Introduction

This chapter deals with the different features that are present in all types of containers in C++.

Although the sequential and associative containers of the Standard Template library are two quite different classes of containers, they have a lot in common. For example, the operations, to create or delete a container, to determine its size, to access its elements, to assign or swap are all independent of the type of elements of a container. It is common for the containers that you can define them with an arbitrary size, and each container has an allocator. That's the reason the size of a container can be adjusted at runtime. The allocator works most of the time in the background. This can be seen for a `std::vector`. The call `std::vector<int>` results in a call `std::vector<int, std::allocator<int>>`. Because of the `std::allocator`, you can adjust except for `std::array` the size of all containers dynamically. However, they have yet more in common. You can access the elements of a container quite easily with an iterator.

Having so much in common, the containers differ in the details. The chapters Sequential Container and Associative Container provide the details.

With the sequential containers std::array, std::vector, std::deque, std::list, and std::forward_list C++ has an expert on each domain.

The same holds true for the associative containers, which can be classified in the order and unordered ones.