Android中的FlowLayout~

> 相信大家在Java的图形化界面中，经常使用到FlowLayout，flowLayout即流式布局，就是说控件会按排分布，当一行装不下的时候自动换到下一行。在安卓中没有这种布局，所以我们可以自己写一个这种布局~

这里面便是我们的流式布局了，下面我们可以一起看一下代码：

```java
/**
 * Created by linSir on 16/7/30.流式布局
 */
public class FlowLayout extends ViewGroup {
    private float mVerticalSpacing; //每个item纵向间距
    private float mHorizontalSpacing; //每个item横向间距

    public FlowLayout(Context context) {
        super(context);
    }
    public FlowLayout(Context context, AttributeSet attrs) {
        super(context, attrs);
    }
    public void setHorizontalSpacing(float pixelSize) {
        mHorizontalSpacing = pixelSize;
    }
    public void setVerticalSpacing(float pixelSize) {
        mVerticalSpacing = pixelSize;
    }
    @Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        int selfWidth = resolveSize(0, widthMeasureSpec);

        int paddingLeft = getPaddingLeft();
        int paddingTop = getPaddingTop();
        int paddingRight = getPaddingRight();
        int paddingBottom = getPaddingBottom();

        int childLeft = paddingLeft;
        int childTop = paddingTop;
        int lineHeight = 0;

        //通过计算每一个子控件的高度，得到自己的高度
        for (int i = 0, childCount = getChildCount(); i < childCount; ++i) {
            View childView = getChildAt(i);
            LayoutParams childLayoutParams = childView.getLayoutParams();
            childView.measure(
                    getChildMeasureSpec(widthMeasureSpec, paddingLeft + paddingRight,
                            childLayoutParams.width),
                    getChildMeasureSpec(heightMeasureSpec, paddingTop + paddingBottom,
                            childLayoutParams.height));
            int childWidth = childView.getMeasuredWidth();
            int childHeight = childView.getMeasuredHeight();

            lineHeight = Math.max(childHeight, lineHeight);

            if (childLeft + childWidth + paddingRight > selfWidth) {
                childLeft = paddingLeft;
                childTop += mVerticalSpacing + lineHeight;
                lineHeight = childHeight;
            } else {
                childLeft += childWidth + mHorizontalSpacing;
            }
        }

        int wantedHeight = childTop + lineHeight + paddingBottom;
        setMeasuredDimension(selfWidth, resolveSize(wantedHeight, heightMeasureSpec));
    }

    @Override
    protected void onLayout(boolean changed, int l, int t, int r, int b) {
        int myWidth = r - l;

        int paddingLeft = getPaddingLeft();
        int paddingTop = getPaddingTop();
        int paddingRight = getPaddingRight();

        int childLeft = paddingLeft;
        int childTop = paddingTop;

        int lineHeight = 0;

        //根据子控件的宽高，计算子控件应该出现的位置。
        for (int i = 0, childCount = getChildCount(); i < childCount; ++i) {
            View childView = getChildAt(i);

            if (childView.getVisibility() == View.GONE) {
                continue;
            }

            int childWidth = childView.getMeasuredWidth();
```

```
            int childHeight = childView.getMeasuredHeight();

            lineHeight = Math.max(childHeight, lineHeight);

            if (childLeft + childWidth + paddingRight > myWidth) {
                childLeft = paddingLeft;
                childTop += mVerticalSpacing + lineHeight;
                lineHeight = childHeight;
            }
            childView.layout(childLeft, childTop, childLeft + childWidth, childTop + childHeight);
            childLeft += childWidth + mHorizontalSpacing;
        }
    }
}
```

到这里，我们便已经创建好了流式布局，接下来我们可以在我们的代码中使用它，下面我展示一下如何使用它。

```
 <com.example.lin_sir_one.tripbuyer.customview.FlowLayout
                    android:id="@+id/customView"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginLeft="75dp"
                    android:layout_marginRight="40dp"
                    android:orientation="horizontal"
                    >
```

```
/**
 * Created by linSir on 16/7/30.买手行程详情界面
 */
public class AddressDetailsActivity extends AppCompatActivity {

    @BindView(R.id.rel_address_details) RelativeLayout rl;
    @BindView(R.id.customView) FlowLayout mFlowLayout;
    private String mNames[] = {
            "美容护肤", "美容护肤", "美容护肤",
            "美容护肤", "美容护肤", "美容护肤",
            "美容护肤", "美容护肤", "美容护肤",
            "美容护肤", "美容护肤", "美容护肤",
    };

    @Override protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_address_details);
        ButterKnife.bind(this);

        ViewGroup.MarginLayoutParams lp = new ViewGroup.MarginLayoutParams(
                LayoutParams.WRAP_CONTENT, LayoutParams.WRAP_CONTENT);
        lp.setMargins(5, 5, 5, 5);


        for (int i = 0; i < mNames.length; i++) {
            TextView view = new TextView(this);
            view.setText(mNames[i]);
            view.setTextSize(12);
            view.setBackgroundDrawable(getResources().getDrawable(R.drawable.text_bg));
            mFlowLayout.addView(view, lp);
            mFlowLayout.setHorizontalSpacing(10);
            mFlowLayout.setVerticalSpacing(10);
//            if (i >= 3) {
//                view.setId(R.id.release_price);
//                view.setVisibility(View.GONE);
//            }

        }

    }

//    @OnClick(R.id.down)
//    public void doew() {
//        TextView view = (TextView) findViewById(R.id.release_price);
//        view.setVisibility(View.VISIBLE);
//    }

}
```

我们在我们的activity中，可以很简单的使用它，使用的截图我在一开始有给出来了，我们只需要简单的设置layout_margin，也可以设置一下，两个textview左右的距离，和上下的距离，这样我们就已经设置好了，就已经完事了。

> onMeasure 我们在这个方法里，需要加以判断，如果控件放在一行中可以放下，我们就放在一起，并且测量控件的长和宽，如果在里面装不下，便会自动换行，而且也会重新记录它的长和宽。

> onLayout 在这个方法里面，我们做的事情是，计算子控件出现的位置，它具有5个参数，第一个是通知我们的控件是否发生了改变，还有四个参数描述了我们的控件的位置。我们会根据控件的显隐状态来判断这个控件是否要来加载，还要根据传过来的参数，来绘制这个控件。

好了，以上便是，我们通过自定义来实现的流式布局啦，大家可以把它应用在安卓中了~