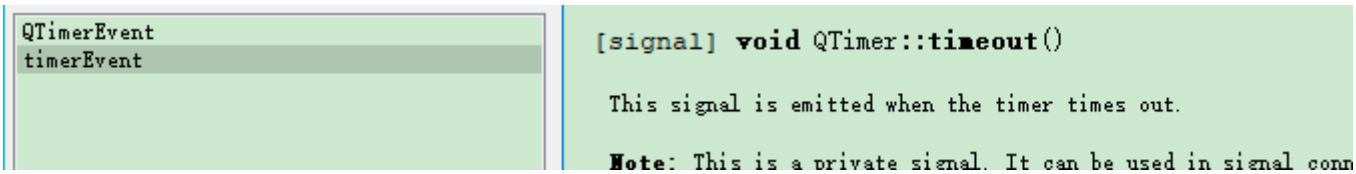
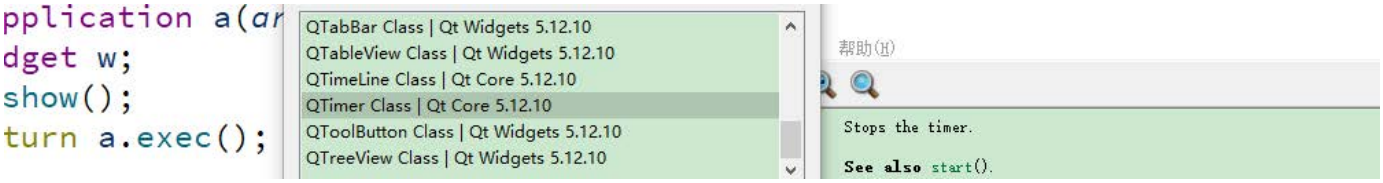
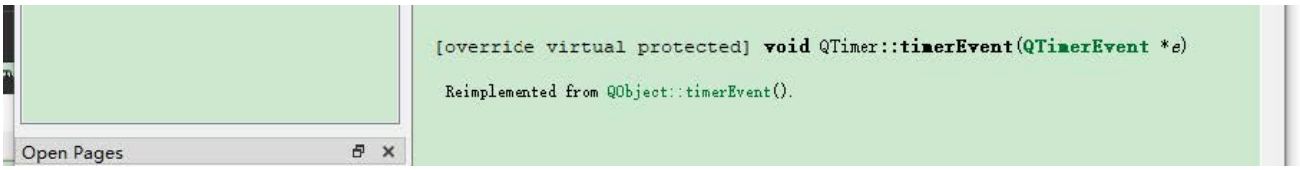
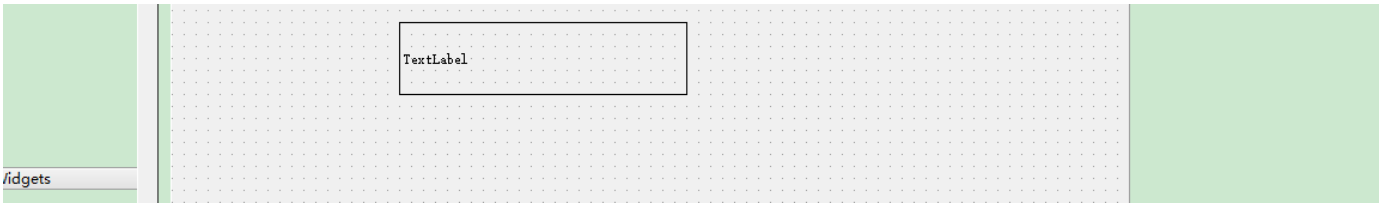
定时器我们需要查询timerEvent







先添加一个标签



widget.h

先包含头文件#include

```
 void timerEvent(QTimerEvent *);
```

实现函数

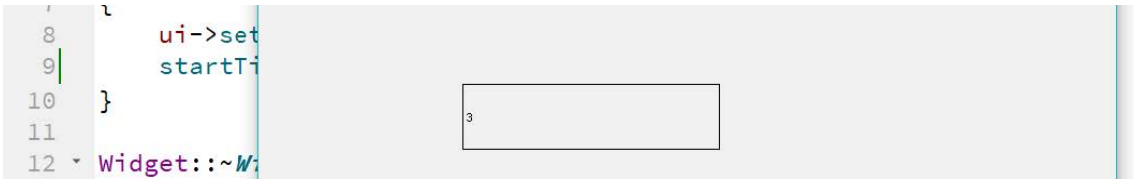
widget.cpp

```
void Widget::timerEvent(QTimerEvent *)
{
    int num=1;
    ui->label->setText(QString::number(num++));
}
```

现在我们启动定时器

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

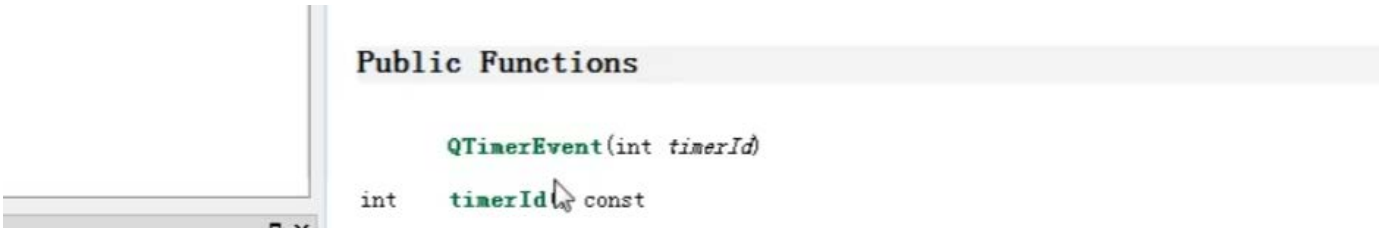
```
Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
    ui->setupUi(this);
    startTimer(1000);//参数1  间隔（单位：ms）
}

Widget::~Widget()
{
    delete ui;
}
void Widget::timerEvent(QTimerEvent *)
{
    static int num=1;
    ui->label->setText(QString::number(num++));
}
```




```
    startTimer(1000);//参数1  间隔（单位：ms）
    startTimer(2000);
```

现在，我们让两个标签的定时器间隔不一样，那么如何区分这两个定时器呢?

如何区分定时器呢,,查文档！！！



timerId定时器标识符，返回值为int

widget.h

```
void timerEvent(QTimerEvent *ev);
    int id1;//定时器1 timerId1
    int id2;//timerId2
```

widget.cpp

使用if判断

```
void Widget::timerEvent(QTimerEvent *ev)
{
    if(ev->timerId()==id1)
    {
       static int num=1;
       ui->label->setText(QString::number(num++));
    }

    if(ev->timerId()==id2)
    {
        static int num2=1;
        ui->label_2->setText(QString::number(num2++));
    }
}
```

总结:

```
1.1    利用事件 void  timerEvent ( QTimerEvent * ev)
1.2    启动定时器 startTimer( 1000) 毫秒单位
1.3    timerEvent 的返回值是定时器的唯一标示   可以和ev->timerId 做比较
```