Code Contribution Guidelines

Contents

Updated: August 18, 2020

Overview

The Ed-Fi Alliance welcomes code contributions from the community. This documentation provides guidelines for making contributions. Although the details and examples that follow are focused on the Ed-Fi ODS / API, the same principles and requirements are applicable to most Ed-Fi Alliance products.

In general, contributions will be released with the next scheduled product release. Exceptions include cases where contributions introduce breaking changes, in which case they are rolled into the next major version release, and cases where a timely change is needed for the current release (in which case the Ed-Fi Alliance will create a new hotfix/minor release with the appropriate change incorporated).

License Agreements

All publicly available Ed-Fi source code is governed by the Apache License, version 2.0. Accepted code contributions will in turn be shared with the community under the same terms.

In order to clarify the intellectual property license granted with contributions from any person or entity, the Alliance must have a Contributor License Agreement ("CLA") on file that has been signed by each Contributor, indicating agreement to the license terms. This license is for your protection as a Contributor as well as the protection of the Alliance and its users; it does not change your rights to use your own Contributions for any other purpose.

Contributors commit to the CLA via click-through agreement when submitting a pull request through GitHub.

*For the full text of the agreement, see **Ed-Fi Individual Contributor License Agreement** in GitHub.*

Some AI coding assistants reproduce existing application code in violation of that code's license terms. One implication of the Ed-Fi Individual Contributor License Agreement is that such tools must not be used to help generate code for Ed-Fi software projects. The prime example is GitHub CoPilot, which should not be used. AI coding assistants that *do give credit* to the original source are allowed, so long as that original source does not use a "viral" license such as the GNU Public License (GPL), and so long as that source is credited in the appropriate NOTICES.md file.

Planning and Executing a Code Contribution

Whether a student looking for real world software projects to support or a large multi-national corporation, and everything in between, there is room at the table for you.

Tracker Ticket

All code contributions must be linked to an Ed-Fi Tracker ticket; for features and bugs coming from the community, this means first creating a case in the Ed-Fi Community Hub. The support team will review that case and create a ticket in Ed-Fi Tracker if there isn't a similar ticket already.

Except for Tracker tickets labeled "up-for-grabs," it is a best practice to begin a discussion with the Product Owner and/or core development team before beginning to work on a ticket. Simply post a new Comment to begin the conversation, which will help uncover any architectural concerns and implications for other work already in progress. The conversation should include a tentative timeframe for completion so that the development team can plan appropriately for the review process.

If you have a suggested / requested change that is not in already in an Ed-Fi Tracker ticket, then please create a new support case.

Existing Ticket

Interested in helping with an existing issue? Tickets with label "up-for-grabs" are immediately available without further discussion. Please assign to yourself. Here are some Tracker queries to help find these up-for-grab tickets:

- ODS / API Up for Grabs

- Analytics Up for Grabs

⚠️ At any given time there might not be any current tickets with this label.

Other queries will be made available after completion of the respective project's migration to the Apache license, which will also correspond with a loosening of permissions to those projects in Tracker.

As mentioned above, please begin a conversation on other tickets before beginning development work.

New Feature or Improvement

See How To: Get Technical Help or Provide Feedback and How To: Submit a Feature Request. Please be sure to search carefully for existing tickets that might already address the request.

New Bug

See How To: Get Technical Help or Provide Feedback. Please be sure to search carefully for existing tickets that might already the incident.

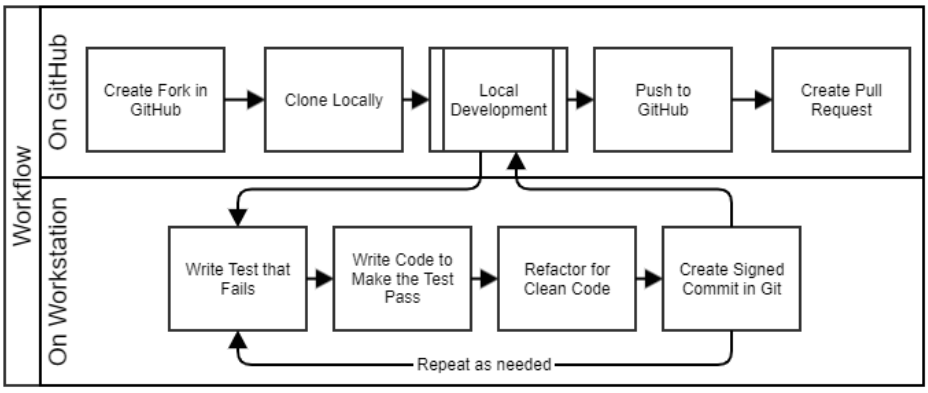
Development Workflow and Standards

In order to contribute code, please follow this development process:

1. In the comments, ask to be assigned to be assigned to the Tracker ticket.
2. Create a fork in GitHub.
    a. References: Forking Workflow (external)

    > ⚠️ Ed-Fi staff and contractors will be invited to join the appropriate organization on GitHub and will be granted write access to appropriate repositories. In those situations, creating a fork is not necessary. Please note that organization members must have Two-Factor Authentication (2FA) enabled, as of September 1, 2022.

3. Create a branch off of the repository's `main` branch. By convention, the Alliance uses the JIRA Issue ID as the branch name (e.g., ODS-3140).
    a. References: Git Feature Branch Workflow (external); *caveat*: the Alliance uses the `main` branch for merges, not the `master` branch as shown in this article
4. Commit your changes and push your changes to GitHub. Keep commits granular and specific, and they must be signed. Multiple small commits are favored over a large commit. Do not include ticket numbers in the commit messages.
    a. References: Signing Git Commits
5. Create a pull request against the original repository's `main` branch. To ensure proper GitHub links in JIRA, pull requests should have the ticket numbers in brackets at the beginning of the title. The title should be a simple version of the addition, change, or fix. This is often similar to the title of the ticket (e.g., "[ODS-3140] PostgreSQL support for Identity Value Mappers").



An Ed-Fi development team member will review your pull request for the following requirements:

1. The pull request has the Ed-Fi Tracker ticket.
2. The changes in the pull request follow the Alliance coding standards.

    a. References: C# Coding Standards, SQL Coding Standards, PowerShell Coding Standards

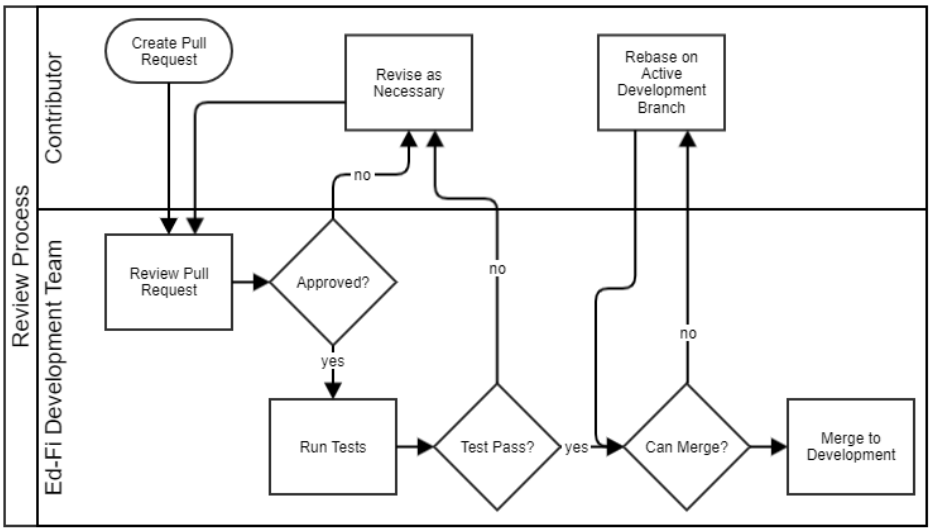
3. The pull request meets the acceptance criteria as defined in the Ed-Fi Tracker ticket.

4. The pull request includes appropriate unit test coverage and/or integration test coverage.
    a. References: Testing Standards

5. Pull request includes Postman tests when a feature or fix directly affects the ODS / API behavior. You can add your tests to Ed-Fi-ODS\Postman Test Suite and/or modify existing tests as appropriate.
    a. References: Testing Standards

6. The pull request changes should be rebased onto the corresponding repository's `main` branch.
    a. References: Git rebase: reapply your changes onto another branch (external)

7. The revised code must pass all existing and new tests. Some tests take long time to run. The reviewer can run these tests on Ed-Fi build servers and provide feedback if the tests have passed or failed.



Google has a nice engineering practices guide to How to do a code review that is recommended reading for all developers contributing or reviewing Ed-Fi code. There may be some Google-specific terminology mixed in there, but the general principles are worth studying.

Multiple Repositories

In the case where changes for a ticket span multiple Git repositories, the pull request will be merged into a feature branch first for testing and validation. The reviewer will create a feature branch based on the repository's `main` branch and then change the target of the pull request to point to the feature branch. The reviewer then will merge the code, keeping the commits in the feature branch until after review comments have been satisfied and functional and unit tests are passing.

Once the above criteria are met, a final pull request will be submitted to move the changes into the repository's `main` branch by the reviewer where the multiple commits will be squashed into a single commit.

Git Operations

Learning Git

While incredibly powerful, Git can be a little daunting to learn. The following tutorials are arranged from basic to more advanced.

- tryGit, an interactive tutorial.
- Learn Git Branching, an interactive tutorial.
- Pull Request Tutorial, with many nice screenshots and some advanced functionality, such as *squash*, *rebase*, and *cherry-pick*.
- Pro Git, the entire book, is available online for free.

Git Security

- All members (staff and contractors) of the Ed-Fi-Alliance-OSS and Ed-Fi-Closed organizations must enable Two-Factor Authentication (2FA) on their GitHub accounts.
    - Open source contributors, including on the Ed-Fi Exchange, do not require 2FA. They will not be granted membership or direct write access in the repositories; instead they will need to fork the repository, make modifications in their forks, and then create the pull request from the fork back to the primary repository.
- All Git commits should be signed to prove authenticity.

See Signing Git Commits.

GUI Clients

Some users prefer to use a GUI client ("windows app") instead of or in addition to the command line. Microsoft Visual Studio and Visual Studio Code both contain robust UI-based Git tools.

The Git web site has a list of stand-alone Git GUI Clients. Some commonly used GUI clients by the ODS development team include: Source Tree, Git Extensions and Git Kraken. For diffs and merges the team primarily uses KDiff3 or Visual Studio Code. However, there are other tools available, including Meld, Diff Merge, and Beyond Compare.

The following command will setup KDiff3 as your default GUI for merges and diffs:

```
git config --global --add merge.tool kdiff3
git config --global --add mergetool.kdiff3.path "C:/Program Files/KDiff3
/kdiff3.exe"
git config --global --add mergetool.kdiff3.trustExitCode false

git config --global --add diff.guitool kdiff3
git config --global --add difftool.kdiff3.path "C:/Program Files/KDiff3
/kdiff3.exe"
git config --global --add difftool.kdiff3.trustExitCode false
```

Further reading on Git can be found on the Git docs site.

Next Steps

You may wish to also review some or all of the following documents:

- Coding Standards - General Principles
- Testing Standards
- Ed-Fi Software Development Lifecycle Home (Ed-Fi Community account required)