Conclusion

- C++ needs a way to identify “isolation boundaries”
  - I.e. `send`

- This introduces strong aliasing and lifetime requirements

- This is not compatible with existing pointers/references

- Reflection can help us write in the styles of other languages which have better thread safety
  - Safely encapsulates pointers

- For “C++ performance” and “Don’t pay for what you don’t use” we need borrow checking:
  - Sean Baxter: “Safe C++” wg21.link/P3390

123

Conclusion

- C++ needs a way to identify “isolation boundaries”
  - I.e. `send`
- This introduces strong aliasing and lifetime requirements
- This is not compatible with existing pointers/references
- Reflection can help us write in the styles of other languages which have better thread safety
  - Safely encapsulates pointers
- For “C++ performance” and “Don’t pay for what you don’t use” we need borrow checking:
  - Sean Baxter: “Safe C++” wg21.link/P3390

What Can C++ Learn About Thread Safety From Other Languages

David Rowland

@drowaudio

Questions?

Slides/video:

drowaudio.github.io/presentations