rochus-keller / Oberon Public

Code Issues 2 Pull requests 3 Discussions Actions Projects Wiki Security In

New issue Jump to bottom

License does not Allow Closed Source Development #41

Open martinvahi opened this issue 3 days ago · 5 comments

martinvahi commented 3 days ago • edited

GCC and other compiler projects tend to have a GPL exception that states that the stdlib of the language is not under GPL. stdlib tends to be linked to the generated binaries/generated_code. Without the possibility to use the compiler for closed source development the compiler will become useless for freelancers like me, who have to deliver code that my clients can use as a closed source project. For example, software components under the the GNU Lesser General License are usable by me, because the GNU Lesser General License allows my clients' projects to stay closed source and I only have to publish my modifications to the components that others offered as open source software.

I am not a hypocrite with my such wishes, because the most restrictive license that I use for my own, personal, projects is BSD license.

Thank You for reading my comment.

rochus-keller commented 3 days ago • edited Owner

I'm mostly using GPL for "traditional" reasons without thinking much, and I actualy don't care what people are using the code for. The runtime stuff of my Oberon compiler is available under LGPL or MPL (see e.g. https://github.com/rochus-keller/Oberon/blob/master/runtime/Out.c). With that option you can do even comfortable static linking.

martinvahi commented 3 days ago Author

2023_12_20_Rochus_Keller_Oberon_compiler_GPL_linking_exception_type_1.pdf

I'll interpret Your current statement here then as the classical GPL linking exeption like it is with the GCC, where
the GCC itself is under GPL, but the code that it generates or links during compilation is not under GPL. I'll use the PDF that I attached to my current comment as the linking exception document and consider that issue solved. Thank You for the answer and thank You for the software :-)

https://archive.ph/L5UF8

martinvahi closed this as completed 3 days ago

rochus-keller commented 3 days ago • edited Owner

Please note that I'm talking about the **runtime**, which is **the code you find in the runtime subdirectory**. It is this and only this code which is available under LGPL and MPL.

The compiler and IDE themselves are currently available under GPL which from my point of view doesn't restrict their use too much. Please check the license headers of the files you're interested in!

It is therefore **wrong** to interpret my statement as a general "linking exception" for the entire project.

> but the code that it generates or links during compilation is not under GPL

This is a misconception. Something that is generated by an algorithm (i.e. not created by a human being) is by definition not subject to (international and local) copyright law. Therefore, the output of the compiler is not GPL, even if the compiler itself is GPL. The "linking exception" has nothing to do with this; it would only be relevant if you intended to use the compiler source code in your own project; in this case GPL applies.

I hope this has made things clear. The documents that you produce for yourself are not suitable as a substitute for the license conditions under which I release the code.

martinvahi commented 3 days ago Author

Thank You for the answer, but isn't it so that a compiler always LINKS the code or binary that it generates to the stdlib parts of the programming language that is used for writing the compilable code the compiler is compiling/translating to a binary or some generated code? stdlib code is part of the compiler project and the compiler reuses code blocks of the stdlib or compiled binaries of the stdlib to generate its output. That is the reason, why the GCC uses the "linking exception".

The GCC "linking exception" does not mean that GCC project code that is outside of the stdlib of the C/C++/whatever_GCC_supports could be linked to proprietary programs, because that would already be reusing the GCC in an arbitrary way as if it were some BSD-licensed or MIT-licensed software.

> Something that is generated by an algorithm (i.e. not created by a human being) is by definition not subject to (international and local) copyright law. Therefore, the output of the compiler is not GPL, even if the compiler itself is GPL. The "linking exception" has nothing to do with this; it would only be relevant if you intended to use the compiler source code in your own project; in this case GPL applies.

If that were the case, then GCC would *not* need the "linking exception". I slightly repeat myself by writing here that the C/C++ stdlib IS PART OF THE COMPILER SOURCE CODE and IT GETS REUSED BY THE COMPILED BINARIES, which are separate projects from the compiler/GCC project. That is also the reason, why the ParaSail programming lanugage has the compiler under GPL, but ParaSail stdlib is allowed to be used in closed source programs. (Formally it was named "GCC Runtime Library Exception")

GCC_RUNTIME3_1.txt

Lawyers in court do not care, what somebody was thinking. They care about concrete licenses and any vagueness will be abused by whichever side that wants to abuse it. In my view Your last statement leaves things vague. I suggest that You write a clear statement to some LICENSE.txt that states clearly, whether closed source software can be compiled with the compiler without imposing any compiler related license to the compiler output AND THE STDLIB THAT THE COMPILER OUTPUT DEPENDS ON. Thank You.

martinvahi reopened this 3 days ago

rochus-keller commented 2 days ago • edited Owner

> isn't it so that a compiler always LINKS the code or binary that it generates to the stdlib parts of the programming language that is used for writing the compilable code the compiler is compiling/translating to a binary or some generated code?

There are also compilers where this is not the case. But in the present case, my compiler generates either ECMA 335 assemblies or C99 files, and adds either the DLLs or C files from the *runtime*. Since the latter are available under LGPL or MPL, you are allowed to both dynamically or statically link of these runtime files to your application.

The GCC toolchain is different in this respect, since also the GCC runtime libraries (e.g. libc) are GPL; therefore FSF issued the GPL runtime library exception, which allows you to dynamically or statically link to libc. I find it more convenient to licence the runtime library under LGPL or MPL. If you build the C99 code generated by my compiler using GCC, the GCC runtime library exception issued by FSF also applies to you (see https://www.gnu.org/licenses/gcc-exception-3.1.html). In contrast the ECMA 335 assemblies directly run on the Mono runtime, which is available under an MIT license.

> GCC project code that is outside of the stdlib of the C/C++/whatever_GCC_supports could be linked to proprietary programs, because that would already be reusing the GCC in an arbitrary way

That sounds like a misconception. If you generate C99 with my compiler, then the result is just another C project which you can compile with any C99 compatible compiler, and thus use the C runtime library und whatever license it is available. This doesn't violate e.g. the GCC runtime library exception in any way.

> If that were the case, then GCC would not need the "linking exception"

No, because you still link machine generated code to the C runtime library, which itself is "human generated" and available under GPL with the runtim library exception.

> In my view Your last statement leaves things vague.

No, and the legal situation is very clear. A judge would have no reason to come to a different conclusion.

> write a clear statement to some LICENSE.txt that states clearly, whether closed source software can be compiled with the compiler without imposing any compiler related license to the compiler output AND THE STDLIB THAT THE COMPILER OUTPUT DEPENDS ON

Everything has already been laid out. The Oberon+ compiler is available under GPL, which doesn't restrict the use of the compiler in a commercial project. The Oberon+ runtime library on which the code generated by the Oberon+ compiler depends is available under LGPL or MPL, which again allows dynamic or static linking also with commercially used applications. In case you generated C99 code using the Oberon+ compiler, this code can be compiled with e.g. CLANG or GCC and linked with their runtime library subject to their license or runtime library exception.

Hope this helps.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants