C++
Information
Tutorials
Reference
Articles
Forum

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

<sstream>
class templates:
basic_istringstream
basic_ostringstream
basic_stringbuf
basic_stringstream
classes:
istringstream
ostringstream
stringbuf
stringstream
wistringstream
wostringstream
wstringbuf
wstringstream

ostringstream
ostringstream::ostringstream
public member functions:
ostringstream::operator=
ostringstream::rdbuf
ostringstream::str
ostringstream::swap
non-member overloads:
swap (ostringstream)

public member function

std::ostringstream::ostringstream

<sstream>

C++98 C++11

```
default (1)          explicit ostringstream (ios_base::openmode which = ios_base::out);
initialization (2)   explicit ostringstream (const string& str,
                                             ios_base::openmode which = ios_base::out);
```

Construct an object and optionally initialize its content

Constructs an `ostringstream` object:

(1) empty constructor (default constructor)

Constructs an `ostringstream` object with an empty sequence as content.
Internally, its `ostream` base constructor is passed a pointer to a `stringbuf` object constructed with an argument based on *which*.

(2) initialization constructor

Constructs a `ostringstream` object with a copy of *str* as content.
Internally, its `ostream` base constructor is passed a pointer to a `stringbuf` object constructed with values based on *str* and *which* as arguments.

(3) copy constructor (deleted)

Deleted (no copy constructor).

(4) move constructor

Acquires the contents of *x*.
First, the function move-constructs both its base `ostream` class from *x* and a `stringbuf` object from *x*'s internal `streambuf` object, and then associates them by calling member `set_rdbuf`.
x is left in an unspecified but valid state.
It is unspecified whether the sequence controlled by the internal `stringbuf` object is the one in *x* before the call, or a copy of it. In any case, both objects have internal *string buffers* that use independent sequences after the call.

The internal `stringbuf` object has at least the same duration as the `ostringstream` object.

Parameters

`str`
A `string` object, whose content is copied.

`x`
An `ostringstream` object, whose value is moved.

`which`
Open mode: Access given by the internal `stringbuf` object to its internal sequence of characters. It is an object of member type `openmode` for which any combination of the following member values is significant:

C++98 C++11

member constant	stands for	access
`ios_base::in*`	**in**put	The sequence supports input operations.
`ios_base::out`	**out**put	The sequence supports output operations.

Other values of type `ios_base::openmode` may also be specified, although whether they have an effect on `ostringstream` objects depends on the library implementation.

* `ios_base::out` is always set for `ostringstream` objects (even if explicitly not set in argument *which*).
Note that even though `ostringstream` is an output stream, its internal `stringbuf` object may be set to also support input operations.

Example

```
1  // ostringstream constructor
2  #include <iostream>      // std::cout, std::ios
3  #include <sstream>       // std::ostringstream
4
5  int main () {
6    std::ostringstream foo;                            // out
7    std::ostringstream bar (std::ostringstream::ate);  // out|ate
8
9    foo.str("Test string");
10   bar.str("Test string");
11
12   foo << 101;
13   bar << 101;
14
15   std::cout << foo.str() << '\n';
16   std::cout << bar.str() << '\n';
```

```
  return 0;
}
```

Output:

```
101t string
Test string101
```

Data races

The *move constructor (4)* modifies *x*.

Exception safety

Strong guarantee: if an exception is thrown, there are no side effects.

See also

ostringstream::str	Get/set content (public member function)