



Compiler Explorer

Is an interactive compiler exploration website. Edit code in C, C++, C#, F#, Rust, Go, D, Haskell, Swift, Pascal, ispc, Python, Java, or any of the other 30+ supported languages components, and see how that code looks after being compiled in real time.

[Bug Report](#) · [Compiler Request](#) · [Feature Request](#) · [Language Request](#) · [Library Request](#) · [Report Vulnerability](#)

Overview

Multiple compilers are supported for each language, many different tools and visualizations are available, and the UI layout is configurable (thanks to GoldenLayout).

Try out at godbolt.org, or run your own local instance. An overview of what the site lets you achieve, why it's useful, and how to use it is available [here](#).

Compiler Explorer follows a Code of Conduct which aims to foster an open and welcoming environment.

Compiler Explorer was started in 2012 to show how C++ constructs are translated to assembly code. It started as a `tmux` session with `vi` running in one pane and `watch gcc -S foo.cc -o -` running in the other.

Since then, it has become a public website serving over 3,000,000 compilations per week.

You can financially support this project on [Patreon](#), [GitHub](#), [Paypal](#), or by buying cool gear on the [Compiler Explorer store](#).

Using Compiler Explorer

FAQ

There is now a FAQ section in the repository wiki. If your question is not present, please contact us as described below, so we can help you. If you find that the FAQ is lacking some important point, please feel free to contribute to it and/or ask us to clarify it.

Videos

Several videos showcase some features of Compiler Explorer:

- Compiler Explorer 2023: What's New?: Presentation for CppNorth 2023.
- Presentation for CppCon 2019 about the project
- Older 2 part series of videos which go into a bit more detail into the more obscure features.
- Just Enough Assembly for Compiler Explorer: Practical introduction to Assembly with a focus on the usage of Compiler Explorer, from CppCon 2021.
- Playlist: Compiler Explorer: A collection of videos discussing Compiler Explorer; using it, installing it, what it's for, etc.

A Road map is available which gives a little insight into the future plans for **Compiler Explorer**.

Developing

Compiler Explorer is written in TypeScript, on Node.js.

Assuming you have a compatible version of `node` installed, on Linux simply running `make` ought to get you up and running with an Explorer running on port 10240 on your local machine: `http://localhost:10240/`. If this doesn't work for you, please contact us, as we consider it important you can quickly and easily get running. Currently, **Compiler Explorer** requires `node` 20 installed, either on the path or at `NODE_DIR` (an environment variable or `make` parameter).

Running with `make EXTRA_ARGS='--language LANG'` will allow you to load `LANG` exclusively, where `LANG` is one for the language ids/aliases defined in `lib/languages.ts`. For example, to only run **Compiler Explorer** with C++ support, you'd run `make EXTRA_ARGS='--language c++'`. The `Makefile` will automatically install all the third-party libraries needed to run; using `npm` to install server-side and client-side components.

For development, we suggest using `make dev` to enable some useful features, such as automatic reloading on file changes and shorter startup times.

You can also use `npm run dev` to run if `make dev` doesn't work on your machine.

Some languages need extra tools to demangle them, e.g. `rust`, `d`, or `haskell`. Such tools are kept separately in the tools repo.

Configuring compiler explorer is achieved via configuration files in the `etc/config` directory. Values are `key=value`. Options in a `{type}.local.properties` file (where `{type}` is `c++` or similar) override anything in the `{type}.defaults.properties` file. There is a `.gitignore` file to ignore `*.local.*` files, so these won't be checked into git, and you won't find yourself fighting with updated versions when you `git pull`. For more information see Adding a Compiler.

Check CONTRIBUTING.md for detailed information about how you can contribute to **Compiler Explorer**, and the docs folder for specific details regarding various things you might want to do, such as how to add new compilers or languages to the site.

Running a local instance

If you want to point it at your own GCC or similar binaries, either edit the `etc/config/LANG.defaults.properties` or else make a new one with the name `LANG.local.properties`, substituting `LANG` as needed. `*.local.properties` files have the highest priority when loading properties.

If you want to support multiple compilers and languages like godbolt.org, you can use the `bin/ce_install install compilers` command in the infra project to install all or some of the compilers. Compilers installed in this way can be loaded through the configuration in `etc/config/*.amazon.properties`. If you need to deploy in a completely offline environment, you may need to remove some parts of the configuration that are pulled from `www.godbolt.ms@443`.

When running in a corporate setting the URL shortening service can be replaced by an internal one if the default storage driver isn't appropriate for your environment. To do this, add a new module in `lib/shortener/myservice.js` and set the `urlShortenService` variable in configuration. This module should export a single function, see the `tinyurl` module for an example.

RESTful API

There's a simple restful API that can be used to do compiles to asm and to list compilers.

You can find the API documentation here.

Contact us

We run a Compiler Explorer Discord, which is a place to discuss using or developing Compiler Explorer. We also have a presence on the cplusplus Slack channel [#compiler_explorer](#) and we have a public mailing list.

There's a development channel on the discord, and also a development mailing list.

Feel free to raise an issue on github or email Matt directly for more help.

Official domains

Following are the official domains for Compiler Explorer:

- <https://godbolt.org/>
- <https://godbo.lt/>
- <https://compiler-explorer.com/>

The domains allow arbitrary subdomains, e.g., <https://foo.godbolt.org/>, which is convenient since each subdomain has an independent local state. Also, language subdomains such as <https://rust.compiler-explorer.com/> will load with that language already selected.

Credits

Compiler Explorer is maintained by the awesome people listed in the AUTHORS file.

We would like to thank the contributors listed in the CONTRIBUTORS file, who have helped shape **Compiler Explorer**.

We would also like to specially thank these people for their contributions to **Compiler Explorer**:

- Gabriel Devillers (*while working for Kalray*)
- Johan Engelen
- Joshua Sheard
- Andrew Pardoe

Many amazing sponsors, both individuals and companies, have helped fund and promote Compiler Explorer.