

Search:

Go

Not logged in

[register](#)

[log in](#)

[Reference](#) <[vector](#)> [vector](#) [size](#)

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)

public member function

std::**vector::size**

<vector>

C++98 | C++11

size_type size() const noexcept;

Return size

Returns the number of elements in the **vector**.

This is the number of actual objects held in the **vector**, which is not necessarily equal to its storage **capacity**.

Parameters

none

Return Value

The number of elements in the container.

Member type `size_type` is an unsigned integral type.

Example

```
1 // vector::size
2 #include <iostream>
3 #include <vector>
4
5 int main ()
6 {
7     std::vector<int> myints;
8     std::cout << "0. size: " << myints.size() << '\n';
9
10    for (int i=0; i<10; i++) myints.push_back(i);
11    std::cout << "1. size: " << myints.size() << '\n';
12
13    myints.insert (myints.end(),10,100);
14    std::cout << "2. size: " << myints.size() << '\n';
15
16    myints.pop_back();
17    std::cout << "3. size: " << myints.size() << '\n';
18
19    return 0;
20 }
```

Output:

```
0. size: 0
1. size: 10
2. size: 20
3. size: 19
```

Complexity

Constant.

Iterator validity

No changes.

Data races

The container is accessed.

No contained elements are accessed: concurrently accessing or modifying them is safe.

Exception safety

No-throw guarantee: this member function never throws exceptions.

See also

vector::capacity	Return size of allocated storage capacity (public member function)
vector::resize	Change size (public member function)
vector::max_size	Return maximum size (public member function)