Contributing to opentelemetry-go-contrib

The Go special interest group (SIG) meets regularly. See the OpenTelemetry community repo for information on this and other language SIGs.

See the public meeting notes for a summary description of past meetings. To request edit access, join the meeting or get in touch on Slack.

Development

There are some generated files checked into the repo. To make sure that the generated files are up-to-date, run `make` (or `make precommit` - the `precommit` target is the default).

The `precommit` target also fixes the formatting of the code and checks the status of the go module files.

If after running `make precommit` the output of `git status` contains `nothing to commit, working tree clean` then it means that everything is up-to-date and properly formatted.

Pull Requests

How to Send Pull Requests

Everyone is welcome to contribute code to `opentelemetry-go-contrib` via GitHub pull requests (PRs).

To create a new PR, fork the project in GitHub and clone the upstream repo:

```
$ git clone https://github.com/open-telemetry/opentelemetry-go-contrib
```

This would put the project in the `opentelemetry-go-contrib` directory in current working directory.

Enter the newly created directory and add your fork as a new remote:

```
$ git remote add <YOUR_FORK> git@github.com:<YOUR_GITHUB_USERNAME>/opentelemetry-go
```

Check out a new branch, make modifications, run linters and tests, update `CHANGELOG.md` and push the branch to your fork:

```
$ git checkout -b <YOUR_BRANCH_NAME>
# edit files
# update changelog
$ make precommit
$ git add -p
$ git commit
$ git push <YOUR_FORK> <YOUR_BRANCH_NAME>
```

Open a pull request against the main `opentelemetry-go-contrib` repo. Be sure to add the pull request ID to the entry you added to `CHANGELOG.md`.

How to Receive Comments

- If the PR is not ready for review, please put `[WIP]` in the title, tag it as `work-in-progress`, or mark it as `draft`.
- Make sure CLA is signed and CI is clear.

How to Get PRs Merged

A PR is considered to be **ready to merge** when:

- It has received two approvals from Approvers/Maintainers (at different companies).
- Feedback has been addressed.
- Any substantive changes to your PR will require that you clear any prior Approval reviews, this includes changes resulting from other feedback. Unless the approver explicitly stated that their approval will persist across changes it should be assumed that the PR needs their review again. Other project members (e.g. approvers, maintainers) can help with this if there are any questions or if you forget to clear reviews.
- It has been open for review for at least one working day. This gives people reasonable time to review.
- Trivial change (typo, cosmetic, doc, etc.) doesn't have to wait for one day.
- `CHANGELOG.md` has been updated to reflect what has been added, changed, removed, or fixed.
- Urgent fix can take exception as long as it has been actively communicated.

Any Maintainer can merge the PR once it is **ready to merge**.

Style Guide

- Make sure to run `make precommit` - this will find and fix the code formatting.
- Check opentelemetry-go Style Guide

Adding a new Contrib package

To add a new contrib package follow an existing one. An empty Sample instrumentation provides base structure with an example and a test. Each contrib package should be its own module. A contrib package may contain more than one go package.

Folder Structure

- instrumentation/<instrumentation-package> (**Common**)
- instrumentation/<instrumentation-package>/trace (**specific to trace**)
- instrumentation/<instrumentation-package>/metrics (**specific to metrics**)

Example

- instrumentation/gorm/trace
- instrumentation/kafka/metrics

Approvers and Maintainers

Approvers:

- Evan Torrie, Verizon Media
- Josh MacDonald, LightStep
- Sam Xie
- David Ashpole, Google
- Gustavo Silva Paiva, LightStep

Maintainers:

- Anthony Mirabella, Centene
- Tyler Yahn, New Relic

Become an Approver or a Maintainer

See the community membership document in OpenTelemetry community repo.