“深度”定制自定义组件

1.1 知识解析

在 Android 中，可以用自定义的要求轻松地建立自定义组件，一般情况下，只需要根据需要继承现有的 view 并作出相关的修改即可。下面将简单介绍如何创造定制化组件：

最为通用的毫无疑问是 View 类（或者其子类，也可以像前面一样，使用 ViewGroup 及其子类），因此，最开始要建立一个基于此类的一个子类。

写一个方法从 XML 文件中取得属性和参数，当然也可以自行定义这些属性和参数。

可能有必要编写本身的事件监听器、修改方法和一些组件本身功能上的程序代码。

如果希望组件能够显示什么东西，可能会重载 onMeasure() /onDraw()方法。当两个方法都用默认的，那 onDraw() 方法将不会做任何事情，并且预设的 onMeasure() 方法会自动的设定一个 100x100 的尺寸，这个尺寸可能并非我们所要的。

其他有必要重载的 on...开头 系列方法都需要重新编写一次。

onDraw()方法将会传送一个 Canvas 对象，通过它即可在 2D 图形上做任何事情，包含其他的一些标准和通用的组件、格式化本文，任何可以想到的东西都可以通过它呈现。

但此不包含 3D 图形,如果想使用 3D 图形,应该把父类由 View 改为 SurfaceView 类，并使用独立的线程。

改变自定义 view 属性后，需要重新绘制，则需要调用 requestLayout()和 invalidate()方法(注意顺序),让它通知系统重绘,不要自己去调用回调方法 onDraw()。

onMeasure() 方法较为复杂一些,因为这个方法是呈现组件和容器互动的关键部分,onMeasure()应该重载，让它能够有效且准确的表现它所包含部分的大小的测量值。

onMeasure()方法会在绘制之前被调用，以便确定其大小

当一个 View 的父组件更新其大小的时候，View 的 onMeasure()方法会被调用。组件基于父组件的限制负责自己的大小的设置

一旦测量宽度和高度出来后，就要立即调用 setMeasuredDimension() 方法将其保存

onMeasure(int widthMeasureSpec, int heightMeasureSpec)方法的 2 个参数分别用于指定父组件所需要的水平和垂直空间的设置，它被封装在 View.MeasureSpec 的静态常量中：

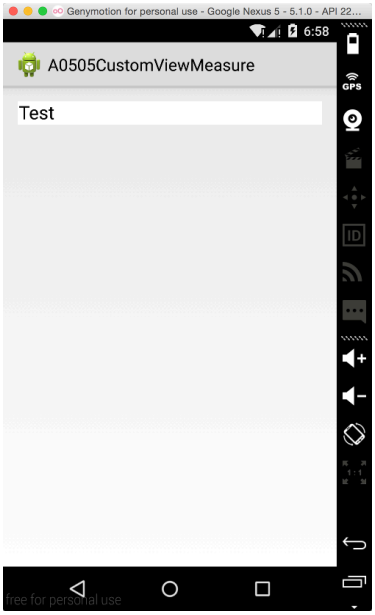
- UNSPECIFIED：不具体指定大小，view 将会设置成任何可能的大小
- AT_MOST：view 将设置成小于或者等于给定的规格的大小
- EXACTLY：view 将设置成刚好给定的 size 大小

view 的大小在其内容大小的基础上，加上一定的 padding 即可，比较常见的就是要在 View 中放置图片或者文字：

图片（Drawable）的大小可以通过 Drawable 的 getIntrinsicWidth()/getIntrinsicHeight()测量

文字宽度可以通过 Paint 的 measureText()方法计算，高度比较繁琐，通过 Paint.FontMetrics，通过它的属性 descent- ascent 计算。

1.2 功能演示



1.3 实战操作

```
@Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        // ....
        setMeasuredDimension(measureWidth(widthMeasureSpec),
measureHeight(heightMeasureSpec));
    }

    private int measureWidth(int measureSpec) {
        int result = 0;
        int specMode = MeasureSpec.getMode(measureSpec);
        int specSize = MeasureSpec.getSize(measureSpec);
        // MeasureSpec.
        if (specMode == MeasureSpec.EXACTLY) {
```

```
                // We were told how big to be
                result = specSize;
            } else {
                // Measure the text
                result = (int) mTextPaint.measureText(mText) + getPaddingLeft() +
getPaddingRight();
                if (specMode == MeasureSpec.AT_MOST) {
                    // Respect AT_MOST value if that was what is called for by
                    // measureSpec
                    result = Math.min(result, specSize);
                }
            }

            return result;
        }

        private int measureHeight(int measureSpec) {
            int result = 0;
            int specMode = MeasureSpec.getMode(measureSpec);
            int specSize = MeasureSpec.getSize(measureSpec);

            mAscent = (int) mTextPaint.ascent();
            if (specMode == MeasureSpec.EXACTLY) {
                // We were told how big to be
                result = specSize;
            } else {
                // Measure the text (beware: ascent is a negative number)
                result = (int) (-mAscent + mTextPaint.descent()) + getPaddingTop() +
getPaddingBottom();
                if (specMode == MeasureSpec.AT_MOST) {
```

```
                // Respect AT_MOST value if that was what is called for by
                // measureSpec
                result = Math.min(result, specSize);
            }
        }
        return result;
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
        canvas.drawText(mText, getPaddingLeft(), getPaddingTop() - mAscent, mTextPaint);
    }
```

1.4 职业素质

在 Android 中，可以用自定义的要求轻松地建立自定义组件，一般情况下，只需要根据需要继承现有的 view 并作出相关的修改即可。