QT鼠标键盘事件总结

- **Qt** 程序需要在**main()**函数创建一个**QCoreApplication**对象，然后调用它的**exec()**函数。这个函数就是开始 **Qt** 的事件循环。在执行**exec()**函数之后，程序将进入事件循环来监听应用程序的事件。
- 当事件发生时，**Qt** 将创建一个事件对象。**Qt** 中所有事件类都继承于**QEvent**。在事件对象创建完毕后，**Qt** 将这个事件对象传递给**QObject**的**event()**函数。**event()**函数并不直接处理事件，而是将这些事件对象按照它们不同的类型，分发给不同的事件处理器（**event handler**）。
- 如上所述，**event()**函数主要用于事件的分发。所以，如果你希望在事件分发之前做一些操作，就可以重写这个**event()**函数了。

鼠标事件

鼠标事件使用的时候，加头文件**#include <QMouseEvent>**

```
void mousePressEvent(QMouseEvent *event); //单击
void mouseReleaseEvent(QMouseEvent *event); //释放
void mouseDoubleClickEvent(QMouseEvent *event); //双击
void mouseMoveEvent(QMouseEvent *event); //移动
void wheelEvent(QWheelEvent *event); //滑轮
```

鼠标按下事件

- 设计界面手势时常用按下事件和移动事件结合

```
//记录第一次按下坐标
QPoint mousePressPos;
void Widget::mousePressEvent(QMouseEvent *event){
 // 如果是鼠标左键按下
 if(event->button() == Qt::LeftButton){
 //记录鼠标按下的坐标
 mousePressPos=event->pos();
 }
 // 如果是鼠标右键按下
  else if(event->button() == Qt::RightButton{
  }
```

```
}
```

鼠标移动事件

- 默认情况下，触发事件需要点击一下，才能触发。可设置为自动触发:**setMouseTracking(true);**

```
//记录偏移量
double xoff,yoff;
void Widget::mouseMoveEvent(QMouseEvent *event){
 //计算鼠标移动后的坐标与之前按下鼠标的坐标偏移量
 if(event->buttons() & Qt::LeftButton){
 xoff=(event->x()- mousePressPos.x());
 yoff=(event->y()- mousePressPos.y());
 update();
 }
}

//获得鼠标的坐标
setMouseTracking(true)
void Widget::mouseMoveEvent(QMouseEvent *event){
  QString pos=QString("%1,%2").arg(event->pos().x()).arg(event->pos().y());
  //在工具提示中显示坐标
  QToolTip::showText(event->globalPos(),pos,this);
}

```

鼠标释放事件

```
void Widget::mouseReleaseEvent(QMouseEvent *event){
}
```

鼠标双击事件

```
void Widget::mouseDoubleClickEvent(QMouseEvent *event){
  // 如果是鼠标左键按下
```

```
 if(event->button() == Qt::LeftButton){
 }
}
```

滚轮事件

```
double scale;
void Widget::wheelEvent(QWheelEvent *event){
 // 当滚轮远离使用者时delta值为滚动的角度默认一下15°
 if(event->delta()>0){
 zoomOut();
 }else{//当滚轮向使用者方向旋转时
 zoomIn();
 }
}
 void Widget::zoomIn(){
  scale+=0.2;
  update();
 }
 void Widget::paintEvent(Qevent*event){
  //乘以scale比例系数重绘图像
 }
```

键盘事件

键盘事件使用时，加头文件**#include <QKeyEvent>**

```
 void keyPressEvent(QKeyEvent *event);
 void keyReleaseEvent(QKeyEvent *event);
```

键盘按下事件

```
void Widget::keyPressEvent(QKeyEvent *event){
 // 是否按下Ctrl键 特殊按键
 if(event->modifiers() == Qt::ControlModifier){
 // 是否按下M键 普通按键 类似
```

```
 if(event->key() == Qt::Key_M)
 ...
 }
 else QWidget::keyPressEvent(event); //保存默认事件

  //如果是处理两个普通按键，得避免自动重复，释放中也要处理
  if(event->key() == Qt::Key_Up){
  // 按键重复时不做处理
  if(event->isAutoRepeat()) return;
  // 标记向上方向键已经按下
  keyUp = true;
  }else if(event->key() == Qt::Key_Left){
  if(event->isAutoRepeat()) return;
  keyLeft = true;
  }
}
```

按键释放事件

```
void Widget::keyReleaseEvent(QKeyEvent *event){
  //如果是处理两个普通按键，得避免自动重复
  if(event->key() == Qt::Key_Up){
  if(event->isAutoRepeat()) return;
  }
  else if(event->key() == Qt::Key_Left){
  if(event->isAutoRepeat()) return;
  }
}
```

设置鼠标样式

```
  //设置为不按下鼠标键触发moveEvent
  this->setMouseTracking(true);
  void mouseMoveEvent(QMouseEvent* event){
```

```
4   QPoint mousepos = event()->pos();
5   //在坐标（0 ~ width，0 ~ height）范围内改变鼠标形状
6   if(mousepos.x() > 0 && mousepos.x() < width &&
7   mousepos.y() > 0 && mousepos.y() < height){
8   this->setCursor(Qt::CrossCursor);
9   }else{//范围之外变回原来形状
10   this->setCursor(Qt::ArrowCursor);
11   }
12  }
```

Qt::ArrowCursor

Qt::UpArrowCursor

Qt::CrossCursor

Qt::IBeamCursor

Qt::WaitCursor

Qt::BusyCursor

Qt::ForbiddenCursor

Qt::PointingHandCursor

Qt::WhatsThisCursor

Qt::SizeVerCursor

Qt::SizeHorCursor

Qt::SizeBDiagCursor

Qt::SizeFDiagCursor

Qt::SizeAllCursor

Qt::SplitVCursor

Qt::SplitHCursor

Qt::OpenHandCursor

Qt::ClosedHandCursor

箭头为空

Qt::BlankCursor