# FINOS on GitHub - settings and processes

TODOs:

1. Agenda and Minutes Meeting, how to use GitHub Wiki
2. Team discussions to deliver mailing List and Real-time Chat

**Objective** **2**
FINOS Vocabulary 2
**FINOS Roles and permission mapping** **3**
GitHub permission definitions 3
FINOS Roles 3
Team definition and settings 3
Commits, Issues and Pull Request permissions 5
Repo Collaboration settings 5
**Team collaboration use cases** **7**
Agenda and Minutes Meeting 7
Mailing List and Real-time Chat 7
Documentation (and Product Marketing) websites 7
**Migration checklist** **8**
First notification email for PMC 8
Second notification email for PMC 9
Alternative structure **9**
External to GitHub hosting 9
GitHub hosting in another organization 10
The `Mirroring` experience 10

# Objective

Provide a prescriptive, complete process to migrate a program from the existing GitHub organization to the FINOS GitHub organization.

Define a detailed description of FINOS GitHub organization, in terms of:

1. FINOS Roles and permission mapping
2. Program/Project Team collaboration use cases, which includes:
    a. Commits, Issues and Pull Request permissions
    b. Agenda and Minutes Meeting, currently managed with Atlassian Confluence[1]
    c. Mailing List and Real-time Chat. Group communication, currently based on (Google Groups) mailing-list (exploratory)[2]
    d. Documentation (and Product Marketing) websites (powered by GitHub Pages)
3. FINOS Staff maintenance processes and tasks
4. Community Handbook guidelines and documentation
5. Migration (from current per-program github org structure) checklist and collaterals
6. Program and project rosters

GitHub Teams (and topics) are used to map FINOS entities and bodies with GitHub permission model, across all FINOS Programs and Projects.

## FINOS Vocabulary

- **Program** - Programs are how the FINOS community organizes its nearly 100 projects. Programs help align efforts, address shared business problems, and build out common technical platforms.
- **Project** - A FINOS initiative that is part of a given program. A project may host digital assets (of different types, not only code) using one or more GitHub repositories, often under the github.com/finos organization

More terms are explained below in the FINOS Roles.

[1] Some programs, such as DT and FO, are ready to move to the /finos github org now but do NOT want to move their collaboration use cases to github yet (i.e., they want to stay on the Wiki)

[2] GitHub teams discussion could replace Google Groups Mailing Lists for threaded conversations and/or project announcements. And Gitter could be used as real-time chat conversations (like Symphony, Slack, or Discord, although this question requires banks to validate accessibility to third-party applications.

# FINOS Roles and permission mapping

## GitHub permission definitions

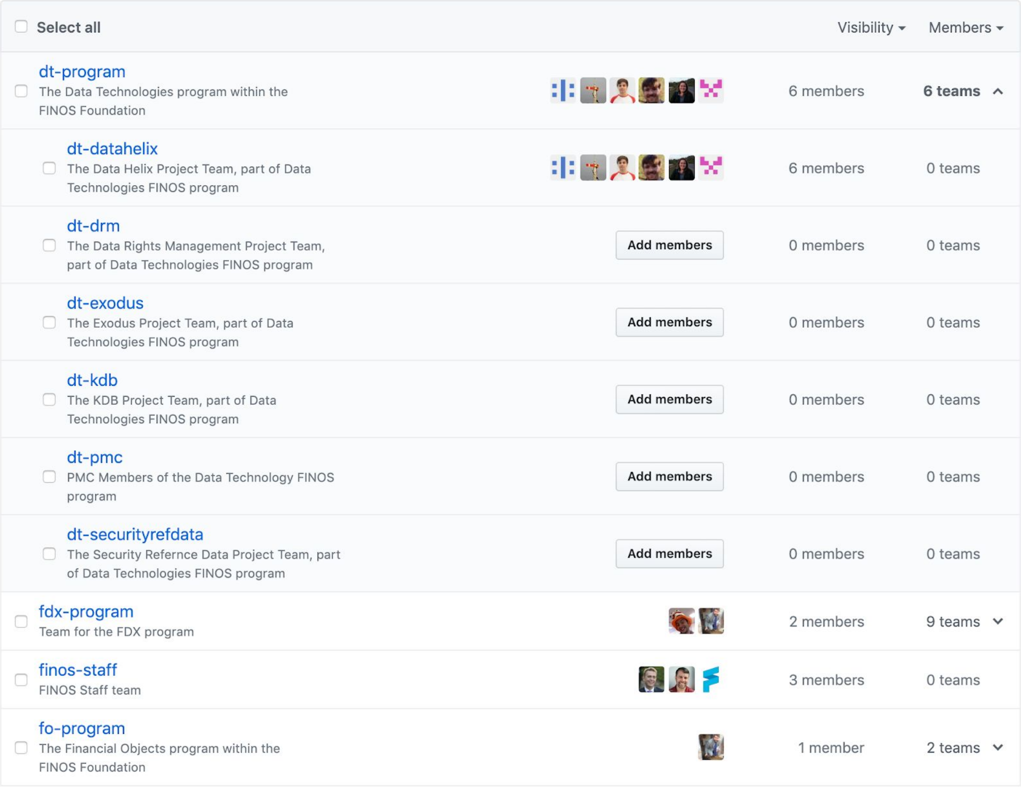
**Admin**: Members will be able to clone, pull, push, and add new collaborators to all repositories (combination of the *maintain* role with the ability to add external collaborators).
**Maintain**: Members can read, clone, and push to this repository. They can also manage issues, pull requests, and some repository settings (combination of the *write*, *triage roles* and some repository management configuration capabilities)
**Read**: Members will be able to clone and pull all (public and private) repositories.
**None**: Members will be able to clone and pull public repositories.

## FINOS Roles

| Finos Role | Description | Permissions | Team name |
|---|---|---|---|
| Project Team | Every GitHub user that is a project maintainer, as granted by the PMC | **Maintain** across all project repos | {program}-{project} |
| External Contributor | Any GitHub user who contributes or participates in a project but is not a regular project team member | **None** across all project repos (default) | No team |
| Project Lead | An **individual** who is responsible for the project | **Admin** across all project repos | Enforced via repository collaboration settings |
| Program Team | Every GitHub user who is part of a program's project team or FINOS leadership | **None** across all program repos (default) | {program}-program |
| PMC Team | Every GitHub user who is part of the PMC | **Maintain** across all program repos | {program}-pmc |
| PMC Lead | An individual who is responsible for the PMC | Not mapped | No team |

## Team definition and settings

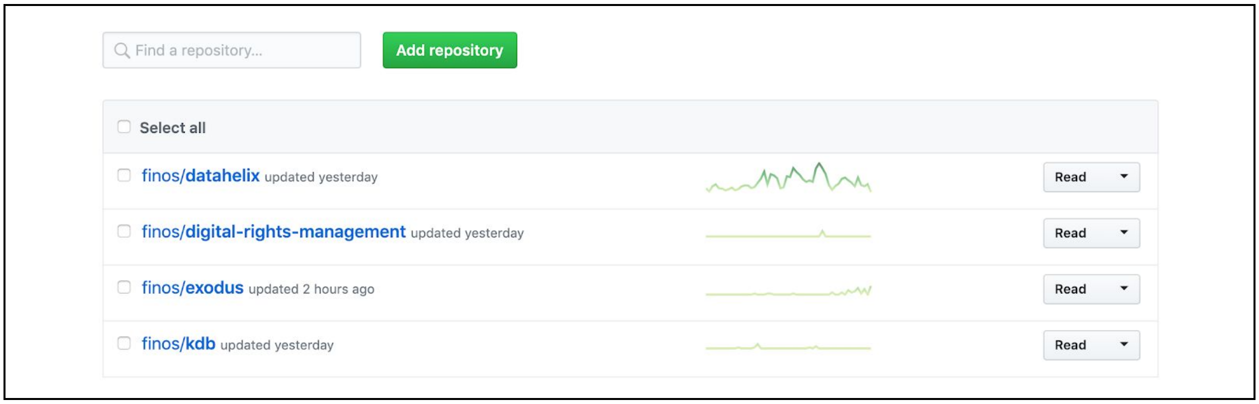
Every program, pmc and project will be mapped as a GitHub team, following a 2-levels hierarchical structure, as shown below.



□ Select all | Visibility ▾ | Members ▾

| Team | | Members | Teams |
|---|---|---|---|
| □ **dt-program**<br>The Data Technologies program within the FINOS Foundation | | 6 members | **6 teams** ⌃ |
| □ dt-datahelix<br>The Data Helix Project Team, part of Data Technologies FINOS program | | 6 members | 0 teams |
| □ dt-drm<br>The Data Rights Management Project Team, part of Data Technologies FINOS program | Add members | 0 members | 0 teams |
| □ dt-exodus<br>The Exodus Project Team, part of Data Technologies FINOS program | Add members | 0 members | 0 teams |
| □ dt-kdb<br>The KDB Project Team, part of Data Technologies FINOS program | Add members | 0 members | 0 teams |
| □ dt-pmc<br>PMC Members of the Data Technology FINOS program | Add members | 0 members | 0 teams |
| □ dt-securityrefdata<br>The Security Refernce Data Project Team, part of Data Technologies FINOS program | Add members | 0 members | 0 teams |
| □ fdx-program<br>Team for the FDX program | | 2 members | 9 teams ⌄ |
| □ finos-staff<br>FINOS Staff team | | 3 members | 0 teams |
| □ fo-program<br>The Financial Objects program within the FINOS Foundation | | 1 member | 2 teams ⌄ |

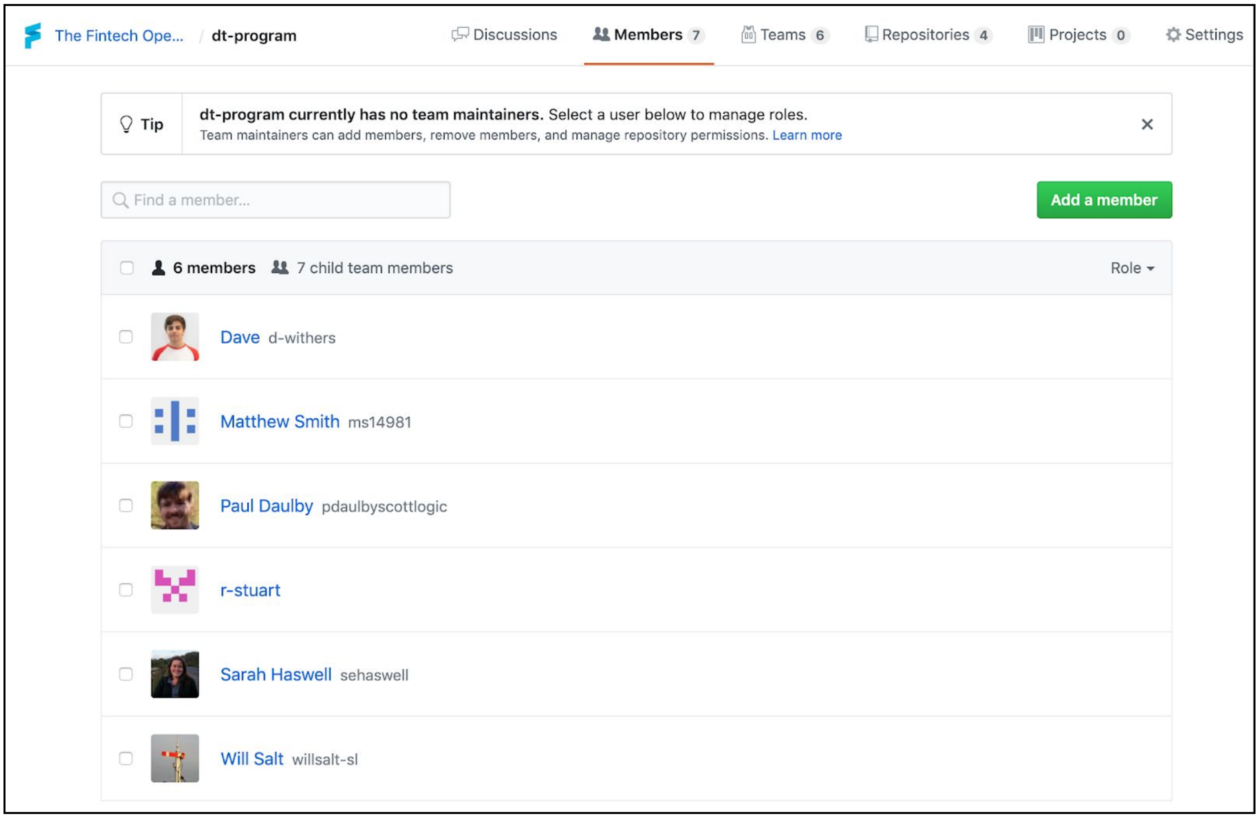
Team inheritance allows all program repositories and members to be listed in single landing pages, shown below.



Find a repository... | Add repository

□ Select all

| Repository | Permission |
|---|---|
| □ **finos/datahelix** updated yesterday | Read ▾ |
| □ **finos/digital-rights-management** updated yesterday | Read ▾ |
| □ **finos/exodus** updated 2 hours ago | Read ▾ |
| □ **finos/kdb** updated yesterday | Read ▾ |



Tip **dt-program currently has no team maintainers.** Select a user below to manage roles.
Team maintainers can add members, remove members, and manage repository permissions. Learn more

□ 6 members 7 child team members

□ Dave d-withers

□ Matthew Smith ms14981

□ Paul Daulby pdaulbyscottlogic

□ r-stuart

□ Sarah Haswell sehaswell

□ Will Salt willsalt-sl

## Commits, Issues and Pull Request permissions

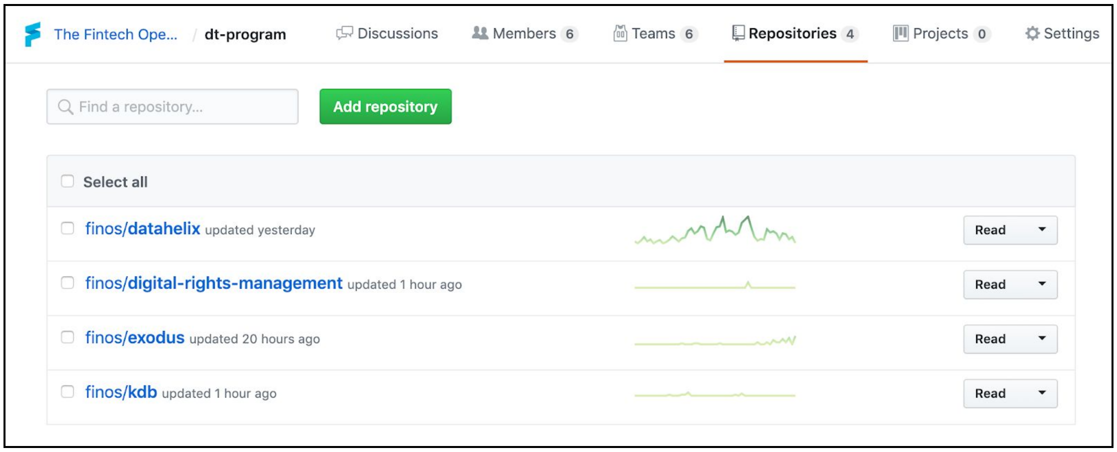
Any GitHub user is able to fork, open issues and submit pull requests to at least one branch of any FINOS hosted repository.

Any GitHub user that is part of a github.com/finos team is also a member of FINOS organization, and therefore the GitHub ID (ie @maoo) can be mentioned on commit messages, issues, pull requests and can be engaged as a issue/pr reviewer/assignee.

**NOTE!** Anyone who left a comment on the issue can be engaged as reviewer/assigned, but any other (GitHub FINOS org) non-member will not show in the list of possible assignees/reviewers.

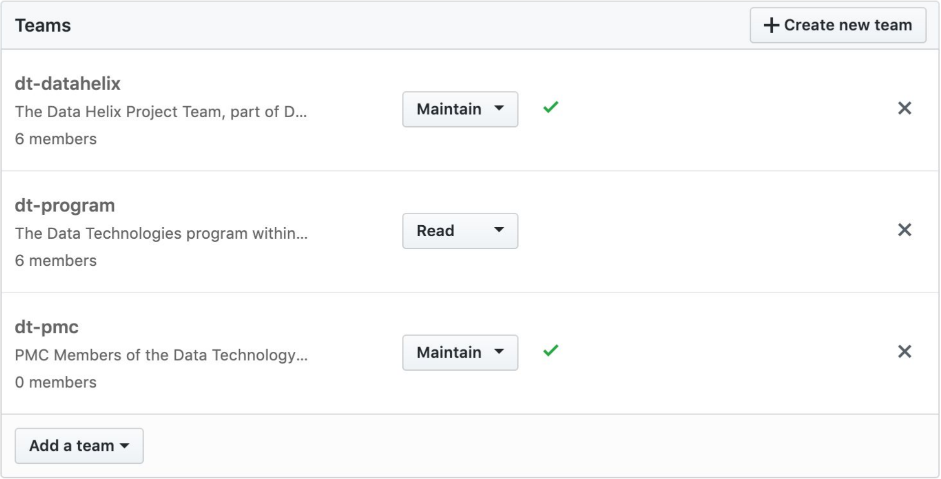
## Repo Collaboration settings

Program-specific GitHub Team (ie ***dt-program***) must have **Read** access, in order to populate the list of repositories at Team level (ie *https://github.com/orgs/finos/teams/dt-program/repositories*)



| | | |
|---|---|---|
| □ finos/datahelix | updated yesterday | Read |
| □ finos/digital-rights-management | updated 1 hour ago | Read |
| □ finos/exodus | updated 20 hours ago | Read |
| □ finos/kdb | updated 1 hour ago | Read |

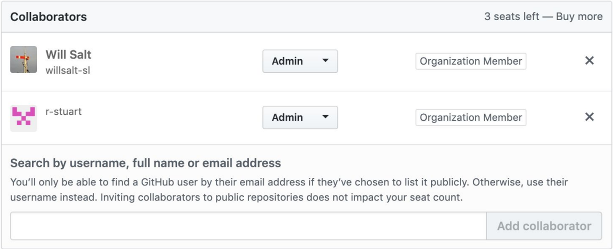
Project-specific GitHub Team (ie ***dt-datahelix***) and program PMC (ie ***dt-pmc***) must have **Maintain** access, in order to give maintainers the right level of access to the repository (see table above).



| | | | |
|---|---|---|---|
| dt-datahelix | The Data Helix Project Team, part of D... | 6 members | Maintain ✓ |
| dt-program | The Data Technologies program within... | 6 members | Read |
| dt-pmc | PMC Members of the Data Technology... | 0 members | Maintain ✓ |

The program-specific group (in this case ***dt-program***) has **Read** access to each program repo, to provide the program landing pages shown above; an alternative solution would be to use GitHub Topics.

The Project Lead will have **Admin** access to the repository, as an individual; note that this permission is already enforced in our current GitHub repository structure.



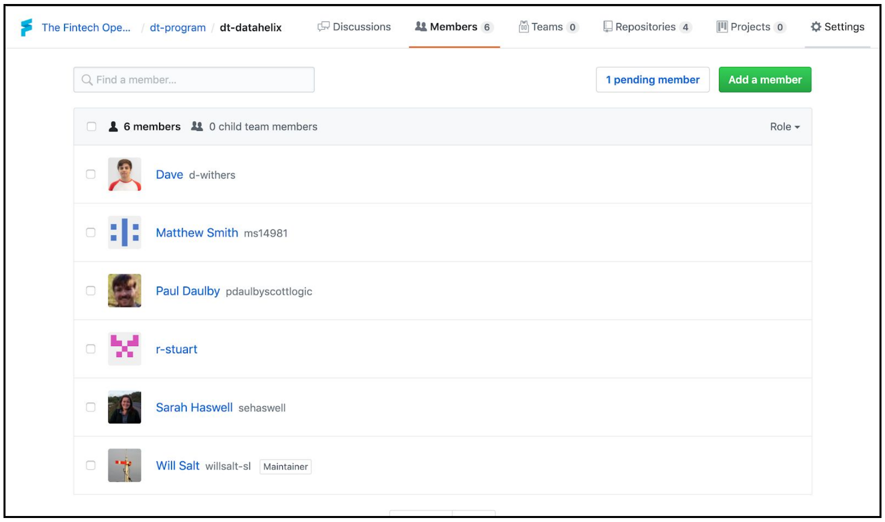
Collaborators — 3 seats left — Buy more

| | | | |
|---|---|---|---|
| Will Salt<br>willsalt-sl | Admin | Organization Member | × |
| r-stuart | Admin | Organization Member | × |

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead. Inviting collaborators to public repositories does not impact your seat count.

The Project Lead must be also configured as **Maintainer** of the related project team (in this case **dt-datahelix**)



□ 6 members 0 child team members Role ▾

□ Dave d-withers

□ Matthew Smith ms14981

□ Paul Daulby pdaulbyscottlogic

□ r-stuart

□ Sarah Haswell sehaswell

□ Will Salt willsalt-sl Maintainer

.

# Team collaboration use cases

## Agenda and Minutes Meeting

TBD
Potentially: Create issue template for your meetings, open issue for each meeting? Use a GitHub Action to automatically open the issue every X weeks.

## Mailing List and Real-time Chat

Team discussions are an alternative solution to Google Groups that allows the project team to discuss project-related matters; it allows private conversations within a team, as well as public ones. It's a good way for teams to share news, as well as major decisions and communications, with the public.

[More, add links, add team inheritance considerations, email notifications (if any, and how to configure them), how pin works, if/how it works together with issues and prs (ie mentions)]

## Documentation (and Product Marketing) websites

This work have been already done, socialized and used extensively; documentation is available on the Project Documentation docs page

# Alternative structure

Some programs may decide to maintain their own GitHub organization and not be joined into the FINOS org. This is likely to maintain legacy program name recognition. *This is an anti-pattern, and I would heavily discourage it in almost all use cases. Any project that believes it needs it's own org to for its own branding is diminishing value from the Foundation.*

## External to GitHub hosting

GitHub provides automatic mirroring for repos hosted outside of GitHub (*e.g. GitLab, a corporate git server, etc*) that will mirror the repositories contents into a repo on GitHub every few hours.

**NOTE: Any code changes made in this location will be overwritten on each mirror operation. Also, if a project is hosting their code and collaboration elsewhere, it may cause more confusion for users who do not get the level of service at the GitHub repository as expected.**

## GitHub hosting in another organization

If a program decides to maintain their own organization on GitHub, the built-in mirroring functionality will no longer work. Instead, we'll have to manually set up either a webhook that will duplicate the changes on each commit, or a scheduled job that will clone, then force push the repository on a routine schedule. Both should be accomplished via GitHub Actions. Jamie can provide these actions upon request.

**NOTE: Any code changes made in the mirrored location would be overwritten on each mirror operation. Also, if a project is hosting their code and collaboration elsewhere, it may cause more confusion for users who do not get the level of service at the GitHub repository as expected.**

## The `Mirroring` experience

When you come to a mirrored repository, there should be a strong signal that the repository is not in active operation at this location, but instead point you towards the proper location. This can be accomplished in the repository description, and should ideally be at the top of the README as well.