## co\_await All The Things!

Engineering

ACCU Conference 2023 2023-04-22

Dietmar Kühl

dkuhl@bloomberg.net

TechAtBloomberg.com

© 2023 Bloomberg Finance L.P. All rights reserved.

# Coroutine Support

- All major compilers ship an implementation: gcc, clang, MSVC++
- Sadly, the C++20 standard library provides no coroutine classes.
- The C++23 standard library only provides a generator
- There is no task or advanced support



## What Could Be co\_await'ed?

- Async operations: I/O, waiting for another thread
- External events: user interaction, other programs
- Unavailable resources: memory, file descriptors
- Reordering of data: batching requests
- Delayed access: advise data to be used soon





# Key Concepts

- Awaiter: specifying how async work is executed
- Promise type: specifying how a coroutine operates





# value = co\_await expression;

```
auto awaiter = operator co_await(expression);

if (awaiter.await_ready()) {
    awaiter.await_suspend(handle-to-coroutine);
    <resume here when handle-to-routine.resume() is called>
}
```

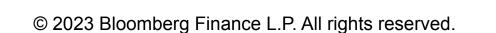
value = awaiter.await\_resume();



# Awaiter Type

- await\_ready(): telling the system whether a suspend is needed.
- await\_suspend(std::coroutine\_handle<> h):
  - Set up something to know to resume the coroutine
  - Arrange for the work to eventually complete (call h.resume())
- await\_resume(): produce the awaited result





TechAtBloomberg.com

```
Cf() { ... co_* ... }
```

```
auto* f = new frame<C::promise_type>();
C rc = f->promise.get_return_object();
invoke([&]{ try {
   co_await f->promise.initial_suspend();
   { ... co_*... }
   co_await f->promise.final_suspend();
  } catch (...) { f->promise.unhandled_exception(); }
return rc;
```

#### TechAtBloomberg.com

Bloomberg



# Promise Type

- initial\_suspend(): how to start a coroutine
- final\_suspend(): how to end a coroutine
- unhandled\_exception(): what to do with escaped exceptions
- get\_return\_object(): how to build the result of the factory function



#### Demo Time

#### Resources

- Lewis Baker's Asymmetric Transfer https://lewissbaker.github.io/
- Gor Nishov's talk on Nano Coroutines <a href="https://www.youtube.com/watch?v=j9tlJAqMV7U">https://www.youtube.com/watch?v=j9tlJAqMV7U</a>
- Code for this presentation <a href="https://github.com/dietmarkuehl/co-await-all-the-things.git">https://github.com/dietmarkuehl/co-await-all-the-things.git</a>

