C++
Information
Tutorials
Reference
Articles
Forum

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

<ios>
types:
basic_ios
fpos
ios
ios_base
io_errc C++11
streamoff
streampos
streamsize
wios
wstreampos
manipulators:
boolalpha
dec
defaultfloat C++11
fixed
hex
hexfloat C++11
internal
left
noboolalpha
noshowbase
noshowpoint
noshowpos
noskipws
nounitbuf
nouppercase
oct
right
scientific
showbase
showpoint
showpos
skipws
unitbuf
uppercase
other functions:
iostream_category C++11

function

std::scientific

<ios> <iostream>

```
ios_base& scientific (ios_base& str);
```

Use scientific floating-point notation

Sets the `floatfield` format flag for the *str* stream to `scientific`.

When `floatfield` is set to `scientific`, floating-point values are written using scientific notation: the value is represented always with only one digit before the decimal point, followed by the decimal point and as many decimal digits as the *precision field* (`precision`). Finally, this notation always includes an exponential part consisting on the letter `e` followed by an optional sign and three exponential digits.

C++98 C++11

The `floatfield` format flag is both a selective and a toggle flag: it can take one or more of the following values:

flag value	effect when set
`fixed`	write floating-point values in fixed-point notation
`scientific`	write floating-point values in scientific notation.
(none)	write floating-point values in default floating-point notation.

The default notation *(none)* is a different `floatfield` value than either `fixed` or `scientific`. The default notation can be selected by calling `str.unsetf(ios_base::floatfield)`.

For standard streams, no `floatfield` is set on initialization (default notation).

The *precision field* can be modified using member `precision`.

Notice that the treatment of the *precision field* differs between the default floating-point notation and the fixed and scientific notations (see `precision`). On the default floating-point notation, the *precision field* specifies the maximum number of meaningful digits to display both before and after the decimal point, while in both the fixed and scientific notations, the *precision field* specifies exactly how many digits to display *after* the decimal point, even if they are trailing decimal zeros.

Parameters

`str`

Stream object whose `floatfield` *format flag* is affected.
Because this function is a manipulator, it is designed to be used alone with no arguments in conjunction with the *insertion* (<<) and *extraction* (>>) operations on streams (see example below).

Return Value

Argument *str*.

Example

```
1  // modify floatfield
2  #include <iostream>     // std::cout, std::fixed, std::scientific
3
4  int main () {
5    double a = 3.1415926534;
6    double b = 2006.0;
7    double c = 1.0e-10;
8
9    std::cout.precision(5);
10
11   std::cout << "default:\n";
12   std::cout << a << '\n' << b << '\n' << c << '\n';
13
14   std::cout << '\n';
15
16   std::cout << "fixed:\n" << std::fixed;
17   std::cout << a << '\n' << b << '\n' << c << '\n';
18
19   std::cout << '\n';
20
21   std::cout << "scientific:\n" << std::scientific;
22   std::cout << a << '\n' << b << '\n' << c << '\n';
23   return 0;
24 }
```

Possible output:

```
default:
3.1416
2006
1e-010

fixed:
3.14159
2006.00000
0.00000

scientific:
3.14159e+000
2.00600e+003
1.00000e-010
```

Data races

Modifies *str*. Concurrent access to the same stream object may cause data races.

Exception safety

Basic guarantee: if an exception is thrown, *str* is in a valid state.

See also

scientific	Use scientific floating-point notation (function)
ios_base::flags	Get/set format flags (public member function)
ios_base::setf	Set specific format flags (public member function)
ios_base::unsetf	Clear specific format flags (public member function)