使用 Qt 开发的 Android 应用，怎样适应 Android 智能手机各种各样的屏幕尺寸？

    说到屏幕尺寸，从 2.8 吋到 8.9 吋的手机屏幕都有，这对程序猿们来讲痛苦可不只一点。 Android 项目本身已经考虑了这个问题，资源文件，比如图标，都有 ldpi / mdpi / hdpi / xhdpi 等等版本， Android 框架会根据屏幕大小自动选择相应的图标，这样在不同尺寸的屏幕上，应用看起来就差不多了。

    那 Qt 应用呢？其实不大用得上 Android 的这种机制（惟有 App Icon 可以搭便车），一切都得自己处理了。那怎么处理呢？

    首先要理解 DPI ，然后是字体大小。

**DPI 与字体大小**

    DPI ， dot per inch ，即每英寸包含的点数。还有一个概念是 PPI ，即每英寸包含的像素数。一般我们用 DPI 就够了，对于专业人士处理超高 DPI 的场景，使用 PPI 可能更精确一些。在 Qt 中，只有 DPI ，所以我们单说它吧。

    这个值越大，像素密度越大，小尺寸的屏幕就可以有大分辨率。比如有的 Android 手机， 3.7 吋屏幕就能提供 960x540 的分辨率，而有的手机， 5 吋屏幕却提供 800x480 的分辨率。这两种不同屏幕的尺寸和分辨率的手机，5 吋屏看起来会有颗粒感，而 3.7 吋看起来则非常细腻。这就是像素密度带来的差别。

    有的屏幕，横向的 DPI 和纵向的 DPI 不一样，即像素点不是正方形，这就更复杂了……

    我们在写应用时，理想的情况是：应当根据 DPI + 屏幕分辨率来设置界面元素的大小。

**QScreen 类**

    在 Qt 中， QScreen 类可以获取到 DPI 相关的信息。

    QScreen 的 physicalDotsPerInch / physicalDotsPerInchX / physicalDotsPerInchY 这一组属性表示物理 DPI 。  logicalDotsPerInch / logicalDotsPerInchX / logicalDotsPerInchY 这一组属性表示逻辑 DPI ， Qt 使用它来计算字体大小，我们可以用它将字体的 pointSize 转换为 pixelSize 。下面咱们就来说它。

    logicalDotsPerInch 是一个 X 、 Y 的简单平均值，多数情况下就够用了，当然如果你有极致追求，请问道于 logicalDotsPerInchX 和 logicalDotsPerInchY 。

**QFont 类**

    QFont 代表字体，字体的大小有两种表示方式： pixelSize 和 pointSize 。即像素大小和点阵大小。如果你使用像素大小来表示字体，那字体将不受 DPI 的影响，在电脑上你可以调整显示器的 DPI 来观察界面的变化。但这不适用于移动场景中适配多样化屏幕尺寸的要求。在针对 Android 设备开发时，我们应当使用字体的 pointSize ，这也是 Qt 应用的默认处理方式。

    废话了不是，默认就是 pointSize ，还啰嗦个甚！

    非也非也！且往下看。

**Qt 中的控件**

    Qt 中有 QLabel / QPushButton / QListWidget / QTabelWidget 等等可以在 Android 设备上使用的控件，它们可以用来显示文本。你找来一堆不同尺寸屏幕的手机，使用 QFont 的 setPointSize() 方法调整一下字体的点阵大小，权衡一下效果，就可以决定你的应用的字体尺寸如何设置了。

    但还有非文本的场景，比如你是图片按钮，那怎么办呢？

    我的答案是：根据字体的点阵大小计算出像素大小，然后拿这个来调整非文本控件的大小。这样子界面上的文本元素和图片等非文本元素才可以匹配起来。

    从 pointSize 到 pixelSize 的计算公式： pixelSize = DPI \* pointSize/72 。

**Qt 应用如何使用 DPI**

    看一个代码片段：

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/foruok/article/details/38510195)

1. **int** main(**int** argc, **char** \*\*argv)
2. {
3. QApplication a(argc, argv);
4. ...
5. QScreen \*screen = a.primaryScreen();
6. QFont f = a.font();
7. **int** pixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;
8. /\*
9. f.setPointSize(25);
10. a.setFont(f);
11. int newPixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;
12. \*/
13. ...
14. }

int main(int argc, char \*\*argv)

{

QApplication a(argc, argv);

...

QScreen \*screen = a.primaryScreen();

QFont f = a.font();

int pixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;

/\*

f.setPointSize(25);

a.setFont(f);

int newPixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;

\*/

...

}

    其实很简单，只要设置了 QApplication 的 字体，你应用的所有界面元素的字体大小都会变。如果你想单独设置某个 Widget 的字体，可以针对它调用 setFont() 方法。

    上面的代码还演示了从 pointSize 到 pixelSize 的换算，一旦你得到了合适的 pixelSize ，就可以以它为基础来设置非文本界面元素的尺寸。

**Qt 获取屏幕分辨率**

    前面我们说最好结合分辨率和 DPI ，一起来调整界面元素。 DPI 的使用已经简单介绍过了，剩分辨率了。

    QScreen 类的 size 属性可以返回屏幕的像素尺寸， availableSize 可以返回应用能够使用的尺寸。两者的区别是， availableSize 移除了窗口管理器占用的尺寸（在电脑上就是任务栏， Android 手机上是状态栏之类的区域）。

    下面是一个简单的示例代码：

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/foruok/article/details/38510195)

1. #ifdef ANDROID
2. QSize iconSize(32, 32);
3. ...
4. QScreen \*screen = qApp->primaryScreen();
5. QFont f = qApp->font();
6. **int** pixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;
7. QSize screenSize = screen->size();
8. if(screenSize.width() > 960 || screenSize.height() > 960)
9. {
10. iconSize \*= ((qreal)pixelSize) / 20;
11. }
12. #endif

#ifdef ANDROID

QSize iconSize(32, 32);

...

QScreen \*screen = qApp->primaryScreen();

QFont f = qApp->font();

int pixelSize = (f.pointSize() \* screen->logicalDotsPerInch()) / 72;

QSize screenSize = screen->size();

if(screenSize.width() > 960 || screenSize.height() > 960)

{

iconSize \*= ((qreal)pixelSize) / 20;

}

#endif

    我们知道该如何设置 Qt on Android 应用来适配多样化的屏幕尺寸，但你的应用运行后是什么效果，就看你如何综合使用前面介绍的这些内容了……