

Re: std::cin and disabling canonical line processing (buffering)

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.stl/2005-07/msg00125.html>

- *From:* "Harold Bamford" <HaroldBamford@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 22 Jul 2005 10:08:03 -0700
-

It works perfectly! Thanks!

"Tom Widmer [VC++ MVP]" wrote:

```
> Harold Bamford wrote:
>> Tom,
>>
>> Oh, this is *so* close!
>>
>> I find that I have to set the console mode to 0 rather than
>> ENABLE_PROCESSED_INPUT in order to get a ^C char -- as expected. No surprise
>> there.
>>
>> And the text-binary translation is now gone, which is great! As an aside, I
>> also found that using:
>>
>> _setmode(0, _O_BINARY)
>>
>> turned off the text translation, but this is clearly an unreliable hack and
>> I prefer your method of opening up the console in binary mode from the get-go.
>>
>> But, (and you *knew* this was coming!):
>>
>> I cannot read the ESC character; it gets silently eaten. My suspicion is
>> that since ESC forms the prefix of several "special" keys that it is being
>> treated as part of a special sequence, such as:
>>
>> // Up Arrow ESC [ A
>> // Down Arrow ESC [ B
>> // Right Arrow ESC [ C
>> // Left Arrow ESC [ D
>> // Home ESC [ 1 ~
>> // Insert ESC [ 2 ~
>> // Delete ESC [ 3 ~
>> // End ESC [ 4 ~
>> // PageUp ESC [ 5 ~
>> // PageDn ESC [ 6 ~
>>
```

Re: std::cin and disabling canonical line processing (buffering)

>> I would have thought that by NOT enabling mouse or window input modes that
>> these would be passed through raw. Apparently not. And just for grins, I
>> tried it with those modes enabled and still cannot get an ESC char.

>>

>> Ideas or suggestions?

>>

>> Thanks so much for your help!

>

> Ok, try this:

>

> #include <windows.h>

> #include <iostream>

> #include <fstream>

>

> class consolebuf: public std::streambuf

> {

> public:

> consolebuf(HANDLE consoleHandle)

> :m_handle(consoleHandle)

> {

> }

>

> protected:

> int underflow()

> {

> DWORD read = 0;

> INPUT_RECORD record;

>

> for(;;)

> {

> if (!ReadConsoleInput(m_handle, &record, 1, &read)

> || read != 1)

> {

> return EOF;

> }

> if (record.EventType == KEY_EVENT)

> {

> if (record.Event.KeyEvent.bKeyDown)

> {

> //here you need to decide what char to return

> if (record.Event.KeyEvent.uChar.AsciiChar != 0)

> {

> m_currentChar =

> record.Event.KeyEvent.uChar.AsciiChar;

> }

> else

> {

> //some kind of special key.

> //for now, just pass the key code

> m_currentChar =

> static_cast<char>(record.Event.KeyEvent.wVirtualKeyCode);

Re: std::cin and disabling canonical line processing (buffering)

Re: std::cin and disabling canonical line processing (buffering)

```
> }
> setg(&m_currentChar, &m_currentChar, &m_currentChar
> + 1);
> return traits_type::to_int_type(m_currentChar);
> }
> }
> }
> }
>
> private:
> HANDLE m_handle;
> char m_currentChar;
> };
>
> int main()
> {
> HANDLE inputHandle = GetStdHandle(STD_INPUT_HANDLE);
> SetConsoleMode(inputHandle, 0);
>
> consolebuf buf(inputHandle);
> std::istream is(&buf);
> char c;
> while(is.get(c))
> {
> if (c == 'x')
> break;
> std::cout << int(c) << std::endl;
> }
> }
>
> Obviously, not all virtual key codes have an obvious char value, so you
> may want to discard any key presses that you don't want to process (just
> continue the for loop).
>
> Tom
>
.
```

• **References:**

- ◆ [std::cin and disabling canonical line processing \(buffering\)](#)
 ◇ From: Harold Bamford
- ◆ [Re: std::cin and disabling canonical line processing \(buffering\)](#)
 ◇ From: Tom Widmer [VC++ MVP]
- ◆ [Re: std::cin and disabling canonical line processing \(buffering\)](#)
 ◇ From: Harold Bamford
- ◆ [Re: std::cin and disabling canonical line processing \(buffering\)](#)
 ◇ From: Tom Widmer [VC++ MVP]
- ◆ [Re: std::cin and disabling canonical line processing \(buffering\)](#)
 ◇ From: Harold Bamford

Re: std::cin and disabling canonical line processing (buffering)

◆ **Re: std::cin and disabling canonical line processing (buffering)**

◇ From: Tom Widmer [VC++ MVP]

- Prev by Date: **Re: std::cin and disabling canonical line processing (buffering)**
- Next by Date: **[STL] How to use set intersection**
- Previous by thread: **Re: std::cin and disabling canonical line processing (buffering)**
- Next by thread: **std::fill n and an array of pointers**
- Index(es):
 - ◆ **Date**
 - ◆ **Thread**