Clang C/C++ Language Working Group Meetings

Meeting Details

Shared Calendar
https://calendar.google.com/calendar/u/0?cid=cW1lZGg0ZXNpMnIyZDN2aTVydGVrdWF1YzRAZ3JvdXAuY2FsZW5kYXIuZ29vZ2xlLmNvbQ

Google Meet
Clang C/C++ Language Working Group Meeting
First and Third Wednesday of each Month · 11:00am – 12:00pm Eastern Time (UTC-5 when DST is off)
Google Meet joining info
Video call link: https://meet.google.com/ifi-uqto-kjw
References
Clang - C++ Programming Language Status
https://clang.llvm.org/cxx_dr_status.html
https://clang.llvm.org/c_status.html
https://clang.llvm.org/c_dr_status.html

Meeting Agenda - March 4th, 2026

Agenda:

- Next meeting will be an hour later for people in ~~blessed~~ non-DST timezones
- Clang 22 is out
- Upcoming WG14 meeting
    - Yeoul's "dependent attributes" are not on the agenda, neither are forward declarations of function parameters
    - Krause, Named Address Space Type Qualifiers for C2Y v3 [N3795]
    - Colomar, Adopt qualifier conversion rules from C++ [N3749]
- Do we want to deploy `-verify-directives` (PR) outside of C++ DR tests?
- Increasing number of "pure-AI" contributions from new contributors. Can we somehow stop wasting time on these? Maybe not allow new contributors use LLMs similar to not allowing using LLMs for good first issues
    - No assisted by label on these
    - Content is often not relevant
    - PR description is too long and does have Markdown that no human uses for commit messages
        - Sometimes they quote obsolete standards in a very unhelpful way (example)
    - https://discourse.llvm.org/t/concerns-about-low-quality-prs-being-merged-into-main/89748
    - Example https://github.com/llvm/llvm-project/pull/181579
- P4032R0 "Strong ordering for meta::info"

Meeting Agenda - February 18th, 2026

Agenda:

- Clang 22 is approaching RC4
  - 24th is deadline for RC4
  - Shafik: there are regressions linked to Matheus' PRs about resugaring
- Results are out for Clang Area Team 2026 elections
  - Less voters this time (thread)
  - First meeting this afternoon, in a few hours
  - Aaron, Eli, Corentin, Erich, and Shafik are members
  - Tags like `clang-area-team` point to the old CAT at the moment
  - Aaron: there are people who did not expect elections to be so soon, then there were wording issues with the titles of Discourse threads
  - Ambrose: I found out about it on the last day
  - Erich: doing over?
  - Aaron: no, we've got enough votes
  - Vlad: how this group will interact with CAT?
  - Aaron: I don't think anything will change. CAT is not as interested in language support. Extensions are an overlap, though
- WG14 meeting
  - Aaron: took place two weeks ago, when we were supposed to have the last meeting
  - Aaron: only 4-5 things voted in that are not issue resolutions
  - Aaron: nothing particularly contentious in the issue resolutions
  - Ambrose: is optional back?
  - Aaron: in a TS
  - Aaron: I'll probably write a paper that optional is not shippable even as a TS. sdcc decided to implement it, because they are the only who can get away with it. There are features that say that this pointer is not null. Then WG14 want to have a feature that pointer must not be null. Both markings are needed. Unless it can be incrementally adoptable, feature is not viable. This feature came up in our community first. We rejected, then the author went to WG14
  - Corentin: C thinks that everything has to be qualifier. _Atomic is bad enough, and I don't want more of that
  - Aaron: specifiers vs qualifiers doesn't bother me too much
  - Corentin: attribute would not pollute the type system
  - Erich: that attribute should be spelled "&"
  - Joshua: several of the committee members have issues with qualifiers and attributes, because they always want them to be droppable
  - Aaron: thankfully, optional was not discussed, but forward declared params were but did not get consensus
  - Vlad: not dead yet, he scheduled the paper and did not schedule the new revision of Yeoul's paper. So maybe coming up again.

- Aaron: Robert did not realize that Chris' paper had no new information, but made it the deadline for February
- Aaron: next meeting is scheduled on March
- Aaron: committee doesn't seem to be interested in a bunch of new helper functions for standard library. Committee's appetite for new features seems to go down
- Corentin: I"m not sure those helper functions help anything
- Aaron: old WG14 members chatted on mastodon, expressing their dissatisfaction with WG14 this cycle
- Aaron: I'm having 1-on-1 conversations to mend bridges between implementers and non-implementers. Hope is that couple of non-implementers will understand implementation concerns, reducing the contention in the committee
- Aaron: I'm having monthly meetings with Martin Uecker, who occasionally contributes to GCC, but otherwise works in biomedicine domain. Hope he'll socialize some of our problems. At some point if those 1-on-1 go well, I'll invite those folks to this meeting for extra perspective.
- Vlad: lamda discussion was a disaster
- Aaron: JeanHeyd came with a paper describing the design space, but committee was like "tl;dr". Immediately after we discussed other papers which went for half-solutions, but no decisions were made
- Joshua: JeanHeyd's paper was interrupted at the end, but had all the information.
- Aaron: I think there's misunderstanding between you and Martin, because he moved on from trampolines
- Joshua: even if you don't propose trampolines, but the way feature works, trampolines will be needed
- Corentin: is there an implementation of nested functions without trampolines?
- Aaron: opt-in, because of ABI
- Corentin: I talked to people at apple, and they are very concerned about security
- Corentin: users will be very confused if lambdas will come without captures
- Aaron: regex for lambdas is hard, but it seems that there are 3 times more lambdas with captures than lambdas without them.
- Aaron: C doesn't really need anything in this space. Rushing the solution is the real problem here
- Vlad: for context, some folks in WG14 think lambdas are perfectly viable without captures
- Aaron: any more questions about WG14?
- Ambrose: any updates on defer TS?
- Aaron: it wasn't on the agenda, but I'll make sure it is on the next one
- Aaron: JeanHeyd gave WG14 update on implementation experience. There was no reactions from WG14
- Corentin: have we got any feedback from our users?
- Aaron: no, because 22 is not out yet. I don't expect any bug reports, because it's just a syntax for attribute cleanup

    - Aaron: I have not heard about GCC implementation of defer. I know there's a patch under review, but it might be too late for stage 4
    - Corentin: reflection was pushed at stage 4 of GCC 16 release
- Remove 80 column limit in documentation files (RFC)
    - Joel created a script to review PRs via git diff (post)
    - Aaron: there is a poll. Please vote
    - Aaron: 80 column limit doesn't help in documentation
    - Vlad: there is an illustration in the thread
    - Ariel: back in the day there was a latex guideline to start every sentence with a new line
    - Aaron: this is where this option on the poll came from
    - Vlad: Mehdi is asking for tooling to enforce one sentence per line
    - Aaron: it should be trivial to implement
    - Corentin: some sentences might be very long
    - Vlad: we're not good at upholding the limit
    - Shafik: I changed my opinion after your office hours to one that is close to yours
    - Erich: will this be discussed in CAT meeting today?
    - Aaron: yes, but it's too early to call consensus on that
    - Hubert: I might be voting for a less popular options
    - Aaron: that's the beauty of it: everyone will be dissatisfied
- Hubert: custom clang-format configs per subproject
    - Hubert: I think they should die, because some of them violate our coding standards with regards to includes
    - Aaron: theoretically, every subprojects has to follow the same coding standards. Practically, this never was the case. MLIR picked their own coding standard, which causes friction now that ClangIR is integrated into Clang
    - Aaron: if our clang-format violates something, we should change it
    - Ambrose: can be disable clang-format for tablegen files?
    - Ambrose: adding //clang-format off at the top of every tablegen file might not be a bad idea
    - Aaron: for the long time we resisted tool-specific annotations
    - Ambrose: I agree on principle, but clang-format is the only tool we have this issue with
    - Corentin: we can disable it per directory https://clang.llvm.org/docs/ClangFormat.html#clang-format-ignore
    - Aaron: this has been annoying for the long time, would be nice to fix that
    - Ambrose: it's annoying for reviewers, too
    - Aaron: I feel terrible to tell people to ignore clang-format
    - Aaron: Hubert, since you're looking at this, would you look at td files?
    - Hubert: I could, but it can't be the same PR
    - Aaron: **I'll take an action item**
- Do we want to deploy `-verify-directives` (PR) outside of C++ DR tests?
- User branches without a PR will soon be deleted (thread)

    - Originally, user-created branches were expected to be deleted, but that work did not happen
    - Aiden is now starting to tackle this; will prune branches on a daily basis based on whether there's a public PR tied to the branch. They will archive the branches when deleting them so data is not lost.
    - Ariel: what's the point of the one-day delay?
    - Vlad: to avoid timing issues between creating the branch before the PR is created; having the delay makes that less likely
- Long-term vision for improving build times (Meta-RFC)
    - Some desire to make clang do more build system work; perhaps can do better caching or better module dependencies, use vfs instead of expensive windows file system calls, avoid process creation overhead, etc.
    - Very large RFC, but author has some short term goals
    - Shafik: this came up during the LLVM social, there was skepticism about this being viable. The biggest beneficiary seems to be Windows but there's quite a bit of risk for different ways to break things. It may be too ambitious, but a more narrow focus may gain traction. Questions of "who will maintain this long term"?
    - Reid: Previously worked with the author, talked in other contexts before. I think the project would benefit from a lot of the short-term goals. Already have to patch LLVM to avoid conflicts with global state (within NVidia). I don't see Windows as being someone else's problem; dylibs on linux are another example. Process startup times can be slow everywhere. Removing extra process launch time from RUN launching a custom shell still saves like 10% of time. Repitching the RFC with a narrow set of goals could be successful.
    - Hubert: Reid got to most of what I was thinking; I think this could be helpful for more than just Windows.
    -
-
- Next agenda:
- Do we want to deploy `-verify-directives` (PR) outside of C++ DR tests?
- Increasing number of "pure-AI" contributions from new contributors. Can we somehow stop wasting time on these? Maybe not allow new contributors use LLMs similar to not allowing using LLMs for good first issues
    - No assisted by label on these
    - Content is often not relevant
    - PR description is too long and does have Markdown that no human uses for commit messages
        - Sometimes they quote obsolete standards in a very unhelpful way (example)
    - https://discourse.llvm.org/t/concerns-about-low-quality-prs-being-merged-into-main/89748
    - Example https://github.com/llvm/llvm-project/pull/181579
-
    -

Meeting Agenda - February 4th, 2026

Agenda

- Historical notes removed from this document and archived on Discourse
    - https://discourse.llvm.org/t/historical-clang-language-wg-meeting-minutes-mar-2025-jun-2025/89546
    - https://discourse.llvm.org/t/historical-clang-language-wg-meeting-minutes-jun-2025-sep-2025/89547
    - https://discourse.llvm.org/t/historical-clang-language-wg-meeting-minutes-sep-2025-dec-2025/89548
    - Required three separate posts due to character count limits. We talk a lot. ;-)

Meeting Agenda - January 21st, 2026

Agenda

- rc1 is out, rc2 is expected next Tuesday (27th). Final release is expected Feb 24
    - Aaron: if there are regressions, they should be prioritized, especially regressions introduced in 22
    - Aaron: release coincides with two WG14 meetings: one in february, one in march right after the final
- WG14 (virtual) meetings happening Feb 2-6 and Mar 9-13
    - Preliminary agenda: https://www.open-std.org/jtc1/sc22/wg14/www/docs/n3790.pdf
    - Quite a few pure inventions (N3694, N3678, N3679, N3674, N3454, N3692, N3458), if you see papers you want me to express feedback on, please let me know (on any paper, not just these)
    - Aaron: my availability is going to be random. Has to read papers and be present in the meetings. But meetings are just half a day.
    - Aaron: preliminary agenda for WG14 meeting is out.
    - Aaron: if you have feedback on papers, let me know
    - Aaron: pure inventions make the most sense to review
    - Aaron: there is a big block of papers about lambdas-like facilities
    - Ariel: what do you mean by fat function pointers
    - Aaron: the way to add more information to functions
    - Corentin: would be nice to have a pointer type
    - Aaron: there are bugfixes for C23 features, like removing UB when no one is exploiting it
    - Hubert: papers are poorly names. UB is a symptom how C writes wording
    - Aaron: UB SG is getting rid of any stupid UB, leaving only meaningful UB. paper titles are indeed are not super helpful
    - Ariel: are fat pointer going to have a new linkage?
    - Ariel: linkage describes how functions are called
    - Hubert: they are more like function descriptors, holding additional information to establish the context, more like a closure
    - Aaron: that's all for WG14 meetings. If you have questions, reach out to me
    - Aaron: if I see something that desperately needs attention, we have a clang wg meeting right in the middle of WG14 meeting
    - Corentin: if any UB papers touch lexing or preprocessor, let me know. In the past C made changes that are incompatible with C++
    - Aaron: looking through the list. So far not seeing anything that matters. THere is a paper that refactors grammar for preprocessor directives
    - Vlad: Alejandro brought this paper in Core. JEns was not impressed with the formatting
    - Hubert: just for the formatting. We're going to apply it to C++, too
    - Aaron: I see your papers on the list
    - Corentin: I'll be there

    - Hubert: my issue with that paper is how POSIX people are going to deal with it, like on fork().
    - Aaron: you guys can talk about this offline. THere are folks from POSIX in the meetings, so they should be represented
    - Ariel: if fat pointer is a funticon descriptor, why this has to be a language feature?
    - Aaron: trying to do what lambdas do in C++. Closures are needed. Two competing approaches are C++ lambdas and GCC nested functions. There is a a third approach
    - Hubert: block?
    - Aaron: surprisingly, they have not been proposed
    - Corentin: we still have implementation objections to GCC implementation of nested functions
    - Aaron: yes, trampolines and executable stack
    - Hubert: since you've came up with this list. There is a paper about language linkage blocks. Why is it problematic?
    - Aaron: there have been disagreements about what those blocks mean
    - Aaron: if I have extern block, I should be able to use static on array extents — kind of confusions people have
    - Aaron: so this is inventive. A lot of inventive stuff come from people without implementations, out of pure enthusiasm, but it costs us money.
- Implementation reality of WG21 standardization P3962R0
    - Hubert: there is WG21 paper about implementer feedback
    - Hubert: there is new version of P1000, which is IS schedule. Wonder how those papers will be discussed
    - Aaron: at the Kone WG21 meeting, there was an evening sessions for people who self-identify as implementers, both for compilers and library. The output of P3962R0, which is a good distillation of the discussion. There is a lot of concern from all implementers how expensive it is to implement the language. A lot of experiments which we need to pay for. We need deployment experience, not just implementation in a fork. Paper is a list of requests to the committee. Like CWG and EWG running simultaneously means that implementers has to choose where to be. I don't know where the paper is going and what the impact is going to be, but I'm happy to see we're not alone thinking that LLVM is drowning in work
    - Shafik: there was also discussion about last C++ version an implementation is going to support. Louis had an idea for a paper to make C++29 a bugfix release.
    - Aaron: I've taking a liberty of alerting WG14 convenor of this paper. We have the similar kind of concern for WG14. Not sure what they will going to do, but would be nice to bring this up in front of the WG14, even if C features are not as expansive as in C++. There is a proposal to bring contracts, but C++ committee itself doesn't understand the feature
    - Hubert: there are people who understand, but (even within that subset) they do not agree whether trade-offs are right
    - Aaron: Jens is aware of C. contracts with come up in headers, so it's a compatibility concern

    - Corentin: I'm sad I didn't put my name on the C++ paper. Was busy with other stuff
    - Hubert:
    - Shafik: it was a therapy session
    - Corentin: impelementations do not agree on everything, but this is fine
    - Aaron: speaking of relfection, GCC put reflection in stage 4, even though stage 3 was not supposed to get new features. Wonder how it goes for them
    - Aaron: this gives us something to test against, at least
    - Michael: they are getting bug reports for it (https://gcc.gnu.org/bugzilla/buglist.cgi?quicksearch=reflection)
    - Aaron: I don't have anything else on P3962R0
    - Hubert: Nina was focusing on things we have solutions for, not everything we discussed.
    - Vlad: Nina also did not put stuff that had solutions, like tests with features
    - Hubert: yeah, but we should not dilute the existing points
    - Vlad: that's why I argued that the message should be that this is just the beginning of our complaints
    - Shafik: I also tried to get it into the paper
    - Aaron: committee doesn't want to compete with its members, but we need the tests
    - Shafik: this bites it over and over again
    - Aaron: if WG21 say "we can't", but we still need this, then we need to collaborate with other implementers on a shared test suite.
    - Shafik: there might be licensing features
    - Aaron: we could've put new tests open source
    - Aaron: we've spent 300 hours on reflection, but didn't get tests out of it
    - Vlad: I've asked Dan several times to put test cases, and it's under our license
- Maintainer List Refresh
    - Verified 32 maintainers in Clang, still waiting to hear from 13
    - If anyone is interested in stepping up as a maintainer, please talk to me about it, it's easy and fun and you'll never regret volunteering to do more work!
    - Aaron: I reached to people privately, which turned to be blessing and a curse. Some had wrong emails, which is now fixed. But 30% did not respond.
    - Aaron: most still continue being maintainers, some step down. We're having additional categories, like coverage. Corentin volunteered to maintain concepts. If you want to be a maintainer of a new area, let me know. If you know someone is doing this work without recognition, nominate them after speaking to them about it. I'm also doing refresh in clang-tools-extra, but it goes slowly. Searching for lead maintainers for projects that do not have such. Found maintainers for libclc,libsycl, libcxxabi, openmp. No lead maintainer for lld, mlir. THere are active maintainers, but sometimes projects do not lend themselves to a position of lead maintainer. I'll work with project council to figure this out
    - Matheus: elf and coff linker used to be entirely separate path, down to copying parts of implementation

    - Aaron: yes, they are three linkers there
    - Aaron: there are times when we want more than one contact point
    - Matheus: my employer is sponsoring students to work on clang, like GSoC.
    - Matheus: for example SARIF. Is someone working on it?
    - Aaron: I did the initial work to bring it for clang sa
    - Aaron: production quality? Maybe, maybe not
    - Matheus: for the compiler, not just static analyzer
    - Aaron: sarif output is getting used, static analyzers integrated clang's sarif output, like msvc
    - Aaron: at least you can get diagnostic fidelity out of it. SARIF lets you have code path inside diagnostic, insights into analysis-based diagnostics
    - Aaron: it's easy to make improvements, because it's already there
    - Ambrose: https://github.com/llvm/llvm-project/pull/174106
    - Matheus: the main thing is to put proposals like GSoC
    - Aaron: if SARIF is important for your employer, we can set up a maintainer for it. Let's pretend you are maintainer. It's easier for you to work with lots of students when you have experience
    - Matheus: #embed is also interesting. Anyone working on it?
    - Aaron: Maria is working on it.
    - Aaron: if you're looking for topics, improve compile times
    - Matheus: I'm working on this
    - Matheus: we're looking more into GSoC level of things
    - Michael: there's a bunch of people in bloomberg who want to look into this. Matheus, we should chat about collaborating
    - Aaron: I can help coordinating
    - Aaron: we have 5 minutes left
    - Hubert: refresh minutes and archive minutes from 2025
    - Hubert: there is a small PR (https://github.com/llvm/llvm-project/pull/176551). Are we going in the right direction? WG14 has a thread about this. People sometime need this without the full breadth of MS extensions. Cc1-only option is a way
    - Aaron: I was planning to review it this week
    - Aaron: this should be driver flag, like "is char signed"
    - Aaron: I can see this useful enough that users would want to control this separately
    - Vlad: we shouldn't do this as cc1-flag
    - Aaron: my understanding is that it's for downstream, not directly exposed to users
    - Hubert: there are going to be users who won't complain about -Xclang
    - Aaron: I'll post on Discourse if something changes for the next meeting.
- AI policy has landed.
    - Aaron is excited. Good incremental progress, but is not going to move the needle for quality of submissions
      Shafik: agreed
    - Hubert: hope it moves the needle

- Aaron: it's awkward to assert that something is extractive
- Ariel: what is extractive?
- Aaron: asking community to do more than you did
- Corentin: not related to clang, but there is an RFC about using AI for Bazel, because it goes into the opposite direction.
- Aaron: yes, I hope we don't make exceptions. Peripheral tier should not get exceptions. But I don't think Bazel should exist, so I might be biased. There is a burden on the community to maintain it, which RFC now tries to address
- Hubert: (muses that the Bazel proposal is drawing lots of energy because some people are afraid of a "slippery slope", but personal not sure it is that dangerous)
- Aaron: we'll iterate on the policy as we learn. It's still good incremental progress
- Hubert: I followed the AI policy thread. I was not aware that ms178, the person who did AI stuff in mesa, also did AI stuff in LLVM

Meeting Agenda - January 7th, 2026

Agenda

- Vlad: reflection meetings start on Jan 9th on a two week cadence (post)
- Aaron: Clang 22 branch date is Jan 13th (next Tuesday)
    - There are discussions about switching to April/October releases but not for 22
    - Speak to release managers if interested
    - Current thread that started the discussion: https://discourse.llvm.org/t/rfc-freeze-the-llvm-library-abi-with-rc1-starting-with-llvm-23/89373
- Vlad: “Implement P1857R3 Modules Dependency Discovery” (PR)
    - Relands 1, 2
    - Taken a couple of tries to get this landed, still not quite ready. Impacted downstreams quite a bit.
    - Probably not making it to Clang 22
    - There was a maintainer suggesting that the author land the changes due to the PR being open for a while.
    - Still would have been better to hold off until there were more eyeballs
- Aaron: “Not assuming there is at most one definition in a redeclaration chain” (RFC)
    - Aaron: hope we’re not landing this in 22, left some comments
    - Aaron: seen overflow behavior discussions, but they are not targeting 22
    - Aaron: going to prioritize recent regressions
    - Aaron: if you see issues after branches, please prioritize those recent regressions
    - Aaron: this RFC is a heads-up for both upstream and downstreams
    - Aaron: for the longest time we’ve been assuming there’s only one definition in the declaration chain, but this is not true in both C and C++
    - Aaron: we’ve been fixing this here and there while working on modules
    - Aaron: can have impact on downstreams and plugins
    - Ariel: multiple definitions?
    - Aaron: yes
    - Hubert: is this only AST level?
    - Aaron: I think so, should not have backend impact
    - Vlad: C++ is modules, why is this problem in C?
      Aaron: in C this is only for a type. We have a rule type compatibility rule between TUs, now we have it intra-TU for type definitions.
    - Hubert: C scoping rules for types are different from C++.
    - Aaron: in C you can define a type everywhere you can name a type, which is not case in C++, e.g. in a compound literal. Change in C made it less surprising for macro users
- Hubert: is there a thread on Discourse about release dates that I can follow?
- Aaron: yes: thread
- Shafik: I’d like to complain about a spree of fuzzing issues
- Shafik: there is a particular user who filed low-quality issues

- Shafik: npopov handled him
- Shafik: when I see a new user filing fuzzing issues, I explain to them how to do that in a useful manner. It takes time, I'd like to see an official policy
- Aaron: <talking while on mute>
- Aaron: it'd be good to have a policy
- Aaron: in order to keep the policy welcoming, ask people to talk to us on discord before starting
- Shafik: that would be nice
- Shafik: my manager is asking whether this is university work, companies or just individuals
- Aaron: then we can guide them before everyone is frustrated. But of course not everybody read policies
- Shafik: at least we don't need to explain the same thing over and over again
- Aaron: you've got a hold of that. I was keeping an eye on your interactions on the bug tracker
- Round table:
    - Aaron
        - Getting back, digging out
        - Catching up on code reviews and mentions
        - If you need my attention for priority, poke me manually
        - Administrative stuff: reflection meeting, project council meeting, CoC meeting
    - Ariel
        - Hexfloat, hoping to upstream shortly
        - Aaron: that's not the parsing part, right?
        - Ariel: different representation
        - Ariel: there's going to be a PR to support hexfloat in APFloat, then another PR to support hexfloat in frontend
        - Aaron: I see
    - Hubert
        - IBM is actively working -finput-charset and -fexec-charset
        - For -finput-charset, I'd like to confirm something with the group.
        - z/OS requires to have pragmas to specify code pages
        - Encountered a lot of troubles with source location and diagnostic infrastructure
        - It feels that adding a layer of indirection to that infrastructure is going to kill performance. Is that correct?
        - Aaron: yes
        - Hubert: we need to reconcile z/OS and non-z/OS approaches
        - Aaron: terrible idea: pre-preprocessing for those code page pragmas
        - Hubert: you could mean two things. First approach needs to be a real pre-preprocessor
        - Aaron: yes, that is what I had in mind

- Hubert: we have people internally who believe this is the way. Otherwise, if we do this in the actual preprocessor, sourcemanager already has a concept of source locations that is not compatible
- Aaron: you could support pragma downstream
- Hubert: interesting idea. Not sure what z/OS stakeholders think of this. If we could isolate it properly, it might be fine to upstream. Depends on the shape of things. Don't want to start with a downstream-first approach
- Ariel: I think there's the case to upstream this. Users want to adapt upstream to z/OS
- Aaron: my gut feeling is that pragma is so weird that upstream will not be comfortable. Needs explaining how prevalent the use case is. Rajan explained to me that this can appear in system headers
- Hubert: yes, if you use national characters in your source, you'd have to use the pragma
-
- Hubert: -fexec-charset has issues with formatted output
- Hubert: one has to do with the source locations and diagnostics. Can't easily map source location to the souce file when -fexec-charset is used
- Hubert: there's divergence between implementations on conversion specifiers when multi-byte encodings are involved. When you hit percent sign, you drop down to code points, then go back to multi-byte. C's concept for string might not be adequate and what users expect. Linux seems to work fine.
- Hubert: practical effect is that I don't think we want multiple string interpretations.
- Aaron: we can follow GCC behavior and try to defend it
- Hubert: GCC has bugs they haven't fixed in decades
- Aaron: when did the C committee doubled down on the current approach?
- Hubert: before C23
- Aaron: because there's divergence on the platform level, it's not necessarily on implementation level.
- Hubert: I don't think disagreement was intentional
- Aaron: is this an issue at the OS level?
- Hubert: yes. There's a limited number of C libraries that support locales but don't deal in UTF-8
- Aaron: we might need a paper saying that the resolution to that DR is not implementable
- Hubert: but C libraries implement the C standard
- Hubert: OS folks need to come to the WG14 and defend what they currently do. We need to double down on UB
- Aaron: committee is not going to like that. Llvm-libc folks are starting to attend WG14 meeting, but they are not going to support locales
- Hubert: UTF-8 makes it hard to negotiate

        - Hubert: the clarification that WG14 made is that character means byte, which doesn't work with UTF-8
        - Aaron: let me know if I can help with WG14
        - Hubert: when we know what we're going to do, I'll reach out to you, because we have more than one issue
        - Aaron: not sure what the community thinks. Too esoteric
    - Islam
    - Mariya
        - Got back after holidays
        - Digging out of emails
        - Want to get back to fixing static variables in consteval functions
    - Nikhil
    - Shafik
        - Fuzzing
        - Soft-off: screening issues
        - Dealing with static analysis internally
        - We should be following the rule of three, because RAII wrappers don't need copy ctors and copy assignment operators
        - Does this need an RFC?
        - Aaron: RFC makes sense, because there are rule of zero and rule of five, but hopefully not contentious
    - Sharath
        - Don't have anything to report
        - Any guides to get started?
        - Aaron: welcome! If you're on discord, go to #clang and look at the pinned message
        - Sharath: I'll go through it and ping you
        - Aaron: works for me
    - Tony
        - Continue to work on fixing attributes using auto
        - Triaged and found around 6 attributes that need to be fixed
        - As reported in the issues about cleanup attribute, I fixed something for objc
        - https://github.com/llvm/llvm-project/pull/164440
    - Vlad
        - Mostly off
        - Reviewed couple of PRs related to python bindings
        - Don't feel comfortable reviewing things related to packaging and deployment, could use someone with expertise to help
        - Aaron: I'll see if I can find someone with expertise to help