

---

Subject: Middle-ground between CONSOLE\_APP\_MAIN and GUI\_APP\_MAIN

Posted by [cbpporter](#) on Fri, 05 Nov 2010 10:00:17 GMT

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

---

I was wondering how could I achieve a middle ground between these two modes, basically a CONSOLE\_APP\_MAIN that expands to WinMain or alternatively a GUI application with a GUI\_APP\_MAIN that does not create any windows and does not use CtrlCore. I do not want to have a console window.

How does TheIDE control the option for creating a console app or not based on the absence or presence of the GUI flag.

A week ago I hacked this together and it works fine except for the acquisition of arguments, but I would like to apply a proper solution:

```
#ifdef PLATFORM_WIN32
```

```
void AppInit__(int argc, const char **argv);  
void AppInitEnvironment__();
```

```
#define CONSOLE_APP_MAIN2 \
```

```
void ConsoleMainFn_(); \
```

```
\
```

```
int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE, LPTSTR lpCmdLine, int  
nCmdShow) \
```

```
{ \
```

```
    UPP::AppInitEnvironment__(); \
```

```
    ConsoleMainFn_(); \
```

```
    UPP::DeleteUsrLog(); \
```

```
    UPP::AppExit__(); \
```

```
    return UPP::GetExitCode(); \
```

```
} \
```

```
\
```

```
void ConsoleMainFn_()
```

```
#endif
```

```
#ifdef PLATFORM_POSIX
```

```
void AppInit__(int argc, const char **argv, const char **envp);
```

```
#define CONSOLE_APP_MAIN2 \
```

```
void ConsoleMainFn_(); \
```

```
\
```

```
int main(int argc, const char **argv, const char **envp) { \
```

```
    UPP::AppInit__(argc, argv, envp); \
```

```
    ConsoleMainFn_(); \
```

```
    UPP::DeleteUsrLog(); \
```

```
UPP::AppExit__(); \
return UPP::GetExitCode(); \
} \
\
void ConsoleMainFn_()

#endif
```

Ideally, is there a way to have you application not open o console, but when it is opened from a console it still writes to it. If I compile it with GUI, i get no output, even if opened from cmd. If I compile it without, I get the output, but when opening it from the shell I get a new console window.

---

---

Subject: Re: Middle-ground between CONSOLE\_APP\_MAIN and GUI\_APP\_MAIN  
Posted by [dolik.rce](#) on Fri, 05 Nov 2010 12:04:54 GMT

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

---

cbpporter wrote on Fri, 05 November 2010 11:00: Ideally, is there a way to have you application not open o console, but when it is opened from a console it still writes to it.  
Hi cbporter,

This is the default behavior for GUI apps on Linux. On windows it is more problematic... Theide uses a wrapper (umk) for command line invocation, as you probably know. My knowledge of windows is very limited, but this article seems to explain the topic quite good and also provides solutions. If you can apply it on U++, it would be IMHO valuable addition.

Honza

---

---

Subject: Re: Middle-ground between CONSOLE\_APP\_MAIN and GUI\_APP\_MAIN  
Posted by [cbpporter](#) on Fri, 05 Nov 2010 12:28:20 GMT

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

---

Thank you for the help. I've tried the solution there, but it opens a new console window, even if I already have one. And indeed, I am trying to emulate the Linux behavior without realizing it.

I have one of the following situations:

1. I compile without GUI flag. Application uses existing console or opens new one. Irrlicht debug messages appear. So do my own console outputs.

This situation is not acceptable, because I don't want windows to create a console.

2. I compile with GUI flag. No console appears and output is not visible.

Here there are two issues.

First, it would be nice if we could pick up the existence of a console and use it to output. This is a convenience feature, so I don't need it but it is useful. Linux model.

Second, I need to include CtrlLib only for GUI\_APP\_MAIN. That is why I am using a CONSOLE\_APP\_MAIN2 that uses WinMain instead of main. This is why I would like to add something like CONSOLE\_WIN\_MAIN that is just CONSOLE\_APP\_MAIN under Linux, and uses WinMain under Windows.

3. I use the example from your link and I have one or two consoles, where the extra one pick up only my outputs, but not Irrlicht debug messages. I guess it uses a different handle and default console has them all set to the same.

---

Subject: Re: Middle-ground between CONSOLE\_APP\_MAIN and GUI\_APP\_MAIN  
Posted by [andrei\\_natanael](#) on Fri, 05 Nov 2010 17:52:28 GMT

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

---

Hello cbpporter,

It may not be what you really want... because user may press enter on that console and introduce new commands... but in this way the program reuse the same console from which it was started.

```
#include <wincon.h>
#include <stdio.h>
#include <fcntl.h>
#include <io.h>
#include <iostream>
#include <fstream>

void SetConsole()
{
    if (AttachConsole(-1) != 0) {
        HANDLE cstdout_handle = GetStdHandle(STD_OUTPUT_HANDLE);
        *stdout = *(FILE*)_fdopen(_open_osfhandle(intptr_t(cstdout_handle), _O_TEXT), "w");
        setvbuf(stdout, NULL, _IONBF, 0);
        Cout() << "\n";
    }
}
```

Run it in GUI\_APP\_MAIN... or embed it in macro. It should work.

Andrei

---

Subject: Re: Middle-ground between CONSOLE\_APP\_MAIN and GUI\_APP\_MAIN  
Posted by [andrei\\_natanael](#) on Fri, 05 Nov 2010 17:54:33 GMT

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

---

P.S.: There is a way to get ride of user actions... but that implies using windows hooks and ignore user input until your program ends

---

---

Subject: Re: Middle-ground between `CONSOLE_APP_MAIN` and `GUI_APP_MAIN`  
Posted by [cbpporter](#) on Mon, 08 Nov 2010 14:10:00 GMT

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

---

Thanks for the help. This also does not work like I want it. No use bothering with it though.

So any chance on getting the WinMain `CONSOLE_APP_MAIN` included into Core as an alternative with a different name? If yes, I'll add parameter recognition like with `GUI_APP_MAIN`. If not, I'm fine with only adding the macro to my code.

---