2014年3月1日 fixed - C++ Reference

Search: Not logged in Reference <ios> fixed register log in

C++
Information
Tutorials
Reference
Articles
Forum

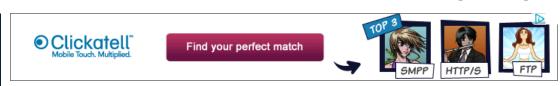
Reference

C library:
Containers:
Input/Output:
<fstream>
<iomanip>
<ios>
<iosfwd>
<iostream>
<istream>
<stream>
<stream>
<stream>
<stream>
<stream>
<hr/>
cotream>
<stream>
<stream>
Cotream>
<hr/>
cotream>
Cotr

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

HTML UI Engine awesomium.com Windowless WebKit Renderer. Use HTML UI in your C++ or .NET app.

iostream_category



function <ios> <iostream>

std::fixed

ios_base& fixed (ios_base& str); Use fixed floating-point notation

Sets the floatfield format flag for the str stream to fixed.

When ${
m float}{
m field}$ is set to ${
m fixed}$, floating-point values are written using fixed-point notation: the value is represented with exactly as many digits in the decimal part as specified by the *precision field* (precision) and with no exponent part.

C++98 C++11

The floatfield format flag is both a selective and a toggle flag: it can take one, both or none 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* ($\langle \langle \rangle$) and *extraction* ($\langle \rangle \rangle$) operations on streams (see example below).

Return Value

Argument str.

Example

```
// modify floatfield
 2 #include <iostream>
                            // std::cout, std::fixed, std::scientific
 4 int main () {
     double a = 3.1415926534;
 5
     double b = 2006.0;
     double c = 1.0e-10;
 9
     std::cout.precision(5);
10
     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;</pre>
     std::cout << a << '\n' << b << '\n' << c << '\n';
18
19
     std::cout << '\n':
20
21
     std::cout << "scientific:\n" << std::scientific;</pre>
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

 $\label{eq:modifies} \textit{Modifies str}. \ \textit{Concurrent access to the same stream object may cause data races}.$

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)

Home page | Privacy policy © cplusplus.com, 2000-2014 - All rights reserved - v3.1 Spotted an error? contact us