
Subject: Stopping ReadStdIn() function
Posted by [Xemuth](#) on Sun, 16 May 2021 18:08:23 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello,

I'm working on a project which implement a tiny command line interface :

```
Upp::String command;  
LOG("Command line interface ready");  
while(command != "exit" && !secureStop){  
    command = ToLower(ReadStdIn());  
    Cout() << ProcessCommandLine(command);  
}
```

secureStop boolean is use to stop my app (in case of SIGINT) however, since my while is in blocking mode because of ReadStdIn() It wont stop my app until I write something to console. Is there a way to set a timeout or stop the ReadStdIn() ?

Subject: Re: Stopping ReadStdIn() function
Posted by [Oblivion](#) on Sun, 16 May 2021 18:53:06 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello xemuth,

Have you tried to set the boolean value, using signal api (sigaction, to be specific)?

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

```
using namespace Upp;
```

```
static std::atomic<bool> done(false);  
static void SecureStop(int) { done = true; }
```

```
CONSOLE_APP_MAIN  
{  
    String s;  
    struct sigaction sa;  
    Zero(sa);  
    sa.sa_handler = &SecureStop;  
    sigaction(SIGINT, &sa, nullptr);
```

```
while(s != "exit" && !done) {  
    s = ReadStdIn();
```

```

}
if(done) {
    Cout() << "SIGINT!" << "\n";
}
}

```

Is this what you need?

Best regards,
Oblivion

Subject: Re: Stopping ReadStdIn() function
 Posted by [Xemuth](#) on Mon, 17 May 2021 07:28:21 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Oblivion, Thanks for your help.

In my actual code I'm doing this :

```
CV2DServer* serverPtr;
```

```

CONSOLE_APP_MAIN
{
    StdLogSetup(LOG_COUT | LOG_FILE | LOG_TIMESTAMP);
    CV2DServer server(LISTENING_PORT, TICK_RATE);
    serverPtr = &server;
    std::signal(SIGINT,static_cast<__p_sig_fn_t>([](int s)->void{serverPtr->StopServer();}));
    server.StartServer();
}

```

It is a bit different from your approach but I guess it is partially equal.

The problem is, in my StartServer which is equal to this :

```

void CV2DServer::StartServer(){
    serverThread.Start(THISBACK(ServerRoutine));
    Upp::String command;
    LOG("Command line interface ready");
    while(command != "exit" && !secureStop){
        command = ToLower(ReadStdIn());
        Cout() << ProcessCommandLine(command);
    }
    server.Close();
    client.Close();
    serverThread.ShutdownThreads();
    serverThread.Wait();
}

```

```
}
```

The ReadStdIn() block my code until I write something in console. It mean, even if I raise a SIGINT event, my code will catch it, turn the boolean secureStop to true but wont stop until I write something in console. This is problematique, that's why I'm looking for a way to stop the ReadStdIn().

I will try the way you do it after work and will update this post. But I'm pretty sure result will be the same

Subject: Re: Stopping ReadStdIn() function
Posted by [Oblivion](#) on Mon, 17 May 2021 08:46:45 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Xemuth,

You are insalling a SIGINT hook and from that point on, it is your responsibility to send a SIGINT to the foreground process:

```
std::signal(SIGINT,static_cast<__p_sig_fn_t>([](int s)->void{serverPtr->StopServer();}));
```

One way might be to set the SIGINT hook to default after you've clean up the application:

```
static void SecureStop(int)
{
    // Cleanup the app here...

    signal(SIGINT, SIG_DFL); // unsintall your hook.
    raise(SIGINT);          // Let OS handle it.
}
```

This way, you shouldn't even need "done" flag...

Example:

```
static std::atomic<bool> done(false);
static void SecureStop(int)
```

```

{
    // Do app cleanup here..
    done = true;
    signal(SIGINT, SIG_DFL); // unsintall the hook.
    raise(SIGINT);
}

CONSOLE_APP_MAIN
{
    StdLogSetup(LOG_COUT|LOG_FILE);

    auto worker = Async([] { // Simulate the signal request...
        int timeout = 5000;
        int t = msec();
        while(msec(t) < timeout) {
            if(CoWork::IsCanceled())
                return;
            Sleep(10);
        }
        if(!done) {
            RLOG("rasigin SIGINT with custom hook...");
            raise(SIGINT);
        }
    });
    signal(SIGINT, &SecureStop);
    String s;
    while(s != "exit") {
        s = ReadStdIn();
    }
    worker.Cancel();
}

```

Please note that this may work, because the code you posted is very simple. This isn't a good way to handle it if it get complex, as you need to make your cleanup-code thread-safe.

Best regards,
Oblivion

Subject: Re: Stopping ReadStdIn() function
 Posted by [Oblivion](#) on Mon, 17 May 2021 09:50:17 GMT
[View Forum Message](#) <> [Reply to Message](#)

You can also write a non-blocking version of ReadStdIn so that you can set a timeout value. (this is for posix, but it can be easily adapted to windows)

```

void SetNonBlocking()
{
    fcntl(STDIN_FILENO, F_SETFL, fcntl(STDIN_FILENO, F_GETFL) | O_NONBLOCK);
}

void SetBlocking()
{
    fcntl(STDIN_FILENO, F_SETFL, fcntl(STDIN_FILENO, F_GETFL) & ~O_NONBLOCK);
}

String ReadStdIn2(int timeout)
{
    String r;
    SetNonBlocking();
    int t = msec();
    do {
        SocketWaitEvent we;
        we.Add(STDIN_FILENO, WAIT_READ);
        if(we.Wait(10) && (we[0] & WAIT_READ)) {
            int c = 0;
            int n = read(STDIN_FILENO, (char*) &c, 1);
            if(c == '\n')
                break;
            if(n > 0) {
                r.Cat(c);
                t = msec();
            }
        }
    }
    while(msec(t) < timeout && !done);
    SetBlocking();
    return r.GetCount() ? r : String::GetVoid();
}

```

Best regards,
Oblivion

Subject: Re: Stopping ReadStdIn() function
 Posted by [Xemuth](#) on Mon, 17 May 2021 16:24:50 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello again Oblivion, I like the timeout approach. here is what I did :

```
String ReadCmd(int timeout)
{
    String r;
    int time = msec();

    while(msec(time) < timeout){
        int c = 0;
        int n = read(STDIN_FILENO, (char*) &c, 1);
        if(c == '\n')
            break;
        if(n > 0) {
            r.Cat(c);
            time = msec();
        }
    }
    return r.GetCount() ? r : String::GetVoid();
}
```

It now work perfectly. Thanks for your precious help !
