# list Class

The STL list class is a template class of sequence containers that maintain their elements in a linear arrangement and allow efficient insertions and deletions at any location within the sequence. The sequence is stored as a bidirectional linked list of elements, each containing a member of some type *Type*.

**C++**

template <   
 class Type,   
 class Allocator=allocator<Type>   
>   
class list

## Parameters

*Type*

The element data type to be stored in the list.

*Allocator*

The type that represents the stored allocator object that encapsulates details about the list's allocation and deallocation of memory. This argument is optional, and the default value is **allocator**<*Type*>*.*

## Remarks

The choice of container type should be based in general on the type of searching and inserting required by the application. Vectors should be the preferred container for managing a sequence when random access to any element is at a premium and insertions or deletions of elements are only required at the end of a sequence. The performance of the class deque container is superior when random access is needed and insertions and deletions at both the beginning and the end of a sequence are at a premium.

The list member functions merge, reverse, unique, remove, and remove\_if have been optimized for operation on list objects and offer a high-performance alternative to their generic counterparts.

List reallocation occurs when a member function must insert or erase elements of the list. In all such cases, only iterators or references that point at erased portions of the controlled sequence become invalid.

Include the STL standard header <list> to define the container template class list and several supporting templates.

### Constructors

|  |  |
| --- | --- |
| list | Constructs a list of a specific size or with elements of a specific value or with a specific **allocator** or as a copy of some other list. |

### Typedefs

|  |  |
| --- | --- |
| allocator\_type | A type that represents the **allocator** class for a list object. |
| const\_iterator | A type that provides a bidirectional iterator that can read a const element in a list. |
| const\_pointer | A type that provides a pointer to a const element in a list. |
| const\_reference | A type that provides a reference to a const element stored in a list for reading and performing const operations. |
| const\_reverse\_iterator | A type that provides a bidirectional iterator that can read any const element in a list. |
| difference\_type | A type that provides the difference between two iterators that refer to elements within the same list. |
| iterator | A type that provides a bidirectional iterator that can read or modify any element in a list. |
| pointer | A type that provides a pointer to an element in a list. |
| reference | A type that provides a reference to a const element stored in a list for reading and performing const operations. |
| reverse\_iterator | A type that provides a bidirectional iterator that can read or modify an element in a reversed list. |
| size\_type | A type that counts the number of elements in a list. |
| value\_type | A type that represents the data type stored in a list. |

### Member Functions

|  |  |
| --- | --- |
| assign | Erases elements from a list and copies a new set of elements to the target list. |
| back | Returns a reference to the last element of a list. |
| begin | Returns an iterator addressing the first element in a list. |
| list::cbegin | Returns a const iterator addressing the first element in a list. |
| list::cend | Returns a const iterator that addresses the location succeeding the last element in a list. |
| list::clear | Erases all the elements of a list. |
| list::crbegin | Returns a const iterator addressing the first element in a reversed list. |
| list::crend | Returns a const iterator that addresses the location succeeding the last element in a reversed list. |
| list::emplace | Inserts an element constructed in place into a list at a specified position. |
| list::emplace\_back | Adds an element constructed in place to the end of a list. |
| list::emplace\_front | Adds an element constructed in place to the beginning of a list. |
| empty | Tests if a list is empty. |
| end | Returns an iterator that addresses the location succeeding the last element in a list. |
| erase | Removes an element or a range of elements in a list from specified positions. |
| front | Returns a reference to the first element in a list. |
| get\_allocator | Returns a copy of the **allocator** object used to construct a list. |
| insert | Inserts an element or a number of elements or a range of elements into a list at a specified position. |
| max\_size | Returns the maximum length of a list. |
| merge | Removes the elements from the argument list, inserts them into the target list, and orders the new, combined set of elements in ascending order or in some other specified order. |
| pop\_back | Deletes the element at the end of a list. |
| pop\_front | Deletes the element at the beginning of a list. |
| push\_back | Adds an element to the end of a list. |
| push\_front | Adds an element to the beginning of a list. |
| rbegin | Returns an iterator addressing the first element in a reversed list. |
| remove | Erases elements in a list that match a specified value. |
| remove\_if | Erases elements from the list for which a specified predicate is satisfied. |
| rend | Returns an iterator that addresses the location succeeding the last element in a reversed list. |
| resize | Specifies a new size for a list. |
| reverse | Reverses the order in which the elements occur in a list. |
| size | Returns the number of elements in a list. |
| sort | Arranges the elements of a list in ascending order or with respect to some other order relation. |
| splice | Removes elements from the argument list and inserts them into the target list. |
| swap | Exchanges the elements of two lists. |
| unique | Removes adjacent duplicate elements or adjacent elements that satisfy some other binary predicate from the list. |

### Operators

|  |  |
| --- | --- |
| list::operator= | Replaces the elements of the list with a copy of another list. |

## Requirements

**Header**: <list>