In C++/WinRT, how can I await multiple coroutines and capture the results?, part 0

devblogs.microsoft.com/oldnewthing/20250730-00/?p=111421

July 30, 2025

A while ago, I investigated ways of awaiting multiple C++/WinRT corotuines and capturing the results. I started with `winrt::when_all()`, but really I could have started even earlier.

The easy way to await multiple coroutines is to await each one!

```
auto op1 = DoSomething1Async();
auto op2 = DoSomething2Async();

auto result1 = co_await op1;
auto result2 = co_await op2;
```

This starts two operations, and only after they have started do we wait for them in sequence.

This works great, as long as you understand that if an exception occurs in `co_await op1`, then the second operation `op2` just keeps on running and eventually has nobody to report results to.

This is probably okay for the most part: The second operation just keeps on running pointlessly. Not necessarily the most efficient, but probably not harmful.

If you want to cancel the second operation if the first operation fails, you can do that:

```
auto op1 = DoSomething1Async();
auto op2 = DoSomething2Async();

auto op2cancel = wil::scope_exit([&] {
    op2.Cancel();
});
auto result1 = co_await op1;
op2cancel.release(); // don't cancel it after all
auto result2 = co_await op2;
```

Note also that in the case where `op2` finishes before `op1`, this code fragment doesn't learn about it right away because it's busy waiting for `op1`. Again, this is not normally a problem.

So if all you care about is getting both results, and you’re not too worried about what happens if one of them fails, you can just `co_await` each one in turn. Pretty simple and straightforward. All the extra complexity I wrote about is trying to solve a problem you don’t care about.