

Search: Go

Not logged in

Reference <vector> vector rbegin

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)

State of Mobility Repo
Download The 2015 State of Mobility
Global Developer Survey. Free EBook



public member function

std::vector::rbegin

<vector>

C++98 C++11

```
reverse_iterator rbegin();
const_reverse_iterator rbegin() const;
```

Return reverse iterator to reverse beginning

Returns a *reverse iterator* pointing to the last element in the *vector* (i.e., its *reverse beginning*).

Reverse iterators iterate backwards: increasing them moves them towards the beginning of the container.

rbegin points to the element right before the one that would be pointed to by member *end*.

Notice that unlike member *vector::back*, which returns a reference to this same element, this function returns a *reverse random access iterator*.

Parameters

none

Return Value

A reverse iterator to the *reverse beginning* 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

```
1 // vector::rbegin/rend
2 #include <iostream>
3 #include <vector>
4
5 int main ()
6 {
7     std::vector<int> myvector (5); // 5 default-constructed ints
8
9     int i=0;
10
11     std::vector<int>::reverse_iterator rit = myvector.rbegin();
12     for (; rit!= myvector.rend(); ++rit)
13         *rit = ++i;
14
15     std::cout << "myvector contains:";
16     for (std::vector<int>::iterator it = myvector.begin(); it != myvector.end(); ++it)
17         std::cout << ' ' << *it;
18     std::cout << '\n';
19
20     return 0;
21 }
```

Output:

```
myvector contains: 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::back	Access last element (public member function)
vector::rend	Return reverse iterator to reverse end (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](#)
© cplusplus.com, 2000-2015 - All rights reserved - v3.1
[Spotted an error? contact us](#)