

co_await All The Things!

ACCU Conference 2023
2023-04-22

Dietmar Kühl

dkuhl@bloomberg.net

TechAtBloomberg.com

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

Engineering

Bloomberg

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;
```

```
autoawaiter = 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



```
C f() { ... 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;
```


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

A decorative graphic in the top right corner of the slide, consisting of a dense, fan-shaped cluster of small, multi-colored dots (red, blue, green, and purple) that tapers towards the top left.

Demo Time

Resources

- Lewis Baker's Asymmetric Transfer
<https://lewissbaker.github.io/>
- Gor Nishov's talk on Nano Coroutines
<https://www.youtube.com/watch?v=j9tIJAqMV7U>
- Code for this presentation
https://github.com/dietmarkuehl/co_wait-all-the-things.git