**GridLayout QML — Позиционирование элементов**

Евгений Легоцкой, 6 Ноябрь 2015 13:16, [Qt QML Android](http://www.evileg.ru/category/baza-znanij/qt-qml-android) http://www.evileg.ru/wp-content/themes/evileg/images/eye.png 321

Помечено: [cell](http://www.evileg.ru/tag/cell), [column](http://www.evileg.ru/tag/column), [GridLayout](http://www.evileg.ru/tag/gridlayout), [qml](http://www.evileg.ru/tag/qml), [qml уроки](http://www.evileg.ru/tag/qml-uroki), [qt](http://www.evileg.ru/tag/qt), [qt уроки](http://www.evileg.ru/tag/qt-uroki), [row](http://www.evileg.ru/tag/row), [ячейка](http://www.evileg.ru/tag/yachejka)

Небольшая заметка, родившаяся из вопроса одного из постоянных читателей сайта. При разработке интерфейса приложения под **QML** для позиционирования объектов в **GridLayout** необходимо использовать функционал вложенных свойств **Layout.**Таких как:

* **Layout.row** — указывает строку, в которой располагается объект;
* **Layout.column** — указывает колонку, в которой располагается объект;
* **Layout.rowSpan** — указывает, на сколько строк должен быть растянут объект;
* **Layout.columnSpan** — указывает, на сколько колонок должен быть растянут объект;
* **Layout.minimumWidth** — минимальная ширина объекта в слое;
* **Layout.minimumHeight** — минимальная высота объекта в слое;
* **Layout.preferredWidth** — предпочтительная ширина объекта в слое;
* **Layout.preferredHeight** — предпочтительная высота объекта в слое;
* **Layout.maximumWidth** — максимальная ширина объекта в слое;
* **Layout.maximumHeight** — максимальная высота объекта в слое;
* **Layout.fillWidth** — заполнение по ширине;
* **Layout.fillHeight** — заполнение по высоте;
* **Layout.alignment** — выравнивание в слое;

**Свойства GridLayout**

Также для полноты картины укажу свойства **GridLayout** и за что они отвечают:

* **columnSpacing** : real — пробелы между колонками (разрыв между объектами);
* **columns** : int — количество колонок;
* **flow** : enumeration — направление расположения объектов в GridLayout
  + GridLayout.LeftToRight (default) — слева направо
  + GridLayout.TopToBottom — сверху вниз
* **layoutDirection** : enumeration — направление перечисления объектов при заданном flow
  + Qt.LeftToRight (default) — слева направо
  + Qt.RightToLeft — справа налево
* **rowSpacing** : real — пробелы между строками (разрыв между объектами);
* **rows** : int — количество строк;

Если задаём:

* flow: GridLayout.TopToBottom
* layoutDirection: Qt.RightToLeft

То в результате получим, что первый элемент строки будет в самом вверху, а последний элемент строки будет в самом низу. В случае наличия двух строк, элементы первой строки будут располагаться справа.

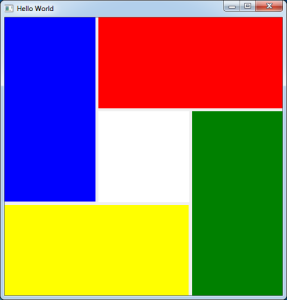
**Работа с GridLayout**

Для демонстрации позиционирования элементов в слое предлагаю следующий программный код. В котором несколько прямоугольников будут растягиваться на несколько ячеек **GridLayout.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68 | import QtQuick 2.5  import QtQuick.Controls 1.4  import QtQuick.Layouts 1.1    ApplicationWindow {      visible: true      width: 480      height: 480      title: qsTr("Hello World")       GridLayout {         id: grid         anchors.fill: parent           rows: 3         columns: 3           Rectangle {              color: "red"              Layout.fillHeight: true              Layout.fillWidth: true              Layout.columnSpan: 2              Layout.rowSpan: 1              Layout.row: 1              Layout.column: 2         }           Rectangle {              color: "blue"              Layout.fillHeight: true              Layout.fillWidth: true              Layout.columnSpan: 1              Layout.rowSpan: 2              Layout.row: 1              Layout.column: 1         }           Rectangle {              color: "green"              Layout.fillHeight: true              Layout.fillWidth: true              Layout.columnSpan: 1              Layout.rowSpan: 2              Layout.row: 2              Layout.column: 3         }           Rectangle {              color: "white"              Layout.fillHeight: true              Layout.fillWidth: true              Layout.columnSpan: 1              Layout.rowSpan: 1              Layout.row: 2              Layout.column: 2         }           Rectangle {              color: "yellow"              Layout.fillHeight: true              Layout.fillWidth: true              Layout.columnSpan: 2              Layout.rowSpan: 1              Layout.row: 3              Layout.column: 1         }     }  } |

**Итог**

Результат работы данного программного кода показан на рисунке. Также выше перечисленная логика расположения объектов применима и к другим объектам **Layout.**

[](http://www.evileg.ru/wp-content/uploads/2015/11/2015-11-06-15-46-20-Skrinshot-ekrana.png)

**Поделиться ссылкой:**