General Information for Committers

⚠️ **This page does not at all covering every piece of general information, but we have to start somewhere. Contributions/Reviews are welcome!** ⚠️

This page outlines some of the most commonly referenced information for committers.

Setting up your committer account

You should received an email detailing how to setup your committer account. However, here are some of the additional useful information.

You will be able to find out some commonly asked questions. Such as: how to setup local dev environments for managing commits, merge pull requests; how to setup your ASF LDAP account to manage JIRA tickets as well as wiki edits; etc.

Submit ICLA form

The first step is to fill and sign the ICLA submission form, send it to secretary@apache.org and get confirmation from an Apache secretary.

The ICLA submission form also contains your preferred account id. Check that the preferred id is not already used by other committers here. Check also tips for email setup.

This will enable and trigger creation of your Apache account.

Tips for email setup

Your new apache email address will be <your id>@apache.org.

By default, all its incoming messages will be forwarded to an email address filled in the ICLA submission form. If you want to have some other dedicated email address for the Apache mail (where e.g. you respond to messages as '<your id>@apache.org'), you can always change it later at https://id.apache.org login Forwarding email address. See also Apache email guide.

For example, Gmail has nice features to integrate with an apache account: Settings Accounts and Import Send mail as Add another email address Email address <your id>@apache.org, once it is added you can make it default. It is convenient to have this setup to not switch between email addresses to reply from all the time.

Apache guide to new committer

The ASF website has a very handy guide for new committers. Please first follow the instructions to setup your new committer accounts.

Linking Github account

Flink project is currently hosted on GitBox. Thus committers can directly push to Github repository.

You will have to setup your own Github account in order to be able to directly push Apache/Flink Github repository. Please use the GitBox Account Linking Utility to link your personal Github account to your ASF LDAP account. After you have linked your GitHub account if you do not have write access Flink GitHub repos and cannot push commits, contact a Flink PMC.

Merging personal JIRA account and ASF LDAP account

Some of us are managing multiple ASF JIRA accounts. For example, one can have an early contributor JIRA account while later during committer onboarding, an ASF LDAP committer account was created. Sometimes you will find it useful to merge these different accounts of yours under the one-single ASF LDAP account.

Unfortunately there's no self-service way to merge accounts at this moment. If you wish to merge your personal JIRA account with your newly created ASF LDAP account, you need to file an INFRA JIRA ticket (similar to this one) to have your accounts merged. Ideally, you should be able to login with your Apache id to Jira and Confluence Wiki.

Wiki edit permissions

If you decide to not merge your personal and ASF JIRA account then you might find out that you do not have edit permission to the Apache Flink wiki page. If you find yourself missing wiki edit permissions, please follow the normal procedure: email your JIRA account (preferably your ASF LDAP account) to dev@apache.flink.org in order to request for permission.

Committer responsibilities

You are encouraged to review (if not done recently) the following topics:

- Committer role according to bylaws
- How To Contribute, in particular reviewing pull requests
- Development process, in particular merging pull requests
- All merged contributions should be verified to
    - comply with licensing
    - comply with code style guide
    - be sufficiently documented before the release
    - not introduce performance regressions, check benchmarking (especially performance critical changes, e.g. code paths executed per record, state entry etc)
- Writing a blogpost
- Check this guide and any missing useful information

GitBox vs. Github

Flink project is currently hosted on GitBox. So there are two ways to setup your fetch/push privilege as committer.

⚠️ *NOTE: Due to recently found synchronization issue between GitBox and Github repositories, committers should only use Github repositories for pushing commits*.

Using Github

Make sure you following the setup to link your Github account with your ASF account.

using GitHub command line

Use the following remote as your push/fetch:

Github settings

```
> git remote add <remote_name> git@github.com:apache/flink
```

For example, if you are setting up a remote "github", your setup should look like this:

Github settings

```
> git remote -v
github  git@github.com:apache/flink (fetch)
github  git@github.com:apache/flink (push)
```

using GitHub web UI

In addition to the command line, you can also use web UI to merge pull requests, either using the "Squash and merge" button or "Rebase and merge" button.

For the "Squash and merge" option, the pull request's commits are squashed into a single commit and merged into the master branch using the fast-forward option. This is often used to squash fixup commits thus can retain the original changes with a clear Git history.

For the "Rebase and merge", all commits from the PR are rebased and added onto the master branch. This is often used when we want to preserve all the commits, i.e., preserve "refactor" and "feature" differentiation in history rather than squash everything.

Using GitBox

Using GitBox, you are not required to link your Github account using the utility. Your remote setup should look like the following:

Github settings

```
> git remote -v
gitbox  https://gitbox.apache.org/repos/asf/flink.git (fetch)
gitbox  https://gitbox.apache.org/repos/asf/flink.git (push)
```

Notices for merging a pull request

1. Always merge the code on condition that tests are passed.
2. Squash fixup commits thus we can retain the original changes with a clear Git history. However, it's important to notice that some pull requests often consist of multiple commits where we want to preserve "refactor" and "feature" differentiation in history rather than squash everything.
3. Check the commit message. Ensure that the Jira id and components are right, e.g., `[FLINK-XXX][docs] XXX`. Ensure the title is clear and meaningful.
4. Once the code has been merged, you should pay attention to the Flink build and check if everything is right. You can subscribe to the build mailing list to get notifications. Create a Jira if something is broken on the master branch.