Passing lambdas to std::thread and calling class methods

I'm having a bit of trouble using std::thread together with lambdas. I have a method TheMethod where I should use std::thread to parallelize some function calls to methods in the same class.

I define a lambda function, and try to pass it as follows to the std::thread instance I create:

```
auto functor =
   [this](const Cursor& c, size_t& result) ->void {result = classMethod(c);};

size_t a;
Cursor cursor = someCursor();

std::thread t1(functor, cursor, a);

t1.join();
```

Unfortunately, the compiler gives me:

```
  /usr/include/c++/4.8/functional:1697:61: error: no type named ‘type’ in ‘class
std::result_of<TheMethod...
```

I tried a lot of combinations in the lambda definition, and in the way of calling the std::thread constructor, but I get the same error always. The thread library is included, I link pthread too.

Thanks for hints!

c++ multithreading c++11 lambda

asked Mar 11 '14 at 17:29

user46317
147 2 12

8 Say `std::thread t1(functor, std::ref(cursor), std::ref(a));` – Kerrek SB Mar 11 '14 at 17:33

Indeed, now it compiles. Thanks @KerrekSB ! – user46317 Mar 11 '14 at 17:35

you should mark the anwer as valid, so next time people will catch that this is the good answer – Gabriel Mar 30 '14 at 0:09

How can I do that? I only see the option to vote up next to Kerrek SBs answer. Sorry, I'm kind of new here. – user46317 Apr 2 '14 at 9:03

1 Answer

You can use std::ref to pass the parameters by reference:

```
std::thread t1(functor, std::ref(cursor), std::ref(a))
```

You could also capture the parameters by reference in the lambda itself:

```
size_t a;
Cursor cursor = someCursor();
std::thread t1([&] {a = classMethod(cursor);});
t1.join();
```

answered Jul 4 '14 at 22:29

alexk7

1,063 12 16