Search: Go

Not logged in

Reference <vector> vector **rend**

register log in

C++

Information
Tutorials
Reference
Articles
Forum

Reference

C library:
Containers:
<array>
<deque>
<forward_list>
<list>
<map>
<queue>
<set>
<stack>
<unordered_map>
<unordered_set>
<vector>
Input/Output:
Multi-threading:
Other:

<vector>

vector
vector<bool>

vector

vector::vector
vector::~vector
member functions:
vector::assign
vector::at
vector::back
vector::begin
vector::capacity
vector::cbegin
vector::cend
vector::clear
vector::crbegin
vector::crend
vector::data
vector::emplace
vector::emplace_back
vector::empty
vector::end
vector::erase
vector::front
vector::get_allocator
vector::insert
vector::max_size
vector::operator=
vector::operator[]
vector::pop_back
vector::push_back
vector::rbegin
vector::rend
vector::reserve
vector::resize
vector::shrink_to_fit
vector::size
vector::swap
non-member overloads:
relational operators (vector)
swap (vector)

छात्रों को "स्टूडेंट क्रेडिट कार्ड"और
4 लाख तक का सब्सिडाइज़्ड लोन

public member function

<vector>

std::vector::rend

C++98 C++11

```
      reverse_iterator rend();
const_reverse_iterator rend() const;
```

Return reverse iterator to reverse end

Returns a *reverse iterator* pointing to the theoretical element preceding the first element in the vector (which is considered its *reverse end*).

The range between vector::rbegin and `vector::rend` contains all the elements of the vector (in reverse order).

Parameters

none

Return Value

A reverse iterator to the *reverse end* of the sequence container.

If the vector object is const-qualified, the function returns a `const_reverse_iterator`. Otherwise, it returns a `reverse_iterator`.

Member types `reverse_iterator` and `const_reverse_iterator` are reverse random access iterator types (pointing to an element and to a const element, respectively). See vector member types.

Example

```
// vector::rbegin/rend
#include <iostream>
#include <vector>

int main ()
{
  std::vector<int> myvector (5);  // 5 default-constructed ints

  std::vector<int>::reverse_iterator rit = myvector.rbegin();

  int i=0;
  for (rit = myvector.rbegin(); rit!= myvector.rend(); ++rit)
    *rit = ++i;

  std::cout << "myvector contains:";
  for (std::vector<int>::iterator it = myvector.begin(); it != myvector.end(); ++it)
    std::cout << ' ' << *it;
  std::cout << '\n';

  return 0;
}
```

Output:

```
5 4 3 2 1
```

Complexity

Constant.

Iterator validity

No changes.

Data races

The container is accessed (neither the const nor the non-const versions modify the container).
No contained elements are accessed by the call, but the iterator returned can be used to access or modify elements. Concurrently accessing or modifying different elements is safe.

Exception safety

No-throw guarantee: this member function never throws exceptions.
The copy construction or assignment of the returned iterator is also guaranteed to never throw.

See also

vector::rbegin	Return reverse iterator to reverse beginning (public member function)
vector::front	Access first element (public member function)
vector::begin	Return iterator to beginning (public member function)
vector::end	Return iterator to end (public member function)

Home page | Privacy policy

Spotted an error? contact us