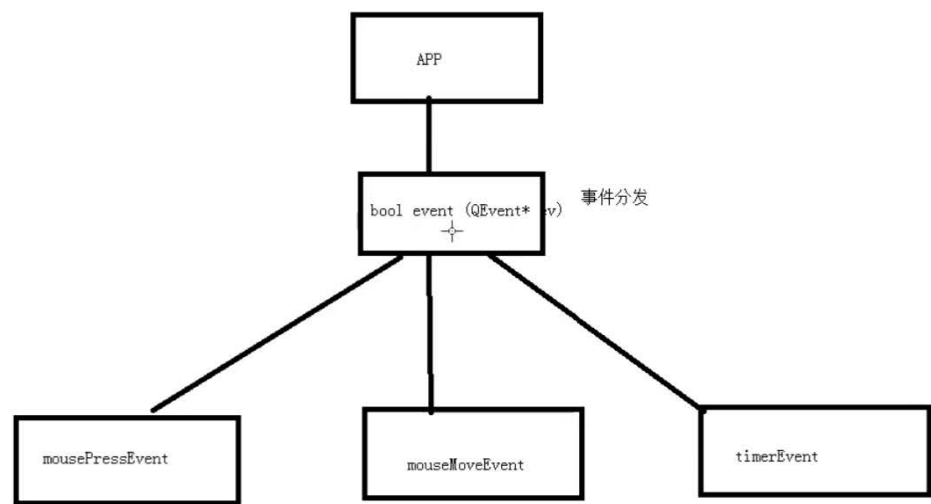


概览



分发器返回值为bool

返回true，代表用户要自己处理该事件

查询文档

索引

查找(L):

QEvent

QEvent

QEvent::ActionAdded

QEvent::ActionChanged

QEvent::ActionRemoved

QEvent::ActivationChange

QEvent::ApplicationFontChange

QEvent::ApplicationLayoutDirectionCha...

QEvent::ApplicationPaletteChange

QEvent::ApplicationStateChange

Detailed Description

QEvent Class

The QEvent class is the base class of all event classes. Event objects contain event parameters. [More...](#)

Header: #include <QEvent>

qmake: QT += core

Inherited By: QChildEvent, QDynamicPropertyChangeEvent, QStateMachine::SignalEvent,

内容 索引 书签 搜索

索引

查找(L):

QEvent

QEvent

QEvent::ActionAdded

QEvent::ActionChanged

QEvent::ActionRemoved

QEvent::ActivationChange

QEvent::ApplicationFontChange

QEvent::ApplicationLayoutDirectionCha...

QEvent::ApplicationPaletteChange

QEvent::ApplicationStateChange

QEvent::MetaCall	43	An asynchronous method invocation via QMetaObject::invokeMethod() .
QEvent::ModifiedChange	102	Widgets modification state has been changed.
QEvent::MouseButtonDb1Click	4	Mouse press again (QMouseEvent).
<u>QEvent::MouseButtonPress</u>	2	Mouse press (QMouseEvent).
QEvent::MouseButtonRelease	3	Mouse release (QMouseEvent).
QEvent::MouseMove	5	Mouse move (QMouseEvent).
QEvent::MouseTrackingChange	109	The mouse tracking state has changed.

mylabel.h

```
bool event(QEvent *e);
```

mylabel.cpp

```

bool mylabel::event(QEvent *e)
{
    //如果鼠标按下，在event事件分发中拦截操作
    if(e->type()==QEvent::MouseButtonPress)
    {
        QString str =QString("Event函数中，鼠标按下了 X= %1 Y=%2").arg(ev->x()).arg(ev->y());
        qDebug()<<str;
        return true;//true代表该事件用户自己处理，不会向下分发
    }
    return QLabel::event(e);//其他事件交给父类处理（默认处理）
}

```

但此时代码是有问题的，什么问题呢？

ev是QMouseEvent类型的参数，而我们定义e是QEvent类型,因此，我们需要转换

```

bool mylabel::event(QEvent *e)
{
    //如果鼠标按下，在event事件分发中拦截操作
    if(e->type()==QEvent::MouseButtonPress)
    {
        QMouseEvent *ev = static_cast<QMouseEvent *>(e);//将e转为 QMouseEvent * 类型（静态类型转换）
        QString str =QString("Event函数中，鼠标按下了 X= %1 Y=%2").arg(ev->x()).arg(ev->y());
        qDebug()<<str;
        return true;//true代表该事件用户自己处理，不会向下分发
    }
    return QLabel::event(e);//其他事件交给父类处理（默认处理）
}

```

```

"鼠标移动了 X= 91 Y=46"
"Event函数中，鼠标按下了 X= 91 Y=46"
"鼠标移动了 X= 91 Y=44"
"鼠标释放了 X= 91 Y=44"

```

生成成功,但我们通常不使用这种拦截方式

总结:

- 5.1 用途：用于事件的分发
- 5.2 也可以做拦截操作，不建议
- 5.3 `bool event(QEvent * e);`
- 5.4 返回值 如果是**true** 代表用户处理这个事件，不向下分发了
- 5.5 `e->type() ==` 鼠标按下 ...