Subject: Conditional compiling
Posted by forlano on Fri, 15 Mar 2019 20:06:32 GMT
View Forum Message <> Reply to Message

Hello,

my program run under Windows and Linux. The source code is the same but there are part of the code that works only in windows.
So far I used

```
#ifdef PLATFORM_WIN32
  // windows code
#else
  // linux code
#endif
```

to choose what to compile. In future I need to compile different the code depending of Windows, Linux and MacOS.

I wonder which def flag will permit me to separate the Linux stuff from the MacOS one.

Thanks a lot,
Luigi

Subject: Re: Conditional compiling
Posted by Zbych on Fri, 15 Mar 2019 21:00:53 GMT
View Forum Message <> Reply to Message

PLATFORM_OSX?

Subject: Re: Conditional compiling
Posted by Klugier on Fri, 15 Mar 2019 22:04:24 GMT
View Forum Message <> Reply to Message

Hello,

It seems that for the UI applications Mirek is using PLATFORM_COCOA. Here is part of TheIDE that creates macOS global menu (ide/idebar.cpp - line 879):

```
void Ide::SetMenuBar()
{
#ifdef PLATFORM_COCOA
 SetMainMenu(THISBACK(MainMenu));
 menubar.Hide();
#else
```

```
 menubar.Set(THISBACK(MainMenu));
#endif
}
```

However, it is only true if Cocoa back-end is enable (currently default and the most advanced one for macOS). So, you can not relay on this flag for your console applications.

In the Core/config.h file there are following platform definitions that are equal on macOS:

```
  #define PLATFORM_MACOS 1
  #define PLATFORM_OSX 1
```

The above declaration are good to check for platform in low level code that doesn't require UI elements. Currently, I would prefer PLATFORM_MACOS over PLATFORM_OSX, because it agrees with current Apple naming convention for their desktop operating system. MAC OS X naming was abandon in 2016. Also, please notice that macOS is POSIX platform, so in most non UI cases you should relay on PLATFORM_POSIX flag.

Sincerely,
Klugier

---

Subject: Re: Conditional compiling
Posted by mirek on Wed, 20 Mar 2019 08:07:21 GMT
View Forum Message <> Reply to Message

Good explanation.

Just one clarification: I have dedicated PLATFORM_COCOA to "native" macos apps (using Cocoa as backend).

In theory, PLATFORM_OSX / PLATFORM_MACOS are active for X11 applications on Mac. Hopefully, that is not important anymore...

Mirek

---