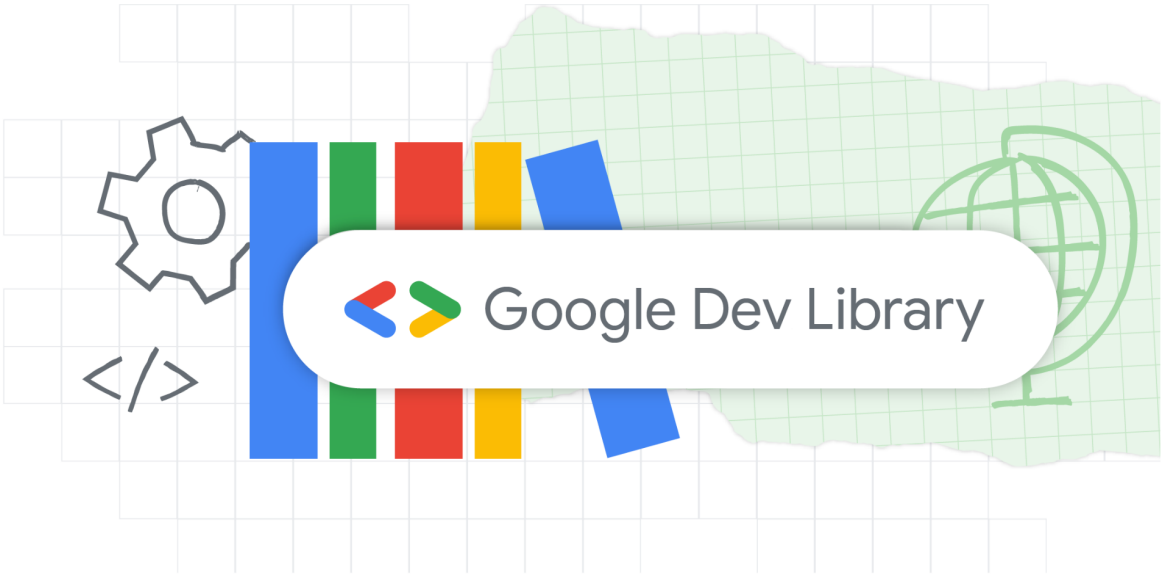
Dev Library Contributor Guidelines



Context: This document will cover the guidelines to be shared with the contributors covering what they should look at, rejection reasons, guidelines to inculcate in their pieces.

The contributor guidelines are crafted for the authors to let them excel their work and create better projects for everyone to use.

Submission Workflow

Dev Library is a platform that displays open-source projects and articles created by external developers using any of the Google technologies like Angular, Android, Flutter, Firebase, Google assistant, Google Cloud, and Machine Learning. We aim to acknowledge the work and bring those projects under one umbrella for other developers to get inspired and start their own projects.

Steps to be followed for submission :-

1. Contributors can make a submission by navigating to our Advocu form link. The form link appears after clicking the Submit button on the Dev Library site.
2. Fill out the project/blog details along with your personal information.
3. Google engineers go through the submissions and reject/accept them based on its quality/ freshness/ relevancy.
4. Post acceptance, we display those projects and articles on our site under each product area with a relevant set of tags to filter out content.

This is the only process we follow for submission and curation of the content any developer submits. For any changes to the submission or your own author profile, feel free to raise a pull request.

Contributor Guidelines

Contributor guidelines are designed to help you craft a quality blog post or polished open-source project with updated technology and minimal errors. Google engineers who go through the submissions, check the content against a number of factors. Based on the factors, the content gets accepted or rejected. In case it gets rejected, we've listed down a few parameters for each product area that can be taken into consideration for improvement and re-submission.
Some common factors across all the product areas are:

- **Freshness:** Content should not be more than **6 months old** including the last update to GitHub projects. Content should **not use deprecated ways** for building an app.
- **Fully functional README:** All GitHub submissions ***must*** include a README outlining what the project does and how to use it.
- **Community Guidelines**: All content should comply with Google's open source community guidelines.
- **Paywall:** Content should ***not*** be behind a paywall. Content should be free to access for all developers.
- **Promotional:** Content should ***not*** be promotional for the author's employer.
- **Relevancy:** Content that is unrelated to the company and useful for the community may be approved. Example: Square (Block) releases a variety of libraries for the community without it being associated with Square's products.
- **Context:** Content should add more context or information than the current official documentation.
- **Language and grammar:** Grammar must be reasonably polished in whichever language the content is published in.

Android

For creating blogs or projects using the Android technology, consider the following factors:

- Code should be relatively clean.
- Code should only be written in Kotlin (or C++ if using the NDK). Java should be discouraged.
- Code should use named variables over "val a = 100" that could cause confusion through ambiguity.

For more information, check out Android's official documentation.

Angular

For creating blogs or projects using the Angular technology, consider the following factors:

- The submitted project should be on the recent release of Angular (version N-2, currently v11+), and actively maintained with the most recent release in the last 6 months.
- All repositories and blog posts should contain demos or working code and installation guides. All code and demos should include ivyEnabled = true
- Blog posts and projects should focus on how to use Angular. It should not be used as a side framework.

For more information, check out Angular's official documentation.

Flutter

For creating blogs or projects using the Flutter technology, consider the following factors:

- All the GitHub projects should be migrated to Dart's null safety. Failing which, the project will definitely be rejected.
- Make sure the minimum Dart SDK version is >= 2.12.0.
- Flutter is a visual SDK so including screenshots in the READMEs improves your chances of acceptance.
- The presence of an analysis_options.yaml file in the project root is a positive signal.
- Having tests is also a huge plus.
- Encourage proper code formatting. The coding style in our samples on dartpad.dev can be taken as a good idea of what properly formatted Flutter code looks like.

For more information, check out Flutter's official documentation.

Firebase

For creating blogs or projects using the Firebase technology, consider the following factors:

- Use Firebase emulators from the command line
- Web content should always be using the **v9 SDKs** or later.
- Don't use `auth != null`.
- Don't use Anonymous Authentication.

For more information, check out Firebase's official documentation.

Google Cloud

For creating blogs or projects using the Google Cloud technology, consider the following factors:

- Avoid articles that contain any comparison with other cloud providers, opinionated articles, market share, or customer articles.
- The major areas of focus for Google Cloud are:
    - Infrastructure
    - Data (Databases and Analytics)
    - Machine Learning
    - Security

    - Networking
    - Workspace/ G Suite
    - App Development
    - Optimisation (Cost, Performance,etc)

For more information, check out Google Cloud's official documentation.

Machine learning

For creating blogs or projects using the Machine Learning technology, consider the following factors:

- Prefer projects and article depicting some practical use-cases or latest product features
- Articles with privacy concerns, or person/face verification are discouraged - though the use of face based models to do other things is fine.
- Focus areas for machine learning projects are:
    - TensorFlow.js and Vertex AI top priority,
    - JAX, Flax,
    - TensorFlow

For more information, check out Machine Learning's official documentation or Tensorflow.

Tone and style guide

For tone and style guide, please refer to the Google developer's style guide. A few quick pointers to keep a check on:

1. Preferably use Active voice while writing the project details. Knowing who is performing the action, who needs to perform it, and on whom it is being performed, is extremely important.
2. Avoid use of foul language.
3. Write inclusive documentation (Avoid he/she, use they or both)
4. Capitalization: Use sentence-case capitalization.
5. Use more numbers (for steps), bullets (for everything else), and sections (wherever possible).
6. Use of visuals makes your article or README much easier to read.