cplusplus.com
Search: Go
Not logged in
Reference <ostream> ostream operator<<
register log in

C++
Information
Tutorials
Reference
Articles
Forum

Reference
C library:
Containers:
Input/Output:
<fstream>
<iomanip>
<ios>
<iosfwd>
<iostream>
<istream>
<ostream>
<sstream>
<streambuf>
Multi-threading:
Other:

<ostream>
class templates:
basic_ostream
classes:
ostream
wostream
manipulators:
endl
ends
flush

ostream
ostream::ostream
ostream::~ostream
member classes:
ostream::sentry
member functions:
ostream::flush
ostream::operator<<
ostream::put
ostream::seekp
ostream::tellp
ostream::write
non-member overloads:
operator<< (ostream)
protected members:
ostream::operator= C++11
ostream::swap C++11

public member function

std::ostream::operator<<

<ostream> <iostream>

C++98 C++11

```
                      ostream& operator<< (bool val);
                      ostream& operator<< (short val);
                      ostream& operator<< (unsigned short val);
                      ostream& operator<< (int val);
                      ostream& operator<< (unsigned int val);
arithmetic types (1)  ostream& operator<< (long val);
                      ostream& operator<< (unsigned long val);
                      ostream& operator<< (float val);
                      ostream& operator<< (double val);
                      ostream& operator<< (long double val);
                      ostream& operator<< (void* val);
stream buffers (2)    ostream& operator<< (streambuf* sb );
                      ostream& operator<< (ostream& (*pf)(ostream&));
manipulators (3)      ostream& operator<< (ios& (*pf)(ios&));
                      ostream& operator<< (ios_base& (*pf)(ios_base&));
```

Insert formatted output

This operator (`<<`) applied to an output stream is known as *insertion operator*. It is overloaded as a member function for:

(1) arithmetic types

Generates a sequence of characters with the representation of *val*, properly formatted according to the locale and other formatting settings selected in the stream, and inserts them into the output stream.
Internally, the function accesses the output sequence by first constructing a `sentry` object. Then (if `good`), it calls `num_put::put` (using the stream's *selected locale*) to perform both the formatting and the insertion operations, adjusting the stream's *internal state flags* accordingly. Finally, it destroys the `sentry` object before returning.

(2) stream buffers

Retrieves as many characters as possible from the input sequence controlled by the *stream buffer* object pointed by *sb* (if any) and inserts them into the stream, until either the input sequence is exhausted or the function fails to insert into the stream.
Internally, the function accesses the output sequence by first constructing a `sentry` object. Then (if `good`), it inserts characters into its associated *stream buffer* object as if calling its member function `sputc`, and finally destroys the `sentry` object before returning.

(3) manipulators

Calls `pf(*this)`, where *pf* may be a *manipulator*.
Manipulators are functions specifically designed to be called when used with this operator.
This operation has no effect on the output sequence and inserts no characters (unless the manipulator itself does, like `endl` or `ends` do).

See `operator<<` for additional overloads (as non-member functions) of this operator.

Parameters

`val`

Value to be formatted and inserted into the stream.
Notice that the type of this argument (along with the stream's *format flags*) influences the format of the representation inserted.

`sb`

Pointer to a `streambuf` object from whose controlled input sequence the characters are copied.

`pf`

A function that takes and returns a stream object. It generally is a *manipulator function*.
The standard manipulators which have an effect when used on standard `ostream` objects are:

manipulator	Effect
`endl`	Inserts a new-line character and *flushes* the stream.
`ends`	Inserts a null character.
`flush`	Flushes the stream.
`boolalpha` / `noboolalpha`	Activates/deactivates the insertion of `bool` values as their alphanumerical representations.
`showbase` / `noshowbase`	Activates/deactivates the insertion of base prefixes in the representations of integer values.
`showpoint` / `noshowpoint`	Activates/deactivates the requirement that the decimal point is always written on the representations of floating-point values.
`showpos` / `noshowpos`	Activates/deactivates the requirement that positive values are preceded with a plus sign (+).
`unitbuf` / `nounitbuf`	Activates/deactivates automatic *flushing* after each insertion operation.

uppercase / nouppercase	Activates/deactivates the use of uppercase letters for characters used in the representations of arithmetic types.
dec / hex / oct	basefield setting: Sets that base used to represent integral numerical values.
fixed / scientific / hexfloat / defaultfloat	floatfield setting: Format used to represent floating-point values.
internal / left / right	adjustfield setting: Position where *fill characters* are inserted to adjust the field to the *field width*.

The following extended manipulators can also be applied to istream objects (these take additional arguments and require the explicit inclusion of the <iomanip> header):

manipulator	Effect
setbase	Sets the numerical base used to represent integral numerical values .
setfill	Sets the *fill character* used to adjust fields to the *field width*.
setprecision	Sets the *decimal precision* used to represent floating-point values.
setw	Sets the *field width* used by the next formatted insertion.
setiosflags/resetiosflags	Set/reset format flags.

Return Value

The ostream object (*this).

Errors are signaled by modifying the *internal state flags*, except for *(3)*, that never sets any flags (but the particular manipulator applied may):

flag	error
eofbit	-
failbit	The function failed to format *val* (it may also be set if the construction of sentry failed). For *(2)*, it is set if no characters could be extracted from the object pointed by *sb*.
badbit	Either the insertion on the stream failed, or some other error happened (such as when this function catches an exception thrown by an internal operation). When set, the integrity of the stream may have been affected. For *(2)*, it is also set if *sb* is a *null pointer*.

Multiple flags may be set by a single operation.

If the operation sets an *internal state flag* that was registered with member exceptions, the function throws an exception of member type failure.

Example

```
// example on insertion
#include <iostream>     // std::cout, std::right, std::endl
#include <iomanip>      // std::setw

int main () {
  int val = 65;

  std::cout << std::right;       // right-adjusted (manipulator)
  std::cout << std::setw(10);    // set width (extended manipulator)

  std::cout << val << std::endl; // multiple insertions

  return 0;
}
```

Output:

```
        65
```

Data races

Modifies the object pointed by *sb (2)*.
Modifies the stream object.
Concurrent access to the same stream object may cause data races, except for the standard stream objects (cout, cerr, clog) when these are *synchronized with stdio* (in this case, no data races are initiated, although no guarantees are given on the order in which characters from multiple threads are inserted).

Exception safety

Basic guarantee: if an exception is thrown, the object is in a valid state.
It throws an exception of member type failure if the resulting *error state flag* is not goodbit and member exceptions was set to throw for that state.
Any exception thrown by an internal operation is caught and handled by the function, setting badbit. If badbit was set on the last call to exceptions, the function rethrows the caught exception.

See also

operator<< (ostream)	Insert characters (public member function)
ostream::put	Put character (public member function)
ostream::write	Write block of data (public member function)
istream::operator>>	Extract formatted input (public member function)