Making contributions with GitHub

There are many ways to be a contributor, and we're here to help you on your way! You may:

- Join the discussion and share ideas in our Slack channel and regular TSWG and TATF meetings.
- Raise an issue or feature request in our issue tracker
- Help another contributor with one of their questions, or a documentation review
- Suggest improvements by supplying a Pull Request or opening a Discussion
- Evangelize our work together in conferences, podcasts, and social media spaces.

Contributing

This section is designed to help you make an edit if you aren't familiar with using GitHub and want to make a change to any repository.

Basic Change Flow

If you're not familiar with GitHub and Git, here are a few simple steps you can use to get going and contribute to the repository *without ever touching the command line*.

There is also a Video Walkthrough of how to do this if you prefer to learn over video.

1. Fork the repo into your account
   blocked URL
   blocked URL
2. Find the file you want to edit. Click the pen tool on the top right of the file to edit it. If you want to add a file, click "Add File". Click "Create new branch". Learn More
   blocked URL
   blocked URL
3. Make your changes. When you are ready, click Pull Request on the bar above the file. Then create `New Pull Request` and choose to set the request to merge to the `main` branch. Put any information you want to describe your changes in the description, and you're done! Learn More.
   blocked URL
   blocked URL

Updating your PR on Changes Requested

If you've gotten some comments about needing to fix a PR, the process is really simple. You do NOT need to create a new PR.

1. Go to the file in **your branch** that you want to edit.
2. Click the edit button (step 2 from when you created the initial edit)
3. Make edits, but this time, instead of creating a **new** branch, commit to the same branch. Your changes will propagate to the PR.
   blocked URL
4. You may want to re-request a review. Click the sync button to do so:

blocked URL

Advanced Change Flow

As a more advanced user, there are a few ways you can manage your GitHub repository:

1. You can use the GitHub web editor
2. You can use GitHub Desktop
3. You can use the CLI
4. Something else! Git is extremely powerful if you want to spend time researching.

Commits

- You must sign your commits

- Please try to keep your commit history clean. We do not enforce this but it is encouraged.

Branching:

- Nobody will work directly on the `main` branch. All changes must occur over a PR and off the main branch.
- We encourage, but do not enforce branching naming using the following schema: `<type>/<description>` ex. `edit/fix-fig4-label`.
- The main branch should always have the latest approved changes
- Custom "feature" branches may be used for special purposes, e.g. for groups to work on a larger section of text
- Branching will occur on forks, not over the main repository. This is to ensure that the main repository is not cluttered with lots of branches from contributors.

Tagging

- Version tags are used for each release of the document
- Releases should be versioned and if needed, appended with a pre-release tag, e.g. "v1", "v3-RC1", "v4-IIW39"
- Versioning should be simple, only major releases numbered, prepended with letter v. e.g. v1, v2, v3.
- Branch protection rules will exist to not allow a person to directly commit to `main`.

Pull Requests

The below documents some basic best practices for your pull requests.

- Please make sure to ask an editor to review your Pull Request.
- Write descriptive and consistent names.
- Create a clear PR title and description.
- Keep PRs as short as possible
- Try to keep a transparent audit trail of your conversation so people can follow it.
- Avoid rewrites by getting feedback early.
- Request additional reviewers to create dialogue.
- Be precise in your comments about what needs to be fixed.

Review

- Review of PRs is done by appointed editors. See the GOVERNANCE file for more information.
- Issues labelled with `status: needs-review` should contain a PR code OR the change text directly in the issue (for those not Git-savvy)
- When you create an issue or a PR, please try to tag them appropriately, including adding a `status: needs-review` label to the issue/PR.
- Every week, the editors will go through the issues and tag them appropriately.
- If the PR is ready to be merged, it is tagged with `accepted` -label
- In normal circumstances, any editor may merge changes tagged with a label `accepted`
- The editor can try to merge the PR to the main branch.
- If PR is not rebased and commit histories are not in sync, the PR can be merged if there are no overlapping changes with the history. In this case, it is the merging editor's responsibility to ensure that the merging is clean and no unwanted changes happen.
- In case of change conflicts, the editor requests the PR creator to rebase against the current main branch and resubmit the PR.
- After review, an `editor` may change the status to `status: last-call`. This would signal a 5-day delay for close.
- If nobody disagrees with the `status: last-call`, the issue is accepted and merged back into `main`.
- Sometimes the editor group may agree to a controlled merging process and decide that merging happens only by a selected editor (e.g. release owner) or during editor meetings. This may happen when a release of the document is coming soon and only some specific changes are allowed.

Issues

- Anyone may raise an issue
- Every week editors will go through the new issues, and label them.
- Please provide as much clarity as possible around an issue topic. Ideally, you are always answering the following:
    - what is the issue?
    - why is it important?
    - what would trigger the closure of the ticket?
- Issues that are not commented on for over 90 days, and reviewed to not be relevant will be closed by an editor
- Please do not hesitate to self-assign yourself if you would like to address an issue.

Labels

Priority Labels

Priority labels are used to describe the impact and focus of the issue. Higher priority means it is more likely to find focus within the group.

Priority	Label	Usage
priority	critical	Progress on this issue is critical to the group's forward progress.
priority	high	It is important for the group to resolve this issue soon.
priority	medium	This issue is important to resolve before the next release.

priority	low	This issue is "nice to have" for the next release, but could be deferred if time runs out.

Type Labels

Type labels are labels the define the nature of the issue and/or the correction itself.

Type	Label	Usage
type	editorial	The issue only involves wording and not normative content.
type	content	The issue involves normative content; resolution requires group consensus.
type	correction	The issue is fixing a recognized problem in the current version.
type	formatting	The issue involves fixing formatting.
type	figure	The issue involves a figure that it missing or needs to be revised.
type	admin	The issue is administrative and NOT about the deliverable.

Status Labels

Status labels are labels that are used to help identify the current state of the issue, so that we may accurately classify the work to do on it.

Status	Label	Usage
status	unassigned	The issue is new and has not yet been assigned to anyone.
status	in-progress	The issue has been assigned and work is in progress.
status	needs-review	A resolution (or concrete step forward) has been proposed and needs review.
status	blocked	Progress is currently blocked; the block should be explained in a comment.
status	on-hold	Progress is currently on hold; the reason should be explained in a comment.
status	deferred	Consensus has been reached that this issue can be deferred to a subsequent version.
status	abandoned	Consensus has been reached that this issue can be abandoned.
status	PR-needed	Consensus has been reached and this issue is now waiting for a PR to be submitted.
status	PR-in-progress	The issue is linked to a PR that is in progress
status	PR-completed	The issue is linked to a PR that is complete and waiting for review.
status	PR-accepted	The issue is linked to a PR that has been accepted and is waiting for merge.
status	PR-merged	The issue is linked to a PR that has been merged; this issue can now be closed.
status	status: last-call	The issue has been resolved by some other mechanism documented in the comments and is now in **the 5-day last call.**

Attribution:

Originally written by Andor Kesselman for the TechArch repo (https://github.com/trustoverip/TechArch/blob/main/CONTRIBUTING.md)