
Subject: CoWork usage question

Posted by [Tom1](#) on Tue, 22 Jan 2019 13:59:18 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi,

As I need more than coffee to solve this out, I decided to put it here :)

The question is: What am I doing wrong, as the following code does not work as I expect, i.e. multiply the items by two and show the results:

```
#include <Core/Core.h>
```

```
using namespace Upp;
```

```
#define ICOUNT 15
```

```
class A{
```

```
public:
```

```
    Buffer<int> ib;
```

```
    A(){
```

```
        ib.Alloc(ICOUNT,0);
```

```
        for(int i=0;i<ICOUNT;i++) ib[i]=i+1;
```

```
    }
```

```
    void operation(int &x){
```

```
        x*=2;
```

```
    }
```

```
    void operationp(int *x){
```

```
        *x*=2;
```

```
    }
```

```
    void Run(){
```

```
        Cout() << "ib = ";
```

```
        for(int i=0;i<ICOUNT;i++) Cout() << ib[i] << " , ";
```

```
        CoWork co;
```

```
        for(int i=0;i<ICOUNT;i++){
```

```
            //co & [&] { operation(ib[i]); }; // Reference variant
```

```
            co & [&] { operationp(&ib[i]); }; // Pointer variant
```

```
        }
```

```
        co.Finish();
```

```
        Sleep(400);
```

```
        Cout() << "\n\nib*2 = ";
```

```
        for(int i=0;i<ICOUNT;i++) Cout() << ib[i] << " , ";
```

```
        Cout() << "\n\n";
```

```
    }
```

```
};
```

```
CONSOLE_APP_MAIN
{
  A ac;
  ac.Run();
}
```

Mostly the results are 'exciting' to say the least.

Best regards,

Tom

Subject: Re: CoWork usage question
Posted by [Tom1](#) on Tue, 22 Jan 2019 14:15:40 GMT
[View Forum Message](#) <> [Reply to Message](#)

Well, the second cup of coffee solved it for me:

```
CoWork co;
for(int i=0;i<ICOUNT;i++){
  //co & [&, i] { operation(ib[i]); }; // Reference variant
  co & [&, i] { operationp(&ib[i]); }; // Pointer variant
}
co.Finish();
```

Must have 'copy capture' for i, or otherwise the index i will run away before used.

Sorry for bothering with such trivial issue.

Best regards,

Tom

Subject: Re: CoWork usage question
Posted by [mirek](#) on Wed, 23 Jan 2019 14:32:57 GMT
[View Forum Message](#) <> [Reply to Message](#)

Tom1 wrote on Tue, 22 January 2019 15:15Well, the second cup of coffee solved it for me:

```
CoWork co;
for(int i=0;i<ICOUNT;i++){
  //co & [&, i] { operation(ib[i]); }; // Reference variant
  co & [&, i] { operationp(&ib[i]); }; // Pointer variant
```

```
}  
co.Finish();
```

Must have 'copy capture' for i, or otherwise the index i will run away before used.

Sorry for bothering with such trivial issue.

Best regards,

Tom

Yeah, got caught there too...

BTW, have you noticed the looper variant?

```
CoWork co;  
co & [&] {  
    for(;;) {  
        int i = co.GetNext();  
        if(i >= ICOUNT) break;  
        operationp(&ib[i]);  
    }  
}
```

It is faster and more resilient w.r.t. those issues.

Mirek

Subject: Re: CoWork usage question
Posted by [Tom1](#) on Thu, 24 Jan 2019 08:35:54 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Mirek,

Well, that was interesting. I did some benchmarking and the looper is indeed a clear winner when there are a lot of small jobs to handle.

I further simplified the syntax for a "i = 0 .. n-1" -type of loop with a simple macro:

```
#define PARALLEL_LOOP(_ICOUNT_,_OPERATION_) { CoWork _co_; _co_ * [&] { for(int i =  
_co_.Next(); i < _ICOUNT_ ; i = _co_.Next()) { _OPERATION_; } };
```

So, I can run the loop with:

```
PARALLEL_LOOP(n, operation(ib[i]));
```

Thanks and best regards,

Tom
