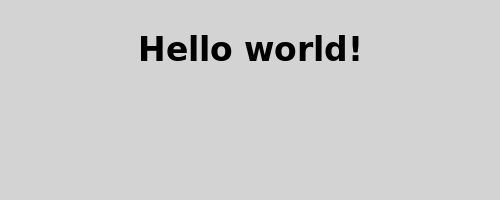
**QML入门教程(1)**

**QML是什么？**

QML是一种描述性的脚本语言，文件格式以.qml结尾。语法格式非常像CSS(参考后文具体例子)，但又支持javacript形式的编程控制。它结合了QtDesigner UI和QtScript的优点。QtDesigner可以设计出.ui界面文件,但是不支持和Qt原生C++代码的交互。QtScript可以和Qt原生代码进行交互，但是有一个缺点，如果要在脚本中创建一个继承于QObject的图形对象非常不方便，只能在Qt代码中创建图形对象，然后从 QtScript中进行访问。而QML可以在脚本里创建图形对象，并且支持各种图形特效，以及状态机等，同时又能跟Qt写的C++代码进行方便的交互，使用起来非常方便。

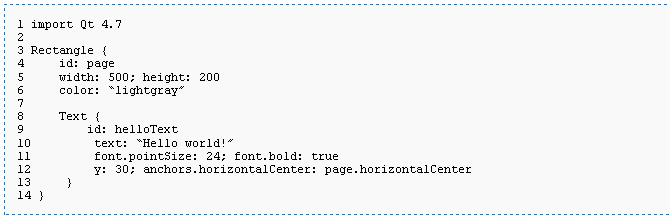
**如何使用？**

在Qt C＋＋文件中通过QDeclarativeView加载，就像使用UiLoader加载.ui文件一样。不过本文不会去介绍如何在Qt C＋＋中使用QML，而是把重点放在QML的语法上，不过别担心看不到.qml文件的效果。Qt提供了一个工具QML Viewer可以查看.qml文件生成的效果，该程序在Qt的bin目录下，应用名字叫qml(Windows下叫qml.exe)。所以你在看到别人提供的.qml文件时，你可以用下面命令**qml filename.qml** 查看.qml的执行结果，咱们的第一个Hello,World生成界面如下



开始QML吧

上面的Hello,World源代码如下



第1行是Qt QML的统一用法，指明当前QML会使用Qt-4.7里已经定义好的类型，比如第3行的Rectangle和第8行的Text。

第3行开始至文章结束处则定义了一个矩形的图形区域对象，第4行则申明了该矩形区域对象的id为”page”可以被其它对象识别并通过该id访问其成员属性，另外几个属性width/height/color则很好理解，语法跟CSS类似，可以写在一行中用分号”;”隔开。

第8行至第12行则是一个文本对象，看它代码的嵌套结构可以知道它是内嵌于Rectangle的。Text的属性除了anchors其它几个都能顾名思义。anchors描诉的是当前对象的位置和其它对象的相对关系，这里看到其水平中心位置在“page“的水平中心位置。如果想对anchors了解更多，请参考锚的解释。

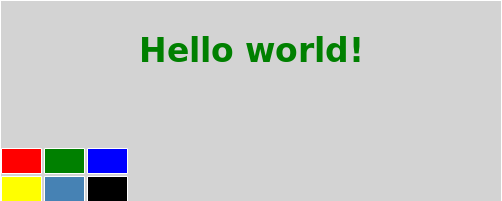
以上就是Hello,World的全部代码，将其存为hellowordl.qml，那么只要执行**qml hellowrold.qml**就能看到文章前头的界面了。

**更多对象？**

在上面的代码中我们用到了Rectangle和Text，那么我还有哪些对象以及这些对象有哪些属性呢。那么请访问QML-Item类,Item类是QML最基础的类，通过查看它的继承类以及这些继承类可用的属性，你可以添加更多你感兴趣的内容。

**QML入门教程(2)**

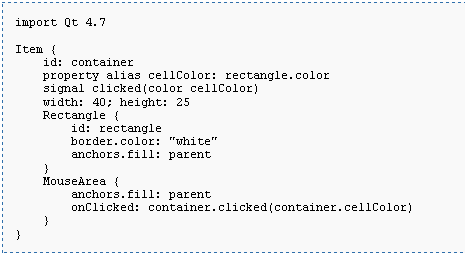
在上一篇文章里我们使用了最基础的QML类型实现了文字Hello,World的显示。这篇文章中将会增加颜色选项面板，用户可以给 Hello,World设置不同的颜色，如下图显示



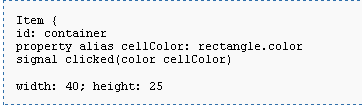
**QML组件**

从图中可以看到选项面板由6个颜色小块组成，它们唯一的区别就是颜色不一样。那么我们就可以用组件(Component)来实现一个颜色块，然后在需要的时候使用这个组件就可以了。组件其实和其它编程语言中的宏，函数，类，结构体等功能差不多，就是代码复用。作为程序员，我知道你懂的。组件由一个单独的QML文件名组成，文件名总是以大写字母开头，要使用该组件的时候直接使用该文件名就可以了。关于如何定义自己的组件，请访问 Defining new Components。我们为一个颜色块定义了一个Cell.qml文件，然后使用Cell作为一个去访问它。

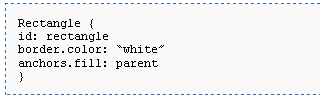
**Cell.qml的内容**



挨个看代码～



Item是最常使用的QML类型，一般用作其它类型的容器，可以理解成最顶级的父类，功能类似于QtGui中的QWidget。用一个属性别名访问其内嵌对象rectangle的color属性。在其它文件中可以用Cell对象的cellColor获得rectangle的color值。 signal clicked(color cellColor)则为对象定义了一个信号，在代码的其它部分可以发出这个信号。



这一部分没有特别好说的，在Item中内嵌了一个id为rectangle白边框的矩形区域，大小占满整个Item。



MouseArea则为Item增加了一块鼠标响应区，看它的anchors知道，在整个Item区域内都是鼠标活动区，都能侦听到鼠标事件。 onClicked那一行则相当于为鼠标单击事件增加了一个处理行为，这里是发出了一个clicked()的信号。这个信号正是我们在Item里定义的那 个signal。 Cell.qml写完了，再来看看程序的主文件。

**main.qml的内容**

****

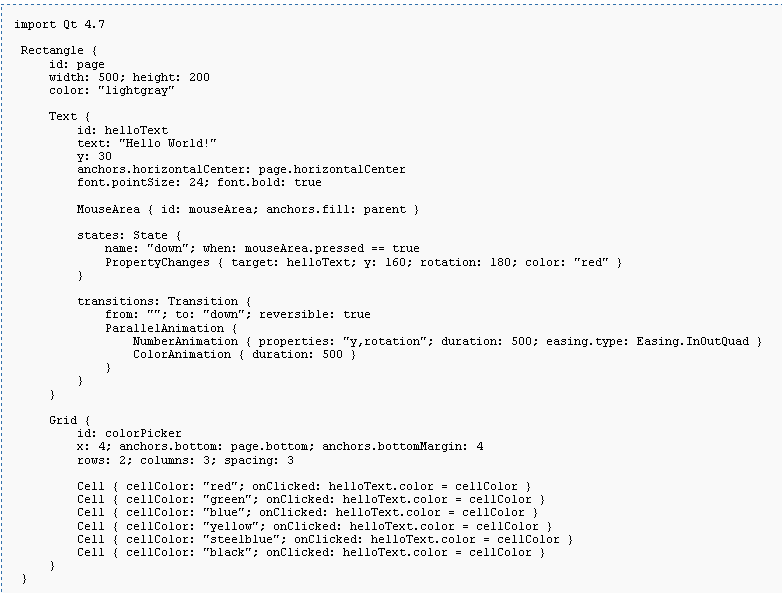
这里在原来的基础上增加了一个Grid网格。x坐标是4，底部挨着page的底部，所以我们看到的是在左下角。新增的6个Cell，名字和Cell.qml是一样的。通过cellColor属性将颜色传给了每个颜色块。当Cell接收到onClicked事件的时候,关联的代码回去修改Hello,World上的颜色。细心的朋友可能会注意到Cell只是定义了 clicked()的信号，并没有定义onClicked()啊，是的这就是Component的语法规则了。如果你在Cell.qml里定义的是 clicked(),那么你在main.qml中引用的时候就该用onClicked()了。

**QML入门教程(3)**

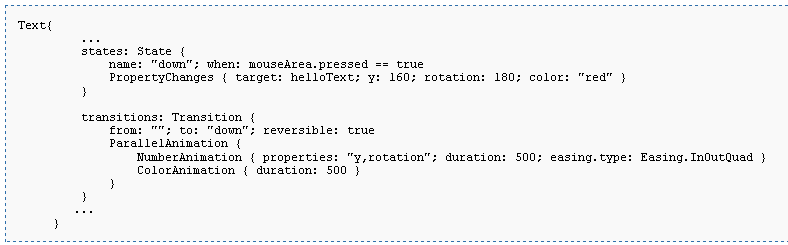
经过前面两个教程，文字也能显示，也能处理鼠标事件了，来点动画吧.

这个教程实现了当鼠标按住的时候，Hello,World从顶部到底部的一个旋转过程，并带有颜色渐变的效果。

**完整的源代码main.qml**

****

除了这个main.qml之外，还有一个Cell.qml也是需要的，和教程(2)中的完全一样。下面来看一看比起教程(2)的代码增加出来的内容。

****

tates内嵌于Text之中，可以为Text元素添加多个状态，现在的这个例子只增加了一个状态。该状态的名为为”down”,然后由“when”指 定了什么时候触发这个状态。PropertyChanges则指定了哪个元素的哪些属性会发生什么样的变化。例子中PropertyChanges利用 “target”指定了id为”helloText”的元素会发生变化，包括其y,rotation,color等属性。  
transitions 是用于增加动画效果的（如果把transitions这一段代码删去，Hello,World的文字也会发生变化,但是看不到中间动画渐变效果)。同样可 以看到transitions是复数形式，意味着可以添加多个动画过程。“from”和”to”指明了当前的动画作用于哪两个状态变化之间。 “from”和”to”的参数名来自于State中的”name”属性。  
ParalleAnimation则指定了有多个 有多个动画并行发生。NumberAnimation用于qreal类型的属性变化,ColorAnimation则用于颜色变化。更多 Animation请在QML文档中查找”Animation and Transitions”。

More Information, please refer to to:

<http://doc.qt.nokia.com/4.7-snapshot/qml-tutorial.html>

<http://doc.qt.nokia.com/4.7-snapshot/qml-item.html>