# 1. 信号和信号处理特性

信号处理器的声明语法为：on<Signal> 信号处理器必须在发出信号的对象的定义中进行声明。

   Rectangle{  
   id:rect  
   MouseArea{  
          anchors.fill:parent  
          onClicked:{  
 rect.color = "red"  
         }  
       }  
   }

## 1.1 定义信号特性

Item{  
 signal clicked  
    signal hovered(）  
    signal action(string action,var actionName)  
 }

## 1.2 属性值改变信号

除了自定义信号，QML类型还提供了一种内建的属性值改变信号。当属性值发生改变时，QML会自动发出该信号。

## 1.3 信号处理器

信号处理器是一类特殊的方法特性。当对应的信号发射时，信号处理器会被QML引擎自动调用。 ​ 在QML的对象定义中添加一个信号，则自动在对象定义中添加一个相应的对象处理器，只不过其中没有具体的对象代码。

//SquareButton.qml  
 import QtQuick 2.2  
 Rectangle{  
   id:root   
        signal activated(real xPosition,real yPosition)  
        signal deactivated  
   }  
    MouseArea{  
   anchors.fill:parent  
        onPressed:root.activated(mouse.x,mouse.y)  
        onReleased:root.deactivated()  
   }

这些信号可以被与SquareButton.qml同目录下的其他QML文件接收，

import QtQuick 2.2  
 SquareButton{  
 onActivated:console.log("Activated at"+  
   xPosition + "," + yPosition)  
 onDeactivated:console.log("Deactivated!")                           
 }

## 1.4 使用Connections类型

有时候可能需要在发射信号的对象外部使用这个信号。Connections类型用于连接外部对象的信号。Connections对象可以接收指定目标target的任意信号。

import QtQuick 2.2  
 Rectangle{  
 id:rect  
    MouseArea{  
        id:mouseArea  
        anchosrs.fill:parent  
   }  
    Connections{  
        target:mouseArea  
        onClicked:rect.color:"red"  
   }  
 }

这段代码中，Connections的target属性是mouseArea,因而这个Connections对象可以接收来自mouseArea的任意信号。 如果要把一个信号与一个或多个方法或者信号关联起来，需要使用connect()函数，支持将信号与一个方法或者另外的信号连接起来。

import QtQuick 2.2  
 Rectangle{  
    id:relay  
    signal messageReceived(string persion,string notice)  
    Component.onCompleted:{  
        relay.messageReceived.connect(sendToPost)  
        relay.messageReceived("Tom","Happy Birthday")  
   }  
    function sendToPost(person,notice){  
        console.log("sending to post:" + person + "," + notice)  
   }  
 }

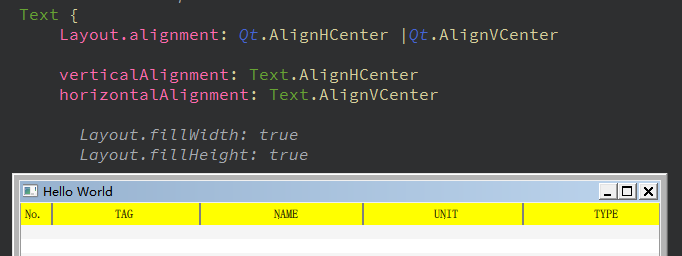
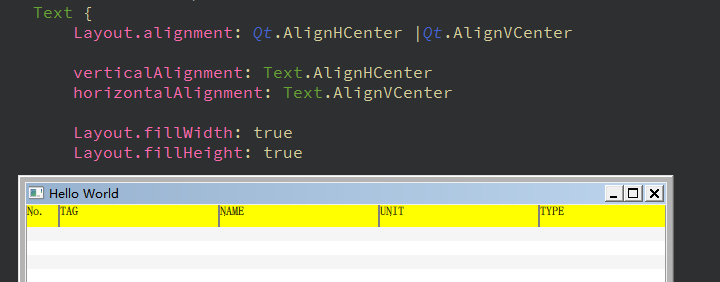
使用disconnect()函数可以解除连接。 使用connect()函数可以构成一个信号链，如下：

improt QtQucik 2.2  
 Rectangle{  
    id:forwarder  
    signal send  
    onSend:console.log("Send clicked")  
    MouseArea{  
        id:mouseArea  
        anchors.fill:parent  
        onClicked:console.log("MouseArea clicked")  
   }  
    Component.onComplete:{  
        mouseArea.clicked.connect(send)  
   }  
 }

当MouseArea发出clicked信号时，自定义的send信号也会被自动发射

# 2.布局

## 2.1布局与文字排列方式

在Text中，当布局和字体排列方式都设置时，布局的作用被体现。

## 2.2 anchors.fill:parent 与 anchors.verticalCenter:parent.verticalCenter比较



anchors.fill:parent会拉伸空间，占满所有父控件的所有空间。

anchors.verticalCenter:parent.verticalCenter只是会将子空间布局到父控件的水平中心位置。