Doc: X3J16/96-0172=WG21/N0990
Date: September 23, 1996
By: Philippe Le Mouël

Imbuing locale objects in IOstreams

Discussion

The major problem still present in iostreams, is related to imbuing locale objects in the iostreams classes. This paper will try to explain the current status of the WP, then propose a reasonable solution to close the associated iostreams issues.

With the current status, users can imbue locale objects from three different classes:

1) *ios_base*: It only affects the locale object maintained in the stream object.
2) *basic_ios* and derived classes: It affects both the locale object maintained in the stream and the locale object maintained in the stream buffer (if *rdbuf()* does not return a null pointer).
3) *basic_streambuf* and derived classes: It only affects the locale object maintained in the stream buffer.

The problem is to define when users are allowed to imbue locale objects in the iostreams classes. The difficulty in deciding whether or not it is safe to imbue a locale object at a certain point is directly related to the *locale::codecvt* facet used by the file buffer. This facet cannot be changed without proper action taken by the file buffer, and the correct action to be taken depends on which kind of conversion the *locale::codecvt* facet is performing (state dependent or not). The current WP does not address this problem at all.

Proposed resolution

1) In section **27.4.4.2 Member functions [lib.basic.ios.members],** remove the function *basic_ios::imbue*, therefore separating stream and stream buffer imbuing. Users can imbue locale objects from two different classes:

    1) *ios_base* and derived classes: It only affects the locale object maintained in the stream object.
    2) *basic_streambuf* and derived classes: It only affects the locale object maintained in the stream buffer.

   The *ios_base::imbue* function does not need restriction on when it can be called, since it does not have any effect on the stream buffer locale. This will close issues 27-205 and 27-209.

2) In section **27.8.1.4 Overridden virtual functions [lib.filebuf.virtuals],** add the following *basic_filebuf::imbue* description:

   void imbue (const locale& loc);

   Postcondition: *loc == getloc( )*
   Effects: Change the locale object used by the file buffer. The new imbued locale has to meet the following requirements:

    1) The new locale object has to provide a *codecvt* facet specialized on the following types:

- *charT, char , typename traits::state_type*

2) If the new *locale::codecvt<charT, char, typename traits::state_type>* facet performs state dependent encoding conversion, the first sequence of characters read from the file after imbuing the new locale has to start in the initial conversion state.

3) Seeking to a position obtained while using a different *codecvt* facet might lead to undefined behavior.

4) The new locale::*codecvt<charT, char, typename traits::state_type>* facet will first be used the next time a call to *underflow*, *overflow*, *seekoff* or *seekpos* occurs. The characters already in the buffer when imbuing the new locale will not be affected by the change of *codecvt* facet.

Note: The file buffer might be flushed as a result of imbuing a new locale object.

This will close issues 27-805 and 27-814.

Note: In section **27.5.2.2.1 locales [lib.streambuf.locales],** correct the description of function *getloc*:

locale getloc( ) const;

Returns: If *pubimbue( )* has ever been called, then the last value of *loc* supplied, otherwise the default locale *locale::locale()*. If called after *pubimbue( )* has been called but before *pubimbue* has returned (i.e. from within the call of *imbue()*) then it returns the previous value.