## Overview

Polyaxon project would love to welcome your contributions. There are several ways to help out:

\* Help other Polyaxon's users on [the community channel](https://polyaxon.com/slack/).

\* Testing and quality assurance.

\* Contribute and help us improve the [documentation](https://github.com/polyaxon/polyaxon/tree/master/site).

\* Create an [issue](https://github.com/polyaxon/polyaxon/issues) on GitHub, if you have found a bug.

\* Create [guides and tutorials](/docs/guides/).

\* Author [plugins and integrations](https://polyaxon.com/integrations/).

\* Author [public reusable components](https://github.com/polyaxon/polyaxon-hub/).

\* Write test cases for open bug issues.

\* Write patches for open bug/feature issues, preferably with test cases included.

\* Blog about different ways you are using Polyaxon and promote Polyaxon to other users.

There are a few guidelines that we need contributors to follow so that we have a chance of keeping on top of things.

## Reporting issues

Polyaxon has probably many issues and bugs, a great way to contribute to the project is to send a detailed report when you encounter an issue. We always appreciate a well-written, thorough bug report, and will thank you for it!

Sometimes Polyaxon is missing a feature you need, and we encourage our users to create and contribute such features.

Check the current [issues](https://github.com/polyaxon/polyaxon/issues) if doesn't already include that problem or suggestion before submitting an issue.

If you find a match, add a quick "+1", Doing this helps prioritize the most common problems and requests.

When reporting issues, please include your host OS (Ubuntu 14.04, CentOS 7, etc), and the version of the libraries you are using.

Please also include the steps required to reproduce the problem if possible and applicable. This information will help us review and fix your issue faster.

## Development guide

If you're a developer looking to help, but you're not sure where to begin:

Check out the [good first issue](https://github.com/polyaxon/polyaxon/labels/good%20first%20issue) label on Github,

which contains small piece of work that have been specifically flagged as being friendly to new contributors.

Or, if you're looking for something a little more challenging, there's a broader [help wanted](https://github.com/polyaxon/polyaxon/labels/help%20wanted) label.

## Contribution Workflow

Before you contribute to Polyaxon, there are a few things that you'll need to do

1. Make sure you have a [GitHub account](https://github.com/signup/free).

2. Submit an [issue](https://github.com/polyaxon/polyaxon/issues), assuming one does not already exist.

3. Clearly describe the issue/feature:

\* Steps to reproduce if it is a bug (Make sure you fill in the earliest version that you know has the issue).

\* Use case(s) and current alternative(s) if it's a feature request.

4. Make sure to coordinate the effort and ensure that the feature is approved and is not being worked on.

5. Fork the repository on GitHub and create a pull request.

### Areas

\* Tracking module: this module is purely based on Python and Polyaxon's APIs.

if you have knowledge about a particular framework and you would like to contribute a callback or an auto-logging module,

we would be more than grateful.

\* Programmatic visualizations: this module is also purely based on Python and leverages some known plotting libraries,

if you would like to provide/improve a backend for driving programmatic visualizations,

or for helping out with templates for driving insights using Python (or another language) please feel free to reach out.

\* Public components: Polyaxon allows users to package information about their runs to ensure reproducibility and portability.

This abstraction also allows to packages reusable components and enables users to use them in a simple way using Polyaxon CLI/Client/API.

\* Platform (API/Scheduler), Compiler, Polyaxonfile: if you would like to contribute to one of the areas that require long term compatibility and maintenance,

please make sure that the feature request is approved, before investing any time. For some features that are quite involved, they can only be developed internally.

### Making Changes

\* Create a topic branch from where you want to base your work.

\* This is usually the master branch.

\* Only target an existing branch if you are certain your fix must be on that branch.

\* To quickly create a topic branch based on master; `git checkout -b my\_contribution origin/master`.

It is best to avoid working directly on the `master` branch. Doing so will help avoid conflicts if you pull in updates from origin.

\* Make commits of logical units. Implementing a new function and calling it in

another file constitute a single logical unit of work.

\* A majority of submissions should have a single commit, so if in doubt, squash your commits down to one commit.

\* Use descriptive commit messages and reference the #issue number.

\* Core test cases should continue to pass. (Test are in progress)

\* Pull requests must be cleanly rebased on top of master without multiple branches mixed into the PR.

### Which branch to base the work

All changes should be be based on the latest master commit.

## Questions

If you need help with how to use this library, please check the list of examples, if it is still unclear you can also open an issue.