

Search:

Go

C++

Information

Tutorials

Reference

Articles

Forum

Reference

C library:

Containers:

Input/Output:

Multi-threading:

Other:

<algorithm>

<bitset>

<chrono>

<codecvt>

<complex>

<exception>

<functional>

<initializer\_list>

<iterator>

<limits>

<locale>

<memory>

<new>

<numeric>

<random>

<ratio>

<regex>

<stdexcept>

<string>

<system\_error>

<tuple>

<typeindex>

<typeinfo>

<type\_traits>

<utility>

<valarray>

<chrono>

classes:

common\_type (duration)

common\_type (time\_point)

duration

duration\_values

high\_resolution\_clock

steady\_clock

system\_clock

time\_point

treat\_as\_floating\_point

functions:

duration\_cast

time\_point\_cast

class typedefs:

hours

microseconds

milliseconds

minutes

nanoseconds

seconds

Ultra Tune Car Servicing

Contact us for your servicing, tyre, vehicle maintenance and repair needs. Go to [ultratune.com.au](http://ultratune.com.au)

header

## <chrono>

### Time library

chrono is the name of a header, but also of a sub-namespace: All the elements in this header (except for the `common_type` specializations) are not defined directly under the `std` namespace (like most of the standard library) but under the `std::chrono` namespace.

The elements in this header deal with time. This is done mainly by means of three concepts:

#### Durations

They measure time spans, like: one minute, two hours, or ten milliseconds. In this library, they are represented with objects of the `duration` class template, that couples a *count representation* and a *period precision* (e.g., ten milliseconds has ten as *count representation* and milliseconds as *period precision*).

#### Time points

A reference to a specific point in time, like one's birthday, today's dawn, or when the next train passes. In this library, objects of the `time_point` class template express this by using a `duration` relative to an *epoch* (which is a fixed point in time common to all `time_point` objects using the same clock).

#### Clocks

A framework that relates a *time point* to real physical time. The library provides at least three clocks that provide means to express the current time as a `time_point`: `system_clock`, `steady_clock` and `high_resolution_clock`.

For typical examples, see `steady_clock` or `system_clock`.

### Classes

#### duration and time\_point:

<code>duration</code>	Duration (class template )
<code>time_point</code>	Time point (class template )

#### clocks:

<code>system_clock</code>	System clock (class )
<code>steady_clock</code>	Steady clock (class )
<code>high_resolution_clock</code>	High resolution clock (class )

#### traits:

<code>treat_as_floating_point</code>	Treat as floating point (class template )
<code>duration_values</code>	Duration values (class template )
<code>common_type (duration)</code>	Specialization of <code>common_type</code> for <code>duration</code> (class template )

### Functions

<code>duration_cast</code>	Duration cast (function template )
<code>time_point_cast</code>	Time_point cast (function template )

### Class instantiation typedefs

The following convenience typedefs of instantiations of `duration` are also defined in this namespace:

<code>hours</code>	Duration in hours (class )
<code>minutes</code>	Duration in minutes (class )
<code>seconds</code>	Duration in seconds (class )
<code>milliseconds</code>	Duration in milliseconds (class )
<code>microseconds</code>	Duration in microseconds (class )
<code>nanoseconds</code>	Duration in nanoseconds (class )