Doc No: X3J16/96-0127
WG21/N0945
Date: July 10, 1996
Project: Programming Language C++
Reply-To: Stephen D. Clamage
stephen.clamage@eng.sun.com

Function `setbuf` in IOStreams

IOStream issues 27-707, 809 and 1001 concern the semantics of the `setbuf` member function in the various `streambuf` classes. This paper presents three proposals to add a missing version of `setbuf`, and to clarify the semantics of three versions.

Proposal 1

Add to 27.7.1 “Template class `basic_stringbuf`” [lib.stringbuf] the following member function as a `protected` overridden virtual function:

`virtual basic_streambuf<charT,traits>* setbuf(charT* s, streamsize n);`

Add to 27.7.1.3 “Overridden virtual functions” [lib.stringbuf.virtuals] the following paragraph:

[begin draft text —

`basic_streambuf<charT,traits>* setbuf(charT* s, streamsize n);`

Effects: Implementation-defined, except that `setbuf(0,0)` has no effect.

Returns: `this`.

— end draft text.]

Discussion:

For consistency, `basic_stringbuf` should have a `setbuf` function as do all the other `streambuf` classes. `stringbuf` originally did not have an overriding `setbuf` because it isn’t clear what such a `setbuf` should do. Making a `stringstream` unbuffered doesn’t seem to make any sense, and substituting the supplied buffer might not work well with all implementations. The recommendation allows an implementation to provide semantics that fit. Quite likely, `setbuf` will have no effect.

The return type is not co-variant because its only use is to test success or failure; the exact value is not intended to be meaningful. Changing the return type to `bool` would be inconsistent with existing practice.

Proposal 2

In 27.8.1.4 “Overridden virtual functions” [lib.filebuf.virtuals], function `setbuf` has no assigned semantics. Add the following description:

[begin draft text —

`basic_streambuf* setbuf(char_type*` *s*`, int` *n*`);`

Effects: If `setbuf(0,0)` is called on a stream before any I/O has occured on that stream, the stream becomes unbuffered. Otherwise (parameters are not both zero, or some I/O operation has already occurred), the results are implementation-defined. *Unbuffered* means that output operations on the stream behave as they do when `pbase()` returns null, and input operations on the stream behave as they do when `gptr()` returns null.

Returns: `this`.

— end draft text.]

Proposal 3

In D.6.1.3 “`strstreambuf` overridden virtual functions” [depr.strstreambuf.virtuals] function `setbuf` has no assigned semantics. Add the following description:

[begin draft text —

`streambuf<char>* setbuf(char*` *s*`, streamsize` *n*`);`

Effects: Implementation-defined, except that `setbuf(0,0)` has no effect.

Returns: `this`.

— end draft text.]

Discussion:

In original iostreams, this function had only implementation-defined semantics. As with `stringbufs`, it isn’t clear that we could assign useful semantics; if we did it might break existing implementations.