# 1.访问Repeater中的按钮的文本

注意：不能够根据id直接访问到Repeater的子元素！！！

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  Repeater{  id:*rep*  model:3  Button{  id:*btn*  y:index \* 60  text: "btn" + index  }  }  Button{  id:*btn2*  x:200  text: "访问"  onClicked: {  *console*.log(*rep*.itemAt(0).text)  // for(var i=0;i<rep.count;i++) //当然也可以遍历访问  // {  // console.log(rep.itemAt(i).text)  // }  }  }  } |  |

**注意使用的是Repeater的itemAt方法来获取**

# 2.访问布局的子元素

## a、Column

#### 跟Repeater不同的是可以直接利用id访问到Column的元素

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  // Repeater{  // id:rep  // model:3  // Button{  // id:btn  // y:index \* 60  // text: "btn" + index  // }  // }  Column{  id:*col1*  spacing: 5  Button{  id:*btn0*  x:200  text: "hello"  }  Text{  id:*tx1*  x:200  text: "hi"  }  Rectangle{  id:*rec*  x:200  y:180  width: 100  height: 60  color:"deeppink"  Text{  id:*txt*  text: "hey"  }  }  }  Button{  id:*btnFw*  x:200  y:400  text: "访问"  onClicked: {    //访问Column的子元组  *console*.log(*btn0*.text,*tx1*.text,*rec*.width \* *rec*.height)  // for(var i=0;i< col1.children.length;i++)//也是可以遍历的  // {  // console.log(col1.children[i].text)  // console.log(col1.children[i] instanceof Rectangle)  // }  }  }  } | 点击按钮的结果： |

注意：找到子元素后还可以使用instanceof操作符来判断它是不是指定的类型如：*console*.log(*col1*.children[*i*] instanceof Rectangle)

# 3、Column只包含多个Repeater

## 采用两层循环就可以搞定

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //Column包含Repeater  Column{  id:*col2*  spacing: 5  Repeater{  id:*rep1*  model:3  Button{  id:*btn*  text: "btn" + index  }  }  Repeater{  id:*rep2*  model:3  Text{  id:*tx1*  text: "hi"+index  }  }  }  Button{  id:*btnFw*  x:200  y:250  text: "访问"  onClicked: {    for(var *i*=0;*i*< *col2*.children.length;*i*++)  {  for(var *j*=0;*j*<*col2*.children[*i*].count;*j*++)  {  *console*.log(*col2*.children[*i*].itemAt(*j*).text)  }  }  }  }  } |  |

# 4. Column包含Repeater和其他元素

**比较复杂，使用遍历来获取Column的子控件的时候需要注意：如果一个Column有Repeater和按钮等其他控件，Repeater里面又有子控件，**

**遍历的时候Repeater也被当做子控件，而且Repeater的子控件也可以被直接访问到，这样子有一些控件就会被重复。解决办法就是设置限制条件为当一个控件表示Repeater的实例**

**的时候，我们才把它算进来。这样子就只有Repeater本身没有算进来，Repeater的子控件也可以算进来**

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //Column包含Repeater  Column{  id:*col2*  spacing: 5  Repeater{  id:*rep1*  model:3  Button{  id:*btn*  text: "btn" + index  }  }  Button{  id:*btn2*  text: "我是btn"  }  Text {  id: *txt*  text: *qsTr*("我是text")  }  }  Button{  id:*btnFw*  x:200  y:250  text: "访问"  onClicked: {    for(var *i*=0;*i*< *col2*.children.length;*i*++)  {  if(*col2*.children[*i*] instanceof Button ||*col2*.children[*i*] instanceof Text){  *console*.log(*col2*.children[*i*].text)  }  }  }  }  } | 点击按钮的效果如下，不会在有重复的控件 |

# 5、访问ListView的子控件

## 简单的ListView实例

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  //访问ListView的子控件  ListView{  id:*lv*  width: 200  height: 100  model:["cpp","python","cSharp","java"]  delegate: Text{  id:*txt*  text:modelData  }  }  Button{  id:*btnFw*  x:200  y:250  text: "访问"  onClicked: {    //访问ListView的子控件  //console.log(lv.count)    for(var *i*=0;*i*<*lv*.count;*i*++){//注意这里需要使用ListView的count属性，如果使用ListView的lv.contentItem.children.length会多出一个来哟  *console*.log(*lv*.contentItem.children[*i*].text)  }  }  }  } | 点击按钮，输出结果 |

## 比较复杂的ListView

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  // Repeater{  // id:rep  // model:3  // Button{  // id:btn  // y:index \* 60  // text: "btn" + index  // }  // }  // Column{  // id:col1  // spacing: 5  // Button{  // id:btn0  // text: "hello"  // }  // Text{  // id:tx1  // text: "hi"  // }  // Rectangle{  // id:rec  // y:180  // width: 100  // height: 60  // color:"deeppink"  // Text{  // id:txt  // text: "hey"  // }  // }  // }  //Column包含Repeater  // Column{  // id:col2  // spacing: 5  // Repeater{  // id:rep1  // model:3  // Button{  // id:btn  // text: "btn" + index  // }  // }  // Button{  // id:btn2  // text: "我是btn"  // }  // Text {  // id: txt  // text: qsTr("我是text")  // }  // }  //访问ListView的子控件，简单ListView  // ListView{  // id:lv  // width: 200  // height: 100  // model:["cpp","python","cSharp","java"]  // delegate: Text{  // id:txt  // text:modelData  // }  // }  //较为复杂的ListView  ListView{  id:*lv*  width: 200  height: 200  model:["cpp","python","cSharp","java"]  delegate: Column{  Text{  id:*txt*  text:"text "+modelData  }  Button{  id:*btn*  text:"btn "+modelData  }  }  }  Button{  id:*btnFw*  x:200  y:250  text: "访问"  onClicked: {  //访问Repeater的子元素  // console.log(rep.itemAt(0).text)  // for(var i=0;i<rep.count;i++)  // {  // console.log(rep.itemAt(i).text)  // }  //访问Column的子元组  //console.log(btn0.text,tx1.text,rec.width \* rec.height)  // for(var i=0;i< col1.children.length;i++)  // {  // //console.log(col1.children[i].text)  // console.log(col1.children[i] instanceof Rectangle)  // }  //一个Column包含两个Repeater时的遍历方法  // for(var i=0;i< col2.children.length;i++)  // {  // if(col2.children[i] instanceof Button ||col2.children[i] instanceof Text){  // console.log(col2.children[i].text)  // }  // }  //访问ListView的子控件  //console.log(lv.count)  /\*  for(var i=0;i<lv.count;i++){//注意这里需要使用ListView的count属性，如果使用ListView的lv.contentItem.children.length会多出一个来哟  console.log(lv.contentItem.children[i].text)  }\*/  //较为复杂的ListView  for(var *i*=0;*i*<*lv*.count;*i*++){  var *col* =*lv*.contentItem.children[*i*];  for(var *j*=0;*j*<*col*.children.length;*j*++)  {  *console*.log(*col*.children[*j*].text)  }  }  }  }  } | 点击按钮后输出： |