Subject: Re: Coroutines package for U++
Posted by Oblivion on Sun, 06 Nov 2022 20:31:00 GMT
View Forum Message <> Reply to Message

Hello Klugier,

Quote:Hello Oblvion,

Why you decieded to create separate package in concuret to Core? Is there Co* family functions like CoWork, AsyncWork, CoPartition and CoDo not enough for your needs?

On the contary, I use them daily. :)

However, It seems that there is a misunderstading here.

C++20 coroutines are not "concurrency" functions. They are not threads. They are suspendable functions, meaning that you can return from the function in the middle of its operation and you can resume it later. This makes async programming without threads impressively easy!

Excerpt:

A coroutine is a function that can suspend execution to be resumed later. Coroutines are stackless: they suspend execution by returning to the caller and the data that is required to resume execution is stored separately from the stack. This allows for sequential code that executes asynchronously (e.g. to handle non-blocking I/O without explicit callbacks), and also supports algorithms on lazy-computed infinite sequences and other uses.

While it is one of the coolest features of C++20, the original design is, well, let's say it isn't very elegant... (no wonder even seasoned developers are having a hard time understanding how to use them - watch the cppcon's ever-increasing videos about them. :))

So, I haven't created a "yet another" thread-based concurrency tool, in the traditional sense. I have implemented and made available something completely new to U++. (hence the upload.)

Best regards,
Oblivion