nov 30, 09 16:39 **test_chrono.cc** Page 1/1

```
#include <iostream>
#include <time.h>

using namespace std;

#include "Chronometre.h"

int main()
{
  Chronometre c1, c2;

  // la fonction "heure courante" n'est pas liée à un objet en particulier

  cout << "Heure système = " << Chronometre::current_time()
       << " secondes depuis le 01/01/1970" << endl;

  cout << "Démarrage c1 et c2" << endl;
  c1.start();
  c2.start();
  cout << "délai 5 secondes" << endl;
  sleep(5);
  cout << "temps intermédiaires c1:" << c1.elapsed()
       << " c2:" << c2.elapsed()
       << endl;
  cout << "création c3 (copie de c1)" << endl;
  Chronometre c3(c1);
  cout << "arrêt c1 " << endl;
  c1.stop();
  cout << "délai 3 secondes" << endl;
  sleep(3);
  cout << "temps intermédiaires c1:" << c1.elapsed()
       << " c2:" << c2.elapsed()
       << " c3:" << c3.elapsed()
       << endl;
  cout << "arrêt c2 " << endl;
  c2.stop();
  cout << "temps intermédiaires c1:" << c1.elapsed()
       << " c2:" << c2.elapsed()
       << " c3:" << c3.elapsed()
       << endl;
  return 0;
}
```

nov 09, 09 13:53 **Chronometre.h** Page 1/1

```
#ifndef CHRONO_H
#define CHRONO_H

#include <time.h>

class Chronometre
  {
  private:
    int    my_elapsed;

    bool   my_is_running;
    time_t my_time_started;

  public:
    static time_t current_time();
    // note : la valeur de current_time() n'est pas liée à un objet en
    // particulier, c'est une méthode de classe. (mot clé : static)

    // constructeurs
    Chronometre ();                                  // par défaut
    Chronometre (const Chronometre & chronometre); // par copie

    void reset ();
    void start();

    void stop ();
    bool is_running() const;
    time_t elapsed () const;

    // note : le mot clé "const" précise que les fonctions
    // ci-dessus ne modifient pas les champs des objets

  };
#endif
```

nov 30, 09 16:37 **Chronometre.cc** Page 1/1

```
#include <iostream>
#include <time.h>

#include "Chronometre.h"

using namespace std;

// méthode de classe

time_t Chronometre::current_time()
{
  return time(NULL);  // appel systeme
}

// Constructeur

Chronometre::Chronometre()
{
  reset();
}

Chronometre::Chronometre(const Chronometre & chronometre)
{
  my_elapsed      = chronometre.my_elapsed;
  my_is_running   = chronometre.my_is_running;
  my_time_started = chronometre.my_time_started;
}

// Méthodes

void Chronometre::reset ()
{
  my_elapsed = 0;
  my_is_running = false;
}

void Chronometre::start()
{
  my_is_running = true;
  my_time_started = current_time();
}

void Chronometre::stop ()
{
  if (my_is_running)
    {
      my_elapsed += (current_time() - my_time_started);
      my_is_running = false;
    }
}

// bonne pratique de documentation : rappeler "const"

bool Chronometre::is_running() const
{
  return my_is_running;
}

time_t Chronometre::elapsed () const
{
  time_t t = my_elapsed;
  if (my_is_running)
    {
      t +=  current_time() - my_time_started;
    }
  return t;
}
```

nov 30, 09 11:58 **Makefile** Page 1/1

```
CXXFLAGS=-Wall -pedantic

LISTINGS=test_chrono.cc Chronometre.h Chronometre.cc Makefile

all: test_chrono listing.pdf

test_chrono: test_chrono.o Chronometre.o
        $(LINK.cc) $^ $(LOADLIBES) $(LDLIBS) -o $@

Chonometre.o: Chronometre.cc Chronometre.h

test_chrono.o: test_chrono.cc Chronometre.h

listing.pdf: $(LISTINGS)
        a2ps -o - -A fill -C $(LISTINGS)| ps2pdf - $@



clean:
        rm -f *~ *.o

mrproper: clean
        rm -f test_chrono listing.pdf

```