Subject: Re: AsyncWork,  IsFinished() may not be working properly
Posted by Oblivion on Sat, 18 Jul 2020 07:59:07 GMT
View Forum Message <> Reply to Message

Hello JeyCi,

And welcome to the U++ forums!

Note that when the worker is removed from the array in the first example, the item count is also decreased by 1.
This obviously invalidates the indices, so the code breaks the for loop and starts it all over again (from 0).
Hence the "two or more zeros" in the log output. It might give the impression of race condition, but it is not. (If that's the problem.)

(The reason is that the first example was about a problem in IsFinished(). It was always returning true.
The example did not rely on those indices for its purpose, so I didn't care to make it more correct...)

Please try this one instead:

```
CONSOLE_APP_MAIN
{
 StdLogSetup(LOG_FILE|LOG_COUT);

 Array<AsyncWork<int>> workers;

 for(int i = 0; i < 4; i++) {
  workers.Add() = Async([=]{ LOG("Started, worker #" << i); Sleep(500); return i; });
 }

 while(!workers.IsEmpty())
  for(int i = 0; i < workers.GetCount(); i++) {
   auto& w = workers[i];
   if(w.IsFinished()) {
    LOG("Stopped, worker #" << w.Get());
    workers.Remove(i);
    break;
   }
  }
}
```

Quote:
I found below code seems working good

Yes, that works, because, unlike the "non-blokcking" first example, this one has a "blocking" loop:

```
 for (auto &fut : workers) {
  LOG("Stopped, worker #" << fut.Get());
 }
```

The Get() method will wait the worker to finish its job unless it is alreay finished. So it is "blocking".

Best regards,
Oblivion