Thanks for deciding to contribute to meshoptimizer! These guidelines will try to help make the process painless and efficient.

## Questions

If you have a question regarding the library usage, please [open a GitHub issue](https://github.com/zeux/meshoptimizer/issues/new).

Some questions just need answers, but it's nice to keep them for future reference in case other people want to know the same thing.

Some questions help improve the library interface or documentation by inspiring future changes.

## Bugs

If the library doesn't compile on your system, compiles with warnings, doesn't seem to run correctly for your input data or if anything else is amiss, please [open a GitHub issue](https://github.com/zeux/meshoptimizer/issues/new).

It helps if you note the version of the library this issue happens in, the version of your compiler for compilation issues, and a reproduction case for runtime bugs.

Of course, feel free to [create a pull request](https://help.github.com/articles/about-pull-requests/) to fix the bug yourself.

## Features

New algorithms and improvements to existing algorithms are always welcome; you can open an issue or make the change yourself and submit a pull request.

For major features, consider opening an issue describing an improvement you'd like to see or make before opening a pull request.

This will give us a chance to discuss the idea before implementing it - some algorithms may not be easy to integrate into existing programs, may not be robust to arbitrary meshes or may be expensive to run or implement/maintain, so a discussion helps make sure these don't block the algorithm development.

## Code style

Contributions to this project are expected to follow the existing code style.

`.clang-format` file mostly defines syntactic styling rules (you can run `make format` to format the code accordingly).

As for naming conventions, this library uses `snake\_case` for variables, `lowerCamelCase` for functions, `UpperCamelCase` for types, `kCamelCase` for global constants and `SCARY\_CASE` for macros. All public functions/types must additionally have an extra `meshopt\_` prefix to avoid symbol conflicts.

## Dependencies

Please note that this library uses C89 interface for all APIs and a C++98 implementation - C++11 features can not be used.

This choice is made to maximize compatibility to make sure that any toolchain, including legacy proprietary gaming console toolchains, can compile this code.

Additionally, the library code has zero external dependencies, does not depend on STL and does not use RTTI or exceptions.

This, again, maximizes compatibility and makes sure the library can be used in environments where STL use is discouraged or prohibited, as well as maximizing runtime performance and minimizing compilation times.

The demo program uses STL since it serves as an example of usage and as a test harness, not as production-ready code.

## Testing

All pull requests will run through a continuous integration pipeline using GitHub Actions that will run the built-in unit tests and integration tests on Windows, macOS and Linux with gcc, clang and msvc compilers.

You can run the tests yourself using `make test` or building the demo program with `cmake -DBUILD\_DEMO=ON` and running it.

Unit tests can be found in `demo/tests.cpp` and functional tests - in `demo/main.cpp`; when making code changes please try to make sure they are covered by an existing test or add a new test accordingly.

## Documentation

Documentation for this library resides in the `meshoptimizer.h` header, with examples as part of a usage manual available in `README.md`.

Changes to documentation are always welcome and should use issues/pull requests as outlined above; please note that `README.md` only contains documentation for stable algorithms, as experimental algorithms may change the interface without concern for backwards compatibility.

## Sensitive communication

If you prefer to not disclose the issues or information relevant to the issue such as reproduction case to the public, you can always contact the author via e-mail (arseny.kapoulkine@gmail.com).