Public class View extends Object

1. 概览

这个类表示用户界面组件的基本构建模型（矩形）。一个View占据了屏幕上的一个矩形区域，负责显示(绘画)内容和处理用户输入事件。

所有用于创建交互式UI的组件(Button,TextView)，都继承于View类。

而ViewGroup的子类，对用户来说都不可见，都是用来布局的。它们都可以包含多个Views和ViewGroups，并且定义了这些子元素的布局属性。

1. 使用方法

在一个窗口中显示的所有View，都被组织成一棵单一的树。你可以通过编写代码的方式为这棵树添加Views；也可以通过配置XML布局文件的方式，描述这棵View树。

有很多View的子类，专门用于控制和显示文字、图片等内容。

当你要创建了一个View树，有几类通用的典型操作。

第一类，设置属性。例如设置TextView的文字。对于不同的组件，其可设置的属性和设置方法可能千差万别。注意，在组件创建时期可以用到的属性就可以在XML布局文件中配置。

第二类，设置焦点。框架将移动焦点，用于响应用户的输入。可以通过View的requestFocus()方法，强制将焦点获取给该View。

第三类，设置监听器。

第四类，设置可见性。可以通过setVisibility()来设置View的可见性.

1. View的定位

View的几何模型是一个矩形。

View的位置是由(left,top)的坐标来描述的。

View的大小由宽和高来表示

这些几何参数的单位都是像素px

我们可以通过View的getWidth和getHeight来取回该View的尺寸

1. 尺寸，内边距以及外边距

一个View的尺寸是由宽和高来描述的。但实际上，一个View拥有两

组尺寸属性：

第一组叫做measuredHeight和measuredWidth(测量出的)。它们表示了该View在其父控件中所需占用的空间。这类尺寸可以通过getMeasuredWidth() and getMeasuredHeight().方法来获得

第二组叫做Height和Width。有时也叫drawing width and drawing height。它们表示了该View在屏幕上的实际尺寸(计算时间再at drawing time and after layout)

1. Layout（布局）

布局包含两个步骤，a measure pass 和 a layout pass。

测量步骤是由int measure(int,int)方法来实现的。采用先到先得的策略计算每一个View的尺寸。在 measure pass 结束后，View树中的每一个View都已经保存了其相关显示尺寸。

显示步骤是由layout(int,int,int,int)方法来实现。在这一过程中，每一个父控件负责根据在measure pass中计算出的尺寸参数，放置每一个子元素。一个父控件，可能会多次调用measure()方法，具体调用的次数是由系统决定的。

measure pass 使用两个类来表示尺寸。View.MeasureSpec类是用来告诉其父控件，它们想如何被测量和放置。另一个类时LayouParams。

View.MeasureSpec封装了父控件的相关限制条件，通过该类的getSize()方法可以得到实际的measuredSize.