How do I flush the cin buffer?

How do I clear the cin buffer in C++?

c++ cin io-buffering

edited Nov 2 '08 at 20:06
Dana the Sane
9,094 5 37 65

asked Nov 2 '08 at 17:31
Tal

10 Answers

possibly:

```
std::cin.ignore(INT_MAX);
```

this would read in and ignore everything until `EOF` . (you can also supply a second argument which is the character to read until (ex: `'\n'` to ignore a single line).

Also: You probably want to do a: `std::cin.clear();` before this too to reset the stream state.

edited Aug 28 '12 at 14:01
bwDraco
919 12 29

answered Nov 2 '08 at 17:37
Evan Teran
49.6k 10 128 196

5 (Old I know.) Clear before, rather, so the stream is put into a good state where it can operate on its buffer. – GManNickG Oct 3 '10 at 10:33

Thanks, GMan! Did it the wrong way, first, and spent quite some time looking for my mistake. – balu Nov 17 '11 at 23:35

@GManNickG, just fixed the answer. – bwDraco Aug 28 '12 at 14:05

@DragonLord: Obvious solution I should have done in hindsight, thanks. :) – GManNickG Aug 28 '12 at 14:17

Just wanted to point out that for my case, I find the '\n' necessary. Otherwise subsequent "cin >>" doesn't work. – Cardin Nov 21 '13 at 9:39

I would prefer the C++ size constraints over the C versions:

```
// Ignore to the end of file
cin.ignore(std::numeric_limits<std::streamsize>::max())

// Ignore to the end of line
cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n')
```

answered Nov 2 '08 at 18:34
Loki Astari
139k 38 198 363

11 More importantly, the values might be different! (streamsize doesn't have to be int) – Roger Pate Nov 16 '09 at 20:35

could you please cite an example where we need to ignore to the end of file because if I use `cin` and use the first statement above to flush the `cin` buffer, then it keeps prompting for input till I enter `EOF` by pressing `ctrl+d` ? – ajay Oct 19 '13 at 19:39

@ajay: No. That is something you will need to decide. I am merely explaining what the above will do. – Loki Astari Oct 20 '13 at 0:52

```
cin.clear();
fflush(stdin);
```

This was the only thing that worked for me when reading from console. In every other case it would either read indefinitely due to lack of \n, or something would remain in the buffer.

EDIT: I found out that the previous solution made things worse. THIS one however, works:

```
cin.getline(temp, STRLEN);
if (cin.fail()) {
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}
```

edited May 21 '14 at 11:34 answered May 19 '14 at 18:47

jyggorath
110 1 5

```
int i;
  cout << "Please enter an integer value: ";

  // cin >> i; leaves '\n' among possible other junk in the buffer.
  // '\n' also happens to be the default delim character for getline() below.
  cin >> i;
  if (cin.fail())
  {
    cout << "\ncin failed - substituting: i=1;\n\n";
    i = 1;
  }
  cin.clear(); cin.ignore(INT_MAX,'\n');

  cout << "The value you entered is: " << i << " and its double is " << i*2 << ".\n\n";

  string myString;
  cout << "What's your full name? (spaces inclded) \n";
  getline (cin, myString);
  cout << "\nHello '" << myString << "'.\n\n\n";
```

edited Sep 10 '10 at 22:26 answered Oct 9 '09 at 9:26

Loki Astari
139k 38 198 363

user187046

How about:

```
cin.ignore(cin.rdbuf()->in_avail());
```

edited Sep 8 '10 at 6:36 answered Feb 11 '09 at 20:30

Loki Astari
139k 38 198 363

asdf

2 the streambuf's in_avail() function is unreliable, and many implementations just return zero. See: connect.microsoft.com/VisualStudio/feedback/details/509337/… gnu's libc++ is similar – James Caccese Jun 3 '11 at 21:50

The following should work:

```
cin.flush();
```

On some systems it's not available and then you can use:

```
cin.ignore(INT_MAX);
```

edited Sep 8 '10 at 17:55
Loki Astari
139k 38 198 363

answered Nov 2 '08 at 17:34
Gunnar Steinn
529 7 22

1 why manually write a loop when you can tell ignore the read INT_MAX chars until it reaches EOF (the default value of the second param). – Evan Teran Nov 2 '08 at 17:39

You are right :) – Gunnar Steinn Nov 2 '08 at 17:50

Gunnar, might be better to edit your post to reflect this, just in case. – Dana the Sane Nov 2 '08 at 20:08

As far as I can tell, the `flush` method is for output only, and deals with already written characters. – Marc van Leeuwen Aug 21 at 14:32

I prefer:

```
cin.clear();
fflush(stdin);
```

There's an example where cin.ignore just doesn't cut it, but I can't think of it at the moment. It was a while ago when I needed to use it (with Mingw).

However, fflush(stdin) is undefined behavior according to the standard. fflush() is only meant for output streams. fflush(stdin) only seems to work as expected on Windows (with GCC and MS compilers at least) as an extension to the C standard.

So, if you use it, your code isn't going to be portable.

See Using fflush(stdin).

Also, see http://ubuntuforums.org/showpost.php?s=9129c7bd6e5c8fd67eb332126b59b54c&p=452568&postcount=1 for an alternative.

edited Mar 31 '12 at 8:37

answered Nov 2 '08 at 18:55
Shadow2531
8,102 3 18 27

7 fflush(stdin); is Undefined Behavior (in the C programming language), explicitly stated so in 7.18.5.2/2 – Cubbi Jan 15 '11 at 5:48

1 +1 for supplying a valid, working kludge. Some people have jobs others have standards. – Mikhail Jul 27 '12 at 16:18

3 @Mikhail: Your job should include writing standard compliant code. I will make sure to avoid using anything you have written in the future. – Ed S. Jan 4 '13 at 21:46

```
#include <stdio_ext.h>
```

and then use function

```
__fpurge(stdin)
```

answered Nov 8 '14 at 12:47
techcomp
41 5

Another possible (manual) solution is

```
cin.clear();
while (cin.get() != '\n')
{
    continue;
}
```

I cannot use fflush or cin.flush() with CLion so this came handy.

answered Aug 7 at 8:15
Bobul Mentol
16 2

`cin.get()` seems to flush it automatically oddly enough (probably not preferred though, since this is confusing and probably temperamental).

edited Mar 19 at 21:58

lennon310
8,377 10 19 37

answered Mar 19 at 21:38

GuestPerson001
1