# 1.Column

## add：transition，move：transition，populate：transition，添加移动动态添加都可以设置动画效果

实例：Column里面有Repeater，点击按钮往Repeater里面动态添加按钮（这个按钮是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*("QML布局")  Column{  id:*col*  spacing: 3  Repeater{  id:*rep*  model:ListModel{  }  Button{  id:*btn*  width:100  height: 60  y:index \* 70  background: Rectangle{  color:"yellow"  }  text: name  //onClicked: console.log("iam button "+modelData)  }  }  move: Transition {  NumberAnimation { properties: "x,y"; easing.type: Easing.OutBounce }  }  //The move transition specifies how items inside a positioner will animate when they are displaced by the appearance or disappearance of other items.  add: Transition {  NumberAnimation { properties: "x,y"; easing.type: Easing.OutBounce }  }  //The add transition specifies how items will appear when they are added to a positioner.  populate: Transition {  NumberAnimation { properties: "x,y"; from: 200; duration: 100; easing.type: Easing.OutBounce }  }  }  Button{  id:*btn2*  x:*parent*.width/2  y:*parent*.height/2  width: 100  height: 60  text: "show"  onClicked: {  // rep.model.insert(0,rep.children)  *rep*.model.insert(0,{"name":*rep*.model.count})  }  } |  |

# 2、Row元素

它和Column元素类似，只不过多了两个属性，其中一个叫做layoutDirection它的值：

Qt.LeftToRight (default)和

Qt.RightToLeft

例外一个叫做**effectiveLayoutDirection**

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("QML布局")  //Row元素  Rectangle{  Row{  id:*col*  spacing: 3  leftPadding: 20  //layoutDirection: Qt.RightToLeft//修改绘制方向为从右到左  // onEffectiveLayoutDirectionChanged: {  // console.log("")  // }  Component.onCompleted: {  *console*.log(*effectiveLayoutDirection*)//使用默认绘制方向时值为0，否则为1  }  Repeater{  id:*rep*  model:ListModel{  }  Button{  id:*btn*  width:100  height: 60  background: Rectangle{  color:"yellow"  }  text: name  //onClicked: console.log("iam button "+modelData)  }  }  move: Transition {  NumberAnimation { properties: "x,y"; easing.type: Easing.OutBounce }  }  //The move transition specifies how items inside a positioner will animate when they are displaced by the appearance or disappearance of other items.  add: Transition {  NumberAnimation { properties: "x,y"; easing.type: Easing.OutBounce }  }  //The add transition specifies how items will appear when they are added to a positioner.  populate: Transition {  NumberAnimation { properties: "x,y"; from: 200; duration: 100; easing.type: Easing.OutBounce }  }  }  }  Button{  id:*btn2*  x:*parent*.width/2  y:*parent*.height/2  width: 100  height: 60  text: "show"  onClicked: {  // rep.model.insert(0,rep.children)  *rep*.model.insert(0,{"name":*rep*.model.count})  }  }  } |  |

# 3、Flow元素

和Row元素很像，但是它有一些比较特殊的地方，当父容器的宽度减小时，它会自动调整它内部的子控件的布局来适应父控件

它有一个flow属性可以控制流式编辑的方向：有两个值：

Flow.LeftToRight (default)

Flow.TopToBottom

## 默认布局方式是从左到右

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("QML布局")  Rectangle{  width: 400  height: 400  border.color: "green"  Flow{ //没有指定布局方式，默认从左到右  anchors.fill: *parent*  anchors.margins: 4  spacing: 10  Text{text: "Text";font.pixelSize: 40}  Text{text: "Items";font.pixelSize: 40}  Text{text: "Flowing";font.pixelSize: 40}  Text{text: "Inside";font.pixelSize: 40}  Text{text: "a";font.pixelSize: 40}  Text{text: "Flow";font.pixelSize: 40}  Text{text: "Item";font.pixelSize: 40}  }  } |  |

## 修改布局为从上到下

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("QML布局")  Rectangle{  width: 400  height: 400  border.color: "green"  Flow{  anchors.fill: *parent*  anchors.margins: 4  flow:Flow.TopToBottom //修改流式布局的方向为从上到下  spacing: 10  Text{text: "Text";font.pixelSize: 40}  Text{text: "Items";font.pixelSize: 40}  Text{text: "Flowing";font.pixelSize: 40}  Text{text: "Inside";font.pixelSize: 40}  Text{text: "a";font.pixelSize: 40}  Text{text: "Flow";font.pixelSize: 40}  Text{text: "Item";font.pixelSize: 40}  }  } |  |

# 4、Grid元素

默认columns属性的值是4，可以修改

基本用法：

|  |  |
| --- | --- |
| //main.qml  import QtQuick 2.15  import QtQuick.Window 2.15  import QtQuick.Controls 2.15  Window {  width: 640  height: 480  visible: true  title: *qsTr*("QML布局")  Grid{  rows: 3  columns: 2 //修改显示的列，默认是4  spacing: 5  Repeater{  model:6  Rectangle{  width: 100  height: 100  color: "yellow"  border.width: 2  border.color: "lime"  }  }  }  } |  |

**注意：Grid的用法和Flow非常相似，只不过Grid可以设置显示的行数和列数，定制性较好**

**5、自定义Grid与OpacityMask的使用**

**注意使用OpacityMask需要**import QtGraphicalEffects 1.15