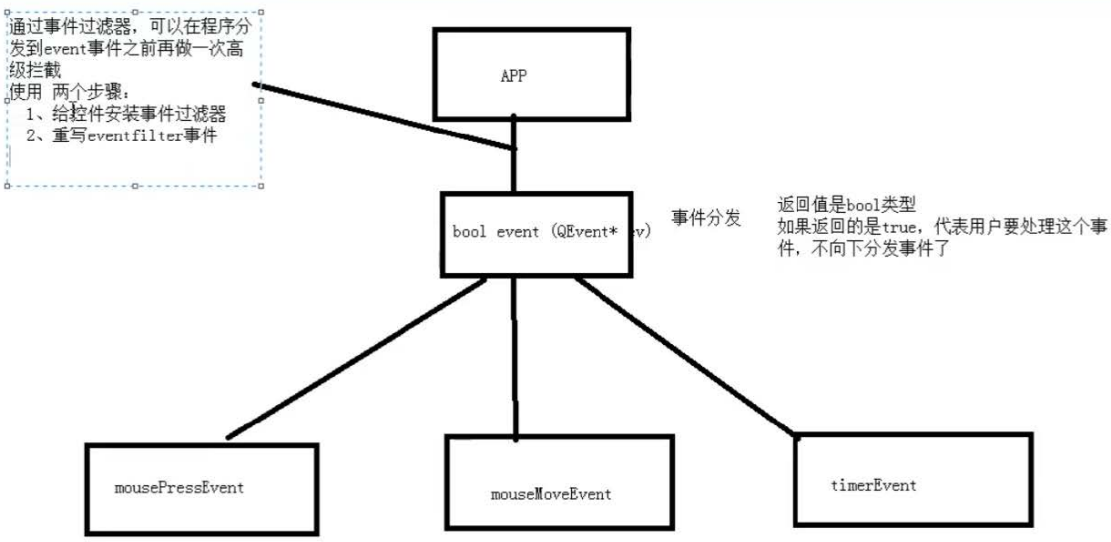
描述：



widget.h

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui { class Widget; }
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();
    bool eventFilter(QObject *,QEvent *);
private:
    Ui::Widget *ui;
};
#endif // WIDGET_H
```

安装事件过滤器

widget.cpp

```
#include "widget.h"
#include "ui_widget.h"

Widget::Widget(QWidget *parent)
    : QWidget(parent)
```

```
    , ui(new Ui::Widget)
{
    ui->setupUi(this);
    ui->label->installEventFilter(this);//给label安装事件过滤器
}

Widget::~Widget()
{
    delete ui;
}
```

重写ventfilter事件

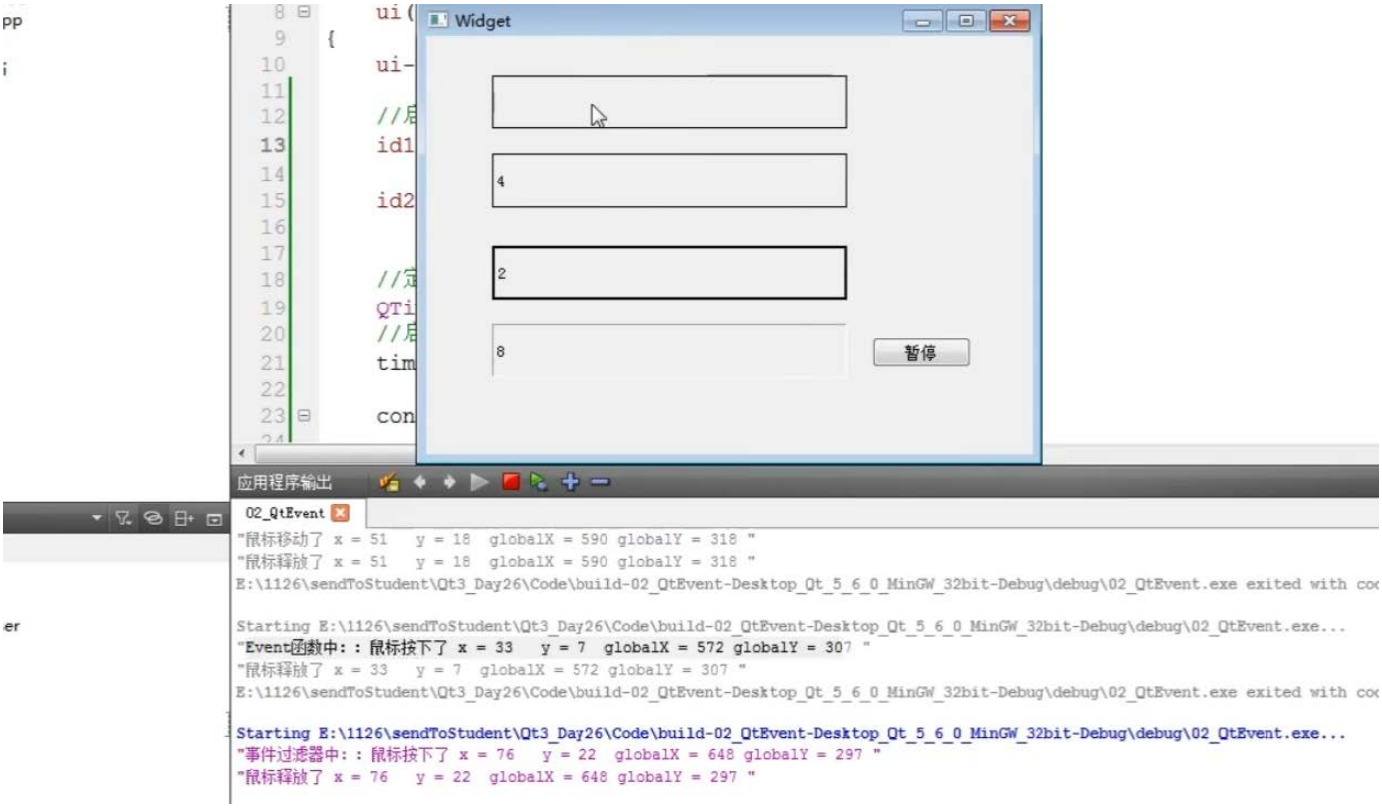
widget.cpp

```
 // 步骤2  重写 eventfilter事件
bool Widget::eventFilter(QObject * obj , QEvent * e)
{
    if(obj == ui->label)
    {
        if(e->type() == QEvent::MouseButtonPress)
        {
            QMouseEvent * ev  = static_cast<QMouseEvent *>(e);
            QString str = QString( "事件过滤器中：：鼠标按下了 x = %1   y = %2  globalX = %3
            qDebug() << str;


            return true; //true代表用户自己处理这个事件，不向下分发
        }

    }

    //其他默认处理
    return QWidget::eventFilter(obj,e);
}
```



更高级的一层拦截

总结:

6.1 在程序将时间分发到事件分发器前，可以利用过滤器做拦截
6.2 步骤
6.2.1 1、给控件安装事件过滤器
6.2.2 2、重写 eventFilter函数 （obj ， ev）