

# Delay

---

## main.cpp

```
#include <QCoreApplication>
#include <QObject>

#include "delay.h"

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    delay *d = new delay();

    QObject::connect(d, SIGNAL(sig_quit()), &a, SLOT(quit()));

    d->start(1000);

    return a.exec();
}
```

## delay.h

```
#ifndef DELAY_H
#define DELAY_H

#include <iostream>

#include <QObject>
#include <QTimer>

class delay : public QObject
{
    Q_OBJECT
public:
    explicit delay(QObject *parent = nullptr);

    void start(const float &time);

public slots:
    void slot_timerElapsed();

private:
    QTimer *timer;

signals:
    void sig_quit();
}
```

```
};  
  
#endif // DELAY_H
```

## delay.cpp

```
#include "delay.h"  
  
delay::delay(QObject *parent) : QObject(parent)  
{  
    timer = new QTimer();  
  
    connect(timer, SIGNAL(timeout()), this, SLOT(slot_timerElapsed()));  
  
    return;  
}  
  
void delay::start(const float &time)  
{  
    std::cout << "Timer Started" << std::endl;  
  
    timer->setSingleShot(true);  
    timer->start(time);  
  
    return;  
}  
  
void delay::slot_timerElapsed()  
{  
    std::cout << "Timer Elapsed" << std::endl;  
  
    emit sig_quit();  
  
    return;  
}
```