Doc: WG21/N0845=X3J16/96-0027
Project: C++ Standard Library
Date: 29 Jan 1996
Reply to: Nathan Myers
          <ncm@cantrip.org>

Make Library Member Typedefs Consistent

Many of the library templates contain typedefs of their arguments, and many do not. This proposal identifies those that do not, and suggests WP text to add them where appropriate.

Member typedefs are omitted where instantiations of the template are expected never to appear in the absence of the argument. These are marked "NO CHANGE" below.

Proposed Resolution:

```
In [lib.numeric.limits] 18.2.1.1, numeric_limits<>:
  NO CHANGE.

In [lib.allocator.requirements] 20.1.4, the template Allocator::types<>:
  NO CHANGE.

In [lib.pairs] 20.2.2, the template pair<>:
  Add public members:
    typedef T1 first_type;
    typedef T2 second_type;

In [lib.default.allocator] 20.4.1, the definition of allocator::types<>:
   NO CHANGE.

In [lib.storage.iterator] 20.4.2, raw_storage_iterator<>
   Add public members:
     typedef OutputIterator iterator_type;
     typedef T element_type;

In [lib.auto.ptr] 20.4.5, auto_ptr<>:
   Add public member:
     typedef T element_type;

In [lib.template.bitset] 23.2.1, bitset<>:
  Add public members:
    typedef bool element_type;
    static const bitset_size = N;

In each of
   [lib.deque] 23.2.2, deque<>,
   [lib.list] 23.2.3, list<>,
   [lib.queue] 23.2.4.1, queue<>,
   [lib.priority.queue] 23.2.4.2, priority_queue<>,
   [lib.stack] 23.2.4.3, stack<>,
   [lib.vector] 23.2.5, vector<>,
   [lib.vector.bool], 23.2.6, vector<bool>,
   [lib.map], 23.3.1, map<>,
   [lib.multimap], 23.3.2, multimap<>,
   [lib.set], 23.3.3, set<>,
   [lib.multiset], 23.3.4, multiset<>:
  Add public member:
    typedef Allocator allocator_type;

In each of
   [lib.map], 23.3.1, map<>,
   [lib.multimap], 23.3.2, multimap<>:
  Add the public member:
    typedef T referent_type;

In [lib.basic.iterators], 24.2.2, in each of
   input_iterator<>,
   forward_iterator<>,
   bidirectional_iterator<>,
   random_access_iterator<>
  Add the public members:
    typedef T value_type;
    typedef Distance distance_type;

In [lib.reverse..bidir.iterator] 24.3.1.1, reverse_bidirectional_iterator<>:
  Add the public members:
    typedef BidirectionalIterator iter_type;
    typedef T value_type;
    typedef Reference reference_type;
    typedef Pointer pointer_type;
    typedef Distance distance_type;

In [lib.reverse.iterator], 24.3.1.3, reverse_iterator<>:
  Add the public members:
    typedef RandomAccessIterator iter_type;
    typedef T value_type;
    typedef Reference reference_type;
    typedef Pointer pointer_type;
    typedef Distance distance_type;

In each of
   [lib.back.insert.iterator], 24.3.2.1, back_insert_iterator<>,
   [lib.front.insert.iterator] 24.3.2.3, front_insert_iterator<>,
   [lib.inset.iterator] 24.3.2.5, insert_iterator<>,
  Add the public members:
    typedef Container container_type;
    typedef typename Container::value_type value_type;

  (In insert_iterator, remove the declaration of member iter.)

In each of
   [lib.istream.iterator] 24.4.1, istream_iterator,
   [lib.ostream.iterator] 24.4.2, ostream_iterator:
  Add the public members:
    typedef T value_type;
    typedef charT char_type;
    typedef traits traits_type;

  In each of istream_iterator and ostream_iterator, add
  template parameters
     charT = char
     traits = ios_traits<charT>

  Replace each use of istream and ostream in the definitions
  with basic_istream<charT,traits> or basic_ostream<charT,traits>,
  respectively.

  Eliminate the istream_iterator<> template parameter Distance.

In each of
   [lib.complex], 26.2.1, complex<>,
   [lib.complex.special], 26.2.2, specializations:
     complex<float>, complex<double>, complex<long double>
   [lib.template.valarray], 26.3.1, valarray<>,
   [lib.template.slice.array], 26.3.4, slice_array<>,
   [lib.template.gslice.array], 26.3.6, gslice_array<>,
   [lib.template.mask.array], 26.3.7, mask_array<>,
   [lib.template.indirect.array], 26.3.8, indirect_array<>,
  Add the public member: (where T is the specialization type)
    typedef T value_type;

In each of
   [lib.ios], 27.4.4., basic_ios<>,
   [lib.streambuf], 27.5.2, basic_streambuf<>:
  Add the public member:
    typedef traits traits_type;
```