

Forum:[Hack Forums](#)

Thread: [MultiThreaded Event Loop from scrach but stucked](#)

zNinja - April 9, 2020, 2:17 am

I am trying to implement my own event loop for my project.

I notice that sometimes, like once in 30 runs, one of threads is blocking the main thread on thread.join method.

I am trying to understand why.

Code is interesting, not too big, but is small to post it here.

So I will put a github link where file is uploaded.

[GitHubLink](#)

Valid output is similar to this

PHP Code

```
<?php
```

```
thread in : lock:11
```

```
thread in : 2
```

```
lock:1
```

```
thread in : 3
```

```
lock:1
```

```
thread in : 4
```

```
lock:1
```

```
thread in : 5
```

```
lock:1
```

```
lock:1
```

```
unlock
```

```
lock:1
```

```
unlock
```

```
thread hello world
```

```
unlock
```

```
unlock
```

```
lock:1
```

```
unlock
```

```
unlock
```

```
unlock
```

```
unlock
```

```
join
```

```
join done
```

```
thread out: 4
```

```
join
```

```
join done
```

```
thread out: 3
```

```
lock:1
```

```
unlock
```

```
join
```

```
join done
```

```
thread out: 2
```

```
join
join done
thread out: 1
join
join done
thread out: 0
lock:1
```

Goal is to all Thread object have been destructed successfully by having " thread out 0" eventually in program output.

Invalid output similar to this

PHP Code

```
<?php
thread in : 1
thread in : 2
thread in : 3
thread in : 4
thread in : 5
lock:1
unlock
lock:1
lock:1
lock:1
unlock
thread hello world
unlock
join
lock:1
unlock
lock:1
unlock
lock:1
lock:1
```

Btw, this is code from scratch. To see if concept can work or not before implementing it into library for use case.

Any help is welcome.

In Thread object on calling the join is blocking.

Thread is abstract class of EventLoop object. Problem is probably in "run" method of EventLoop object but I am not able to figure out exactly where and why.

I can not see any dead lock so far.

Edit.

Oh, so sorry. I realize just now I did not write on c/c++ section.

If any can move this topic to c/c++.

saltycracker - April 9, 2020, 6:28 am

First question.. Why a multi-thread event loop?

Why not a thread for the main exe and a thread for the event loop? You should explore how a GUI implements a event loop. Its basically a thread for the event loop and numerous callback functions

to be called at certain events.

BTW... Multi thread applications are the hardest apps to debug... Its kind of why the industry has move to proven thread models and/or embrace immutable data structures while multi-threading.

I doubt you'll find anyone to debug this for you because multi-threading apps behave differently on different systems. Meaning! I might not even be able to reproduce your problem on my system.

zNinja - April 9, 2020, 7:24 am

(Apr 9, 2020 06:28 AM)saltycracker Wrote: [\[CITE\]](#) First question.. Why a multi-thread event loop?My main goal is to learn more about event loops in the first place and implement as flexible as possible for multi thread scenario.

Multi thread because some apps will probably like to have separated ui thread from I/O and other background tasks but in a way where they can predict which thread will be used for what type of task.

And in most cases you will need to have at least 3-4 threads prepared for such app implementation. Example: main for gui, one from file I/O, database... etc.. one for network communication... so whole application is responsive.

(Apr 9, 2020 06:28 AM)saltycracker Wrote: [\[CITE\]](#) Why not a thread for the main exe and a thread for the event loop? You should explore how a GUI implements a event loop. Its basically a thread for the event loop and numerous callback functions to be called at certain events.EventLoop object should provide just what GUI event loop does.

I think that GUI does implement event loop in a very similar way.

But that is where problem exists. So it is what I need to discuss about event loop implementation.

I can not notice what is wrong with my implementation, so it is a sign about there is more about c++ I need to reveal, just to find a source to learn from.

(Apr 9, 2020 06:28 AM)saltycracker Wrote: [\[CITE\]](#) I doubt you'll find anyone to debug this for you because multi-threading apps behave differently on different systems. Meaning! I might not even be able to reproduce your problem on my system.I do not need for someone to debug instead of me, I need review of EventLoop implementation by some more experienced programmer to point me out to example of bad practice implementation from my code.

saltycracker - April 9, 2020, 8:40 am

(Apr 9, 2020 07:24 AM)zNinja Wrote: [\[CITE\]](#) My main goal is to learn more about event loops in the first place and implement as flexible as possible for multi thread scenario.

Multi thread because some apps will probably like to have separated ui thread from I/O and other background tasks but in a way where they can predict which thread will be used for what type of task.

And in most cases you will need to have at least 3-4 threads prepared for such app implementation. Example: main for gui, one from file I/O, database... etc.. one for network communication... so whole application is responsive.

EventLoop object should provide just what GUI event loop does.

I think that GUI does implement event loop in a very similar way.

But that is where problem exists. So it is what I need to discuss about event loop implementation.

I can not notice what is wrong with my implementation, so it is a sign about there is more about c++ I need to reveal, just to find a source to learn from.

I do not need for someone to debug instead of me, I need review of EventLoop implementation by some more experienced programmer to point me out to example of bad practice implementation from my code.

Your code doesn't work and you want someone to point out the errors. We call that debugging the program.

You have a few choices.

1. You can start cruising on github for projects that implement a multithread event loop in C++ and compare and contrast.
2. You can purchase one or many large books on the topic and start reading.
3. You might be able to find a discussion group for C++'ers that specializes in multi-threading.

zNinja - April 9, 2020, 9:46 am

(Apr 9, 2020 08:40 AM)saltycracker Wrote: [\[CITE\]](#) Your code doesn't work and you want someone to point out the errors. We call that debugging the program.

You have a few choices.

1. You can start cruising on github for projects that implement a multithread event loop in C++ and compare and contrast.
2. You can purchase one or many large books on the topic and start reading.
3. You might be able to find a discussion group for C++'ers that specializes in multi-threading.

I have found what is wrong.

std::atomic_bool m_running; causing race condition event it should not by some standards.

So this line [if\(is_running\(\)\)](#) is returning false since thread function is not yet executed before destroying the object, so at m_tread.join() EventLoop starts blocking the thread which destructed the Thread object.

Also this line can probably cause undefined behavior.

<https://github.com/z-ninja/nwOS/blob/z-ninja-bug/main.cpp#L335>

if you see the constructor.

Conclusion: atomic variables are not for this use case.

[quote='saltycracker' pid='59613091' dateline='1586446820']

Your code doesn't work and you want someone to point out the errors. We call that debugging the program.

/quote]

I was thinking on writing the code (best practice) if there is something I do off standard.

I end up with this patch for now.

<https://github.com/z-ninja/nwOS/blob/z-ninja-bug/main.cpp#L336>