Subject: AsyncWork and pick/std::move problems...
Posted by mirek on Wed, 18 Sep 2024 13:20:14 GMT
View Forum Message <> Reply to Message

[quote title=mirek wrote on Wed, 18 September 2024 11:44]Lance wrote on Tue, 17 September 2024 17:34Thank you mirek for all the hard work!

I have a question. In CoWork.h, line 186, etc,

```
  void Do(Function&& f, Args&&... args) { co.Do([=]() { ret = f(args...); }); }
```

shouldn't it be something like

```
void Do(Function&& f, Args&&... args) { co.Do([=]() { ret = f(std::forward<Args>(args)...); }); }
```

OK, upon further investigation, the problem is if you try to e.g. std::move vector into AsyncWork (and there is little reason to use std::forward if you do not), it needs to be moved into the lambda, which is not a trivial thing for parameter pack.

Here is some related info

 https://stackoverflow.com/questions/47496358/c-lambdas-how-t
o-capture-variadic-parameter-pack-from-the-upper-scope

- I have tried to use C++17 variant, but so far it fails... this is my experimental code so far:

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

using namespace Upp;

struct Foo {
 template<class Function, class... Args>
 void  Do(Function&& f, Args&&... args) {
  CoWork().Do([f, a = std::make_tuple(std::forward<Args>(args) ...)]() mutable {
   return std::apply([f](auto&& ... as){ f(as...); }, std::move(a));
  });
 }
};

template< typename Function, class... Args>
void Async2(Function&& f, Args&&... args)
{
 Foo h;
 h.Do(f, std::forward<Args>(args)...);
```

```
}

CONSOLE_APP_MAIN
{
 Vector<double> data;
 data << 1 << 2 << 3;
 Async2([](Vector<double>&& data) -> double {
  return Sum(data);
 }, pick(data));
}
```

but it does not compile... Do not have enough time/energy to solve it now...

Mirek
