the item.

Generally, open tickets/issues will be reviewed periodically. It might be that there is a conference call or email discussion to comprehensively review open items, to review items that have had any activity in the past 6 months, or some other criteria. Generally, the goal will be to identify whether any items should be acted on. With limited resources and a large list of wants and bugs, it's not possible to do everything.

Ticketing systems are a way for people and organizations using the software to communicate priorities and needs.

GitHub's built-in ticketing system is GitHub Issues. Each issue is stored on a repository. But the common way to handle issues (i.e. support tickets) in GitHub is for an organization to put all issues on a single repository. This might be the most generic repository or a dummy repository with no content specifically created to hold issues. It is extremely uncommon to have issues spread across many repositories, and instead, if GitHub is used, the issues for an entire conceptual software platform will all be stored in one place even though the code is stored in several places.

The best way to locate the ticketing system for a code base is to use a search engine. It may or may not be in GitHub. It is common for organizations publishing code with GitHub to use a different platform as a ticketing system.



Illegal string offset causing data to not be stored in DataObject
#5294 opened yesterday by ViRuSTriNiTy

Updated fr_CA locale 1
#5293 opened 2 days ago by mhvezina

Performance issue using LIMIT and OFFSET in search users query 1
#5288 opened 4 days ago by sourcedump

Allow journals that don't publish in OJS to define a remote URL for articles

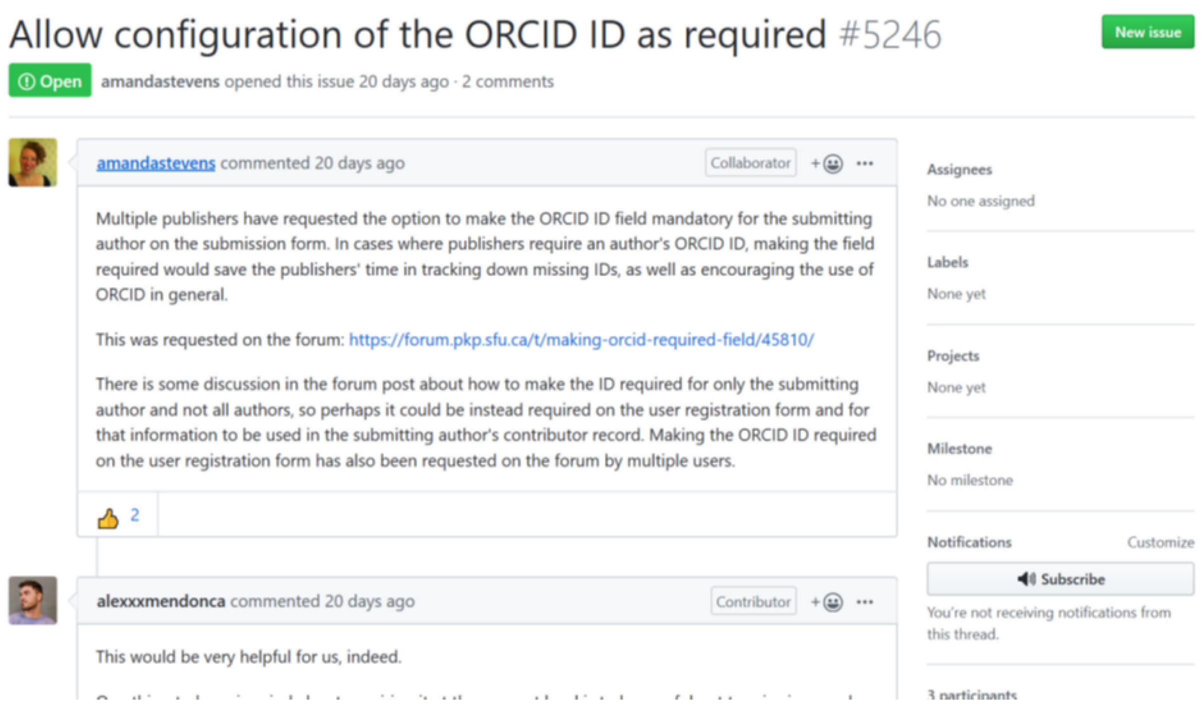
Figure 4, above. Screenshot of the ticketing system for the Open Journal Systems software platform. The GitHub for the software platform, which can be viewed at https://github.com/pkp, contains code spread across 64 different repositories. All tickets are stored on a single repository.

The best way to use a ticketing system is:

- When a problem in the software or a missing feature impacts your ability to do work, look for the authoritative ticketing system for reporting problems with the software and give feedback there.
- Usually, if you have the problem, someone else will have had the problem, too. So, first search for an existing support ticket. Then, interact with the ticket. GitHub will allow you to upvote or comment on an existing issue, and doing this can put that issue on the agenda of meetings held between computer programmers who are active in software development. It can also help to get a fix for the issue resourced, because most people contributing to open source are doing so as part of employment and are working on problems that are either a high priority for their employer or are high profile and so are viewed as a form of publishing that raises the employer's profile and clout. More activity on the ticket shows that there is general interest in moving it forward, and that activity on it will increase the clout of whatever organization coded a fix. If you are able to find the ticketing system for a software platform and add a me-too comment or upvote the ticket/issue, then it is more likely that the issue will be resourced.
- Consider bringing the ticket to the attention of others who you know are affected by the same bug or who want the same feature. This is especially effective if you have discussed the item with someone at another institution and know that it's important at the other organization, too.

Opening a ticket is another possible way to interact with a ticketing system. Most software communities restrict who is and is not able to open a ticket, and most needs of a software are already known and reported but not yet resourced. The most meaningful activity for people using software is likely to be expressing which existing and known needs should be prioritized.

Figure 5, next page. A GitHub Issue for the Open Journal Systems journal publishing software. Another user has made a me-too comment and added detail on needs of his organization. From this view, it is possible for anyone with a GitHub account to click the smiley face and upvote the issue or to scroll to the bottom and add a comment describing how the issue affects people using the software.


Allow configuration of the ORCID ID as required #5246

Open amandastevens opened this issue 20 days ago · 2 comments

amandastevens commented 20 days ago Collaborator

Multiple publishers have requested the option to make the ORCID ID field mandatory for the submitting author on the submission form. In cases where publishers require an author's ORCID ID, making the field required would save the publishers' time in tracking down missing IDs, as well as encouraging the use of ORCID in general.

This was requested on the forum: https://forum.pkp.sfu.ca/t/making-orcid-required-field/45810/

There is some discussion in the forum post about how to make the ID required for only the submitting author and not all authors, so perhaps it could be instead required on the user registration form and for that information to be used in the submitting author's contributor record. Making the ORCID ID required on the user registration form has also been requested on the forum by multiple users.

👍 2

alexxxmendonca commented 20 days ago Contributor

This would be very helpful for us, indeed.

Assignees: No one assigned

Labels: None yet

Projects: None yet

Milestone: No milestone

Notifications: Customize

You're not receiving notifications from this thread.

3 participants

LIBRARY METRICS

Updates on COUNTER and the New Code of Practice 5

Rachel Decker
Hugh & Hazel Darling Law Library
Chapman University Fowler School of Law

In January 2019, Project COUNTER released its Code of Practice 5 (COP5), which was a significant update from COP4 (released in 2014). While there is already a large body of literature discussing COUNTER since its inception, this seemed like a good time to re-introduce the standard to newer librarians, summarize the changes in COP5, and review some of the advocacy efforts related to expanding COUNTER to legal content providers.

What is COUNTER?
For the uninitiated, Project COUNTER (http://www.projectcounter.org) is an international non-profit membership organization of libraries, publishers, and vendors. Since its launch in 2002, it has developed its *Code of Practice*, which is a standard designed to count usage statistics of electronic resources in a library setting, which enables more reliable, consistent, and comparable statistics across platforms. Compliance with COUNTER is not mandatory, but many information vendors have voluntarily made their usage data COUNTER-compliant at the urging of librarians whose institutions and organizations require "cost-per-use" and other metrics to justify their electronic resource spending. Usage data has always had a multitude of problems, and COUNTER aims to alleviate some of those problems through standardization and wide-spread adoption.

The arrangement of COP5 uses three categories of metrics: 1) Usage, 2) Access Denials, and 3) Searches. These metrics are delivered in four types of Master Reports: 1) Platform Reports, 2) Database Reports, 3) Title Reports, and 4) Item Reports. Master Reports have additional metadata to describe the type of database or platform and include a series of added elements and attributes which provide more granular information about items and usage. Users have the option of viewing Master Reports in full or using pre-filtered "standard views," which represent common use cases. Admittedly, COUNTER data uses a specific vocabulary, and those new to COUNTER might want to consult their *Friend-*