1. CrossApp适配解决方案

屏幕适配没有绝对完美的方案，不可能做到一套适配方案适用所有设备。当一个app运行在不同设备时，由于设备的物理尺寸和分辨率不尽相同，所以某个界面里的一些UI元素可能会发生变化，很多情况下需要开发者自己去控制，并且在程序中我们应尽量使用相对坐标。

我们的app运行到PPI较低的设备时，界面可能会被放大，当运行到PPI较高的设备时界面又可能会被缩小。但实际上，有时我们希望某些UI元素保持不变，因此对于某些大小锁定的UI元素来说，我们需要固定它在不同设备上显示都保持一样的大小，比如小说阅读器里的文本字体大小，并且对于屏幕尺寸大的设备应该显示更多的内容，例如我们可以固定tableView的每个cell的大小，从而在尺寸较大的屏幕上显示更多的cell，对此，CrossApp提供了一种辅助的适配方案。

以iphone4和iphone5为基准进行适配，以下是相关参数。

|  |  |  |  |
| --- | --- | --- | --- |
| 设备 | 屏幕尺寸 | 分辨率 | PPI |
| iphone4 | 3.4英寸 | 960\*640 | 326 |
| iphone5 | 4.0英寸 | 1136\*640 | 326 |

CrossApp辅助适配方案适配的原则是尽量保持UI的物理尺寸相近，但仍旧可能会有细微变化，具体的变化值由系统决定。在不同设备下，ppi值会有所不同，在保持UI的物理尺寸不变的情况下，在不同设备上的显示效果会有所差别。屏幕尺寸大的设备，显示的内容会相对较多，ppi值越高，则画面越精细。

这里有一个修正系数的概念，因为以iphone4和iphone5为基准，所以iphone4和iphone5的修正系数为1，但对于其他设备，这个系数是个近似值，当我们设置UI的size时，如果需要保持其在不同设备上的显示大小一样，通常需要乘以这个修正系数，CrossApp会根据不同设备的规格自动计算这个修正系数的值。使用的时候我们只需要把需要设定的数值传递给内联函数\_px就可以保证在不同设备上的UI物理尺寸不变。例如，设定button的物理尺寸为100\*50，

CADipSize size = typeView->getBounds().size;

CAButton\* squareRectBtn = CAButton::create(CAButtonTypeSquareRect);

squareRectBtn->setCenter(CADipRect(size.width\*0.5, size.height\*0.5, \_px(100), \_px(50)));

squareRectBtn->setTitleForState(CAControlStateAll, "SquareRect");

this->getView()->addSubview(squareRectBtn);

这里涉及到PPI和DPI的概念：

PPI：每英寸所包含的像素，在iOS设备上，通常我们以PPI来表示设备屏幕的显示精细度。

DPI：每英寸所包含的点，在Android设备上，通常以DPI来表示设备屏幕的显示精细度。通常情况下，PPI和DPI可以表示同一个概念，PPI主要针对显示设备，DPI更多应用于打印输出设备，但都表示每英寸所包含的像素点。