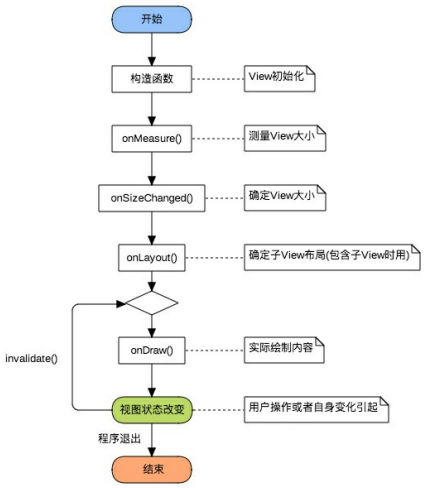
2016-01-16 • 自定义View,自定义ViewGroup,流程 • View 22476 times.

安卓自定义View进阶-分类与流程

本章节为什么要叫进阶篇？(虽然讲的是基础内容)，因为从本篇开始，将会逐渐揭开自定义View的神秘面纱，每一篇都将比上一篇内容更加深入，利用所学的知识能够制作更加炫酷自定义View，就像在台阶上一样，每一篇都更上一层，帮助大家一步步走向人生巅峰，出任CEO，迎娶白富美。~~误~~，是帮助大家更加了解那些炫酷的自定义View是如何制作的，达到举一反三的效果。

自定义View绘制流程函数调用链(简化版)



一.自定义View分类

我将自定义View分为了两类(sloop个人分类法，非官方)：

1.自定义ViewGroup

自定义ViewGroup一般是利用现有的组件根据特定的布局方式来组成新的组件，大多继承自ViewGroup或各种Layout，包含有子View。

> 例如：应用底部导航条中的条目，一般都是上面图标(ImageView)，下面文字(TextView)，那么这两个就可以用自定义ViewGroup组合成为一个Veiw，提供两个属性分别用来设置文字和图片，使用起来会更加方便。

2.自定义View

在没有现成的View，需要自己实现的时候，就使用自定义View，一般继承自View，SurfaceView或其他的View，不包含子View。

> 例如：制作一个支持自动加载网络图片的ImageView，制作图表等。

PS： 自定义View在大多数情况下都有替代方案，利用图片或者组合动画来实现，但是使用后者可能会面临内存耗费过大，制作麻烦更诸多问题。

二.几个重要的函数

1.构造函数

构造函数是View的入口，可以用于**初始化一些的内容，和获取自定义属性。**

View的构造函数有四种重载分别如下:

```
public void SloopView(Context context) {}
public void SloopView(Context context, AttributeSet attrs) {}
public void SloopView(Context context, AttributeSet attrs, int defStyleAttr) {}
public void SloopView(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {}
```

可以看出，关于View构造函数的参数有多有少，先排除几个不常用的，留下常用的再研究。

有四个参数的构造函数在API21的时候才添加上，暂不考虑。

有三个参数的构造函数中第三个参数是默认的Style，这里的默认的Style是指它在当前Application或Activity所用的Theme中的默认Style，且只有在明确调用的时候才会生效，以系统中的ImageButton为例说明：

```
public ImageButton(Context context, AttributeSet attrs) {
    //调用了三个参数的构造函数，明确指定第三个参数
    this(context, attrs, com.android.internal.R.attr.imageButtonStyle);
}

public ImageButton(Context context, AttributeSet attrs, int defStyleAttr) {
    //此处调了四个参数的构造函数，无视即可
    this(context, attrs, defStyleAttr, 0);
}
```

注意：即使你在View中使用了Style这个属性也不会调用三个参数的构造函数，所调用的依旧是两个参数的构造函数。

由于三个参数的构造函数第三个参数一般不用，暂不考虑，第三个参数的具体用法会在以后用到的时候详细介绍。

排除了两个之后，只剩下一个参数和两个参数的构造函数，他们的详情如下：

```
//一般在直接New一个View的时候调用。
public void SloopView(Context context) {}

//一般在layout文件中使用的时候会调用，关于它的所有属性(包括自定义属性)都会包含在attrs中传递进来。
public void SloopView(Context context, AttributeSet attrs) {}
```

以下方法调用的是一个参数的构造函数：

```
//在Avtivity中
SloopView view  new SloopView(this);
```

以下方法调用的是两个参数的构造函数：

```
//在layout文件中 - 格式为: 包名.View名
<com.sloop.study.SloopView
  android:layout_width"wrap_content"
  android:layout_height"wrap_content"/>
```

关于构造函数先讲这么多，关于如何自定义属性和使用attrs中的内容，在后面会详细讲解，目前只需要知道这两个构造函数在何时调用即可。

GcsSloop

Just do IT later.

嗨，我是 GcsSloop，一名来自2.5次元的魔法师，
Android自定义View系列文章作者，非著名程序员。

欢迎来到我的魔法世界!

博客 GitHub 目录 专栏
自定义控件 关于

2.测量View大小(onMeasure)

Q: 为什么要测量View大小？

A: View的大小不仅由自身所决定，同时也会受到父控件的影响，为了我们的控件能更好的适应各种情况，一般会自己进行测量。

测量View大小使用的是onMeasure函数，我们可以从onMeasure的两个参数中取出宽高的相关数据：

```
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    int widthsize  MeasureSpec.getSize(widthMeasureSpec);       //取出宽度的确切数值
    int widthmode  MeasureSpec.getMode(widthMeasureSpec);       //取出宽度的测量模式

    int heightsize  MeasureSpec.getSize(heightMeasureSpec);     //取出高度的确切数值
    int heightmode  MeasureSpec.getMode(heightMeasureSpec);     //取出高度的测量模式
}
```

从上面可以看出 onMeasure 函数中有 widthMeasureSpec 和 heightMeasureSpec 这两个 int 类型的参数， 毫无疑问他们是和宽高相关的， **但它们其实不是宽和高， 而是由宽、高和各自方向上对应的测量模式来合成的一个值：**

测量模式一共有三种， 被定义在 Android 中的 View 类的一个内部类View.MeasureSpec中：

模式	二进制数值	描述
UNSPECIFIED	00	默认值，父控件没有给子view任何限制，子View可以设置为任意大小。
EXACTLY	01	表示父控件已经确切的指定了子View的大小。
AT_MOST	10	表示子View具体大小没有尺寸限制，但是存在上限，上限一般为父View大小。

在int类型的32位二进制位中，31-30这两位表示测量模式,29~0这三十位表示宽和高的实际值，实际上如下：

以数值1080(二进制为: 1111011000)为例(其中模式和实际数值是连在一起的，为了展示我将他们分开了)：

模式名称	模式数值	实际数值
UNSPECIFIED	00	00000000000000000000001111011000
EXACTLY	01	00000000000000000000001111011000
AT_MOST	10	00000000000000000000001111011000

PS: 实际上关于上面的东西了解即可，在实际运用之中只需要记住有三种模式，用 MeasureSpec 的 getSize是获取数值， getMode是获取模式即可。

注意：
如果对View的宽高进行修改了，不要调用 super.onMeasure(widthMeasureSpec, heightMeasureSpec); 要调用 setMeasuredDimension(widthsize, heightsize); 这个函数。

3.确定View大小(onSizeChanged)

这个函数在视图大小发生改变时调用。

Q: 在测量完View并使用setMeasuredDimension函数之后View的大小基本上已经确定了，那么为什么还要再次确定View的大小呢？

A: 这是因为View的大小不仅由View本身控制，而且受父控件的影响，所以我们在确定View大小的时候最好使用系统提供的onSizeChanged回调函数。

onSizeChanged如下：

```
@Override
protected void onSizeChanged(int w, int h, int oldw, int oldh) {
    super.onSizeChanged(w, h, oldw, oldh);
}
```

可以看出，它又四个参数，分别为 宽度，高度，上一次宽度，上一次高度。

这个函数比较简单，我们只需关注 宽度(w), 高度(h) 即可，这两个参数就是View最终的大小。

4.确定子View布局位置(onLayout)

确定布局的函数是onLayout，它用于确定子View的位置，在自定义ViewGroup中会用到，他调用的是子View的layout函数。

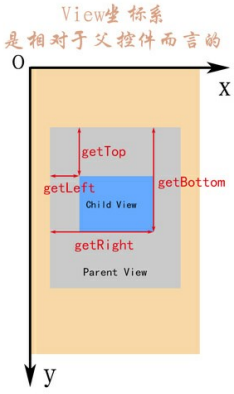
在自定义ViewGroup中，onLayout一般是循环取出子View，然后经过计算得出各个子View位置的坐标值，然后用以下函数设置子View位置。

```
child.layout(l, t, r, b);
```

四个参数分别为：

名称	说明	对应的函数
l	View左侧距父View左侧的距离	getLeft();
t	View顶部距父View顶部的距离	getTop();
r	View右侧距父View左侧的距离	getRight();
b	View底部距父View顶部的距离	getBottom();

具体可以参考 坐标系 这篇文章。



PS：关于onLayout这个函数在讲解自定义ViewGroup的时候会详细讲解。

5.绘制内容(onDraw)

onDraw是实际绘制的部分，也就是我们真正关心的部分，使用的是Canvas绘图。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
}
```

关于Canvas绘图是本章节的重点，会分几篇文章进行详细讲解，敬请期待OwO。

6.对外提供操作方法和监听回调

自定义完View之后，一般会对外暴露一些接口，用于控制View的状态等，或者监听View的变化.

本内容会在后续文章中以实例的方式进讲解。

三.重点知识梳理

自定义View分类

> PS ：实际上ViewGroup是View的一个子类。

类别	继承自	特点
View	View SurfaceView 等	不含子View
ViewGroup	ViewGroup xxLayout等	包含子View

自定义View流程：

步骤	关键字	作用
1	构造函数	View初始化
2	onMeasure	测量View大小
3	onSizeChanged	确定View大小
4	onLayout	确定子View布局(自定义View包含子View时有用)
5	onDraw	实际绘制内容
6	提供接口	控制View或监听View某些状态。

About

本系列相关文章

作者微博: GcsSloop

参考资料：

View
ViewGroup
View.MeasureSpec
onMeasure，MeasureSpec源码 流程 思路详解
Android中自定义样式与View的构造函数中的第三个参数defStyle的意义
android view构造函数研究
Android View构造方法第三参数使用方法详解
Android 自定义View onMeasure方法的实现
Android API指南(二)自定义控件02之 onMeasure
Android中View的绘制过程 onMeasure方法简述
Android自定义View基础-坐标系

最近的文章

安卓自定义View进阶-Canvas之绘制图形

在上一篇自定义View分类与流程中我们了解自定义View相关的基本知识，不过，这些东西依旧还是理论，并不能拿来(zhuang)用(B), 这一次我们就了解一些能(zhuang)用(B)的东西。在.....

2016-01-21 • CustomView

继续阅读

更早的文章

安卓自定义View基础-颜色

简要介绍安卓中的颜色相关内容，包括颜色的定义，创建颜色的几种方式，以及颜色的混合模式等。一.简单介绍颜色安卓支持的颜色模式： 颜色模式 备注 .....

2016-01-11 • CustomView

继续阅读