chrono程序库

duration（时间段）：在chrono库中，duration是一个数值（表现tick数）和一个分数（表现时间单位，以秒计）的组合，其中的分数有ratio表示

例：

*chrono*::*duration*<int> five\_seconds(5); // 5秒，默认单位为秒

*chrono*::*duration*<double, *ratio*<60>> half\_minute(0.5); // 半分钟, ratio<60>表示1分钟

*chrono*::*duration*<long, *ratio*<1, 1000>> one\_millisecond(1); // 1毫秒，ratio<1. 1000>表示1毫秒

例：程序stl\_test26

// duration的算术运算

*chrono*::*seconds* d1(42); // 42秒

*chrono*::*milliseconds* d2(10);

*chrono*::*milliseconds* d3 = d1 - d2;

*cout* << d3.*count*() << *endl*; // 41990毫秒

*chrono*::*duration*<int, *ratio*<1, 3>> d4(1); // 1/3秒

*chrono*::*duration*<int, *ratio*<1, 5>> d5(1); // 1/5秒

// chrono::seconds d6 = d4 + d5; // error，时间单位不匹配

*chrono*::*duration*<int, *ratio*<1, 15>> d6 = d4 + d5; // ok

*cout* << d6.*count*() << *endl*; // 8

// 时间单位匹配

*chrono*::*seconds* ten\_seconds(10);

*chrono*::*hours* one\_hour(1);

*chrono*::*seconds* d7 = one\_hour; // ok，小时到秒可以转换

*cout* << d7.*count*() << *endl*; // 3600

// chrono::hours d8 = ten\_seconds; // error，秒到小时无法转换

// 强制从小时间单位转换到大时间单位

*chrono*::*seconds* seconds(65);

*chrono*::*minutes* minute = *chrono*::*duration\_cast*<*chrono*::*minutes*>(seconds);

*cout* << minute << *endl*; // [1 of 60/1]

输出为：

41990

8

3600

Clock和Timepoint

例：程序stl\_test27

template <typename T>

void PrintClockData()

{

*cout* << "- precison: ";

typedef typename T::*period* P; // 时间单元类型的别名

// 获取时钟精度

// 如果时钟精度小于1/1000

if (*ratio\_less\_equal*<P, *milli*>::*value*)

{

// 将时钟精度乘以1000

typedef typename ratio\_multiply<P, *kilo*>::*type* TT;

*cout* << *fixed* << double(TT::*num*) / double(TT::*den*) << " milliseconds" << *endl*;

}

else {

*cout* << *fixed* << double(P::*num*) / P::*den* << " seconds" << *endl*;

}

// 判断该时钟是否稳定

// 稳定的时钟当系统物理时间朝前拨，它也不会减少

// 最适合计算两个时间点的差距

*cout* << "- is\_steady: " << *boolalpha* << T::*is\_steady* << *endl*;

}

int main(int argc, char\* argv[])

{

*cout* << "system clock: " << *endl*;

PrintClockData<*chrono*::*system\_clock*>();

*cout* << "high resolution clock: " << *endl*;

PrintClockData<*chrono*::*high\_resolution\_clock*>();

*cout* << "steady clock: " << *endl*;

PrintClockData<*chrono*::*steady\_clock*>();

return 0;

}

Timepoint：时间点

例：程序stl\_test28

*string* AsString(*chrono*::*system\_clock*::*time\_point*& tp)

{

// 转换系统时间

*time\_t* t = *chrono*::*system\_clock*::*to\_time\_t*(tp);

char str[30];

*ctime\_s*(str, sizeof(str), &t);

return str;

}

// 系统时间的起点

// 要包含时区的信息

*chrono*::*system\_clock*::*time\_point* tp;

*cout* << "epoch: " << AsString(tp) << *endl*;

// 当前时间

tp = *chrono*::*system\_clock*::*now*();

*cout* << "now: " << AsString(tp) << *endl*;

// 系统时钟的最小时间

// 在VS下该值不可见

tp = *chrono*::*system\_clock*::*time\_point*::*min*();

*cout* << "min: " << AsString(tp) << *endl*;

// 系统时钟的最大时间

// 在VS下该值不可见

tp = *chrono*::*system\_clock*::*time\_point*::*max*();

*cout* << "max: " << AsString(tp) << *endl*;