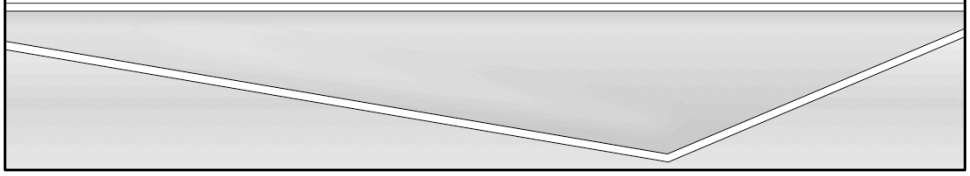
Please check for updated slides at:
https://github.com/desertjim/CustomViewGroups

LEVERAGING CUSTOM VIEWGROUPS

By James Baca

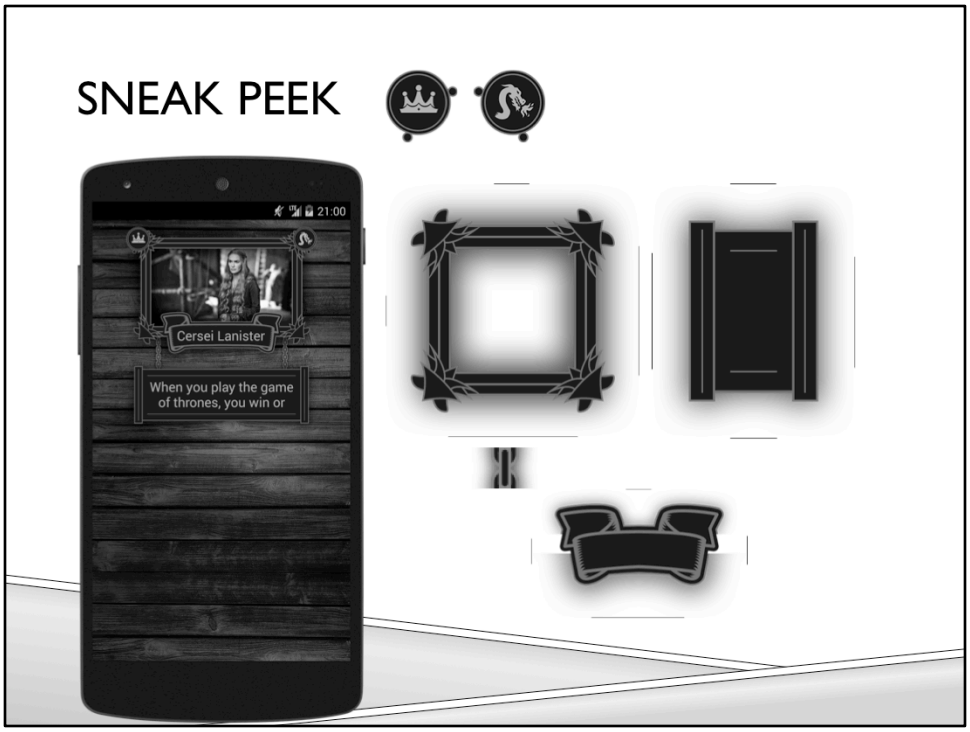
Email List: http://jamesbaca.net

Twitter: desertjim



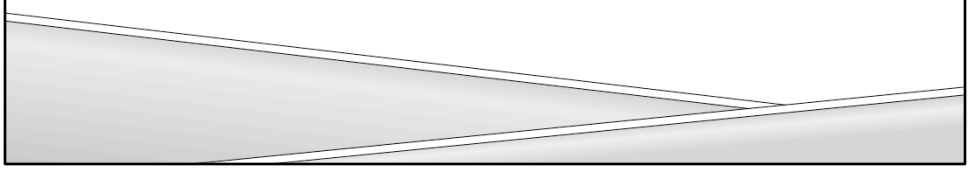
SNEAK PEEK


21:00
Cersei Lanister
When you play the game of thrones, you win or


SNEAK PEEK
21:00
Cersei Lanister
When you play the game
of thrones, you win or

TASK #1

- Standup and introduce yourself to three people
  1. Find out their name, if they watch the show, and which house/character they like



WHAT IS A VIEWGROUP?

- AKA Layout, Container
- LinearLayout
- RelativeLayout
- GridView
- ViewPager
- Absolute(worst)Layout *shudder*
- Anything that extends the ViewGroup

WHY CUSTOM VIEWGROUPS?

- Jank
- Maintenance
- Crazy Mad Designer ;)
- Flexible

STANDARD LAYOUT FAIL

TASK #2

- Download the project from github
- https://github.com/desertjim/CustomViewGroups
- Download two images to use for your portrait from the internet(one that is portrait, and one that is landscape)

RECOMMENDATIONS

- Android Studio
- Hierarchy Viewer

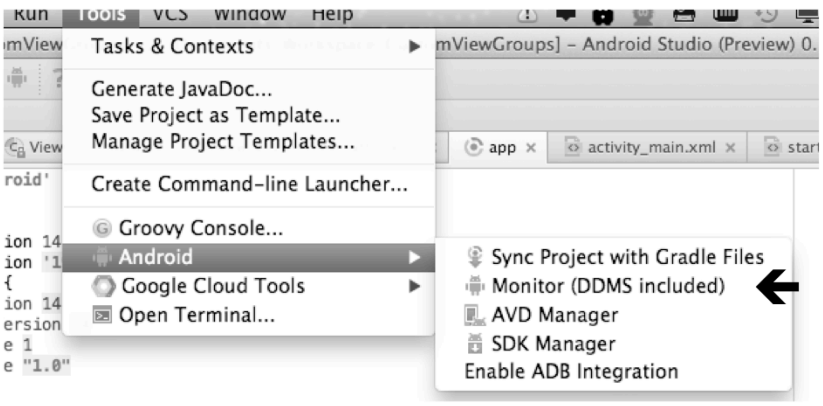
TASK #3

- Install the initial.apk(check the releases on github.com)

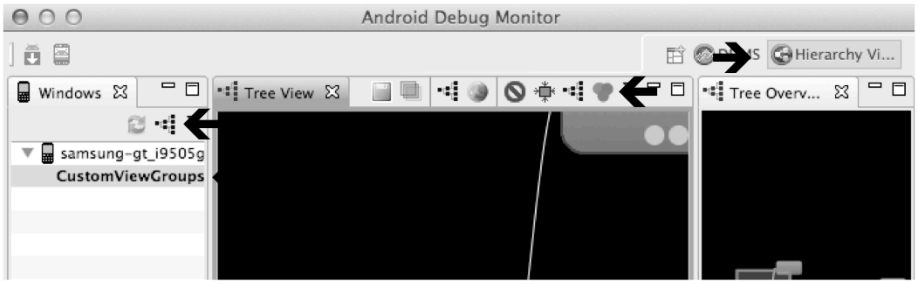
HIERARCHY VIEWER

- Where's my child?!
- Performance Analysis
- Developer Phones Just Work*
- Others: View Server

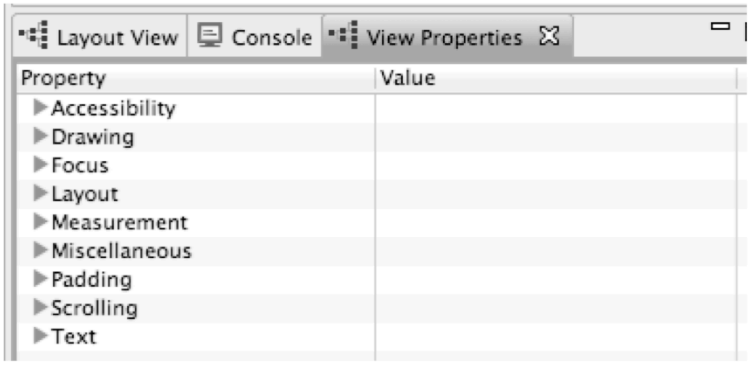
LOAD THE HIERARCHY VIEWER



LOAD THE HIERARCHY VIEWER



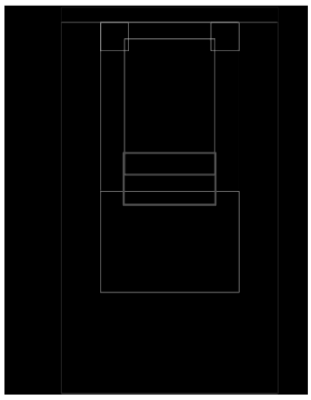
HIERARCHY VIEWER: VIEW PROPERTIES



TASK #4

- Using Hierarchy Viewer find the following values for the name scroll:
  - Width
  - Heigth
  - measuredWidth
  - measuredHeight
- Note what property category they were found under

HIERARCHY VIEWER: LAYOUT VIEW

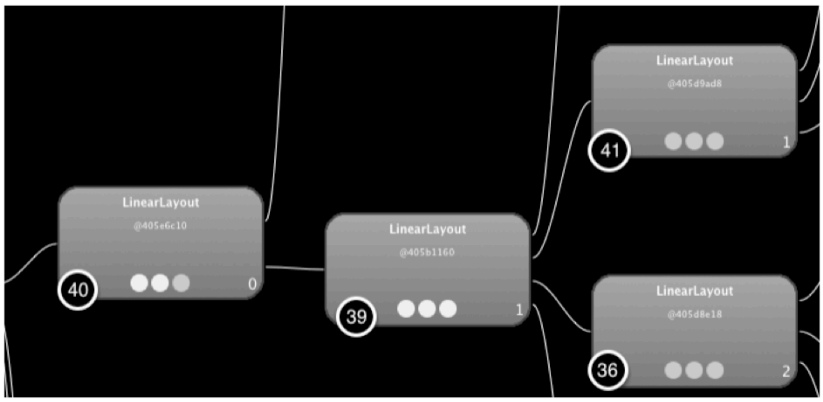


WAY WAY SIMPLIFIED OVERVIEW

- Header Comments in View.java:

Category	Methods
Creation	Constructors
Layout	onMeasure(int mSpecW, int mSpecH)
Layout	onLayout(int l, int t,int r, int b)
Drawing	dispatchDraw(Canvas)

HIERARCHY VIEWER: TREE VIEW



USING A CUSTOM VIEWGROUP IN XML

- Namespace must match
- E.g. given class com.example.ProfileLayout
  - <com.example.ProfileLayout …
- Use xml namespace to use your custom attributes
  - xmlns:profile="http://schemas.android.com/apk/res-auto"

TASK #5

- In activity_main.xml add standard views(ImageView, TextView etc) for the following items. Use wrap_content for layout_width and layout_height
  - Crown image
  - Dragon image
  - Portrait image
  - Frame image
  - Chain 1view
  - Chain 2 view
  - Caption Box text
  - Name Scroll text

CONSTRUCTORS

- View(Context)
- View(Context, AttributeSet)
- View(Context, AttributeSet, int)

ATTRIBUTE SET?!

- Collection of xml attributes and xml values
- Used for setting custom (layout) parameters

CUSTOM XML ATTRIBUTES

- Reusable
- Measure Directions
- Layout Directions
- Heck Even Draw Directions

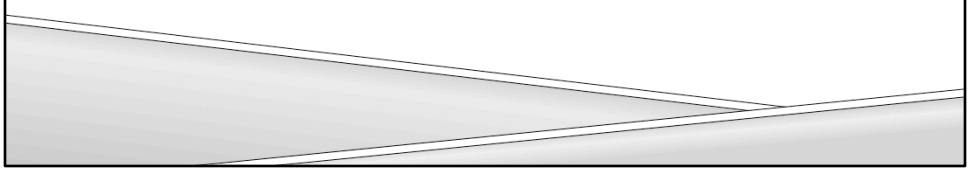
CREATING CUSTOM XML ATTRIBUTES

- res/values/attrs.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="MySuperAwesomeLayout_LayoutParams">
        <attr name="layout_badgeCorner" format="Integer"/>
        <attr name="layout_alignTop" format="boolean"/>
        <attr name="hiddenText" format="string"/>
    </declare-styleable>
</resources>
```

TASK #6

- Create a custom attribute for identifying if a view is the portrait

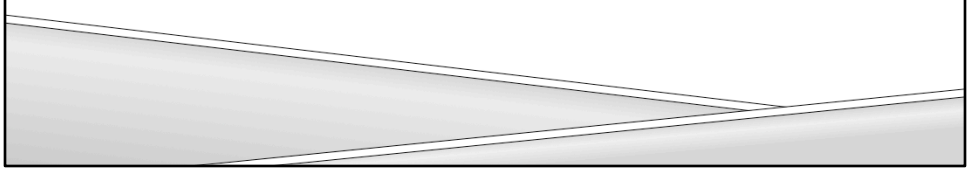


WHAT ARE LAYOUTPARAMS

- Directions
  - Size
  - Placement
- Common examples
  - Layout_width
  - Layout_height
  - In relative layout:
    - layout_centerHorizontal
    - layout_below

WHY CUSTOM LAYOUTPARAMS

- Reusable
- Simplification

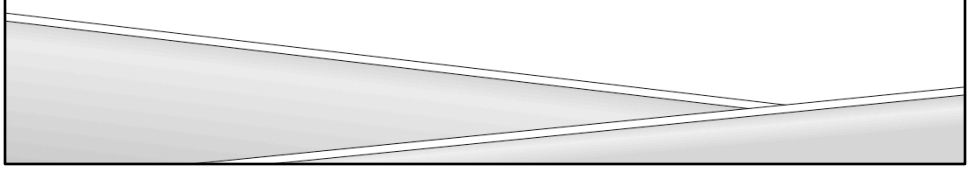


EXAMPLES OF LAYOUTPARAMS

- LinearLayout.LayoutParams
- ViewGroup.MarginLayoutParams
- RelativeLayout.LayoutParams

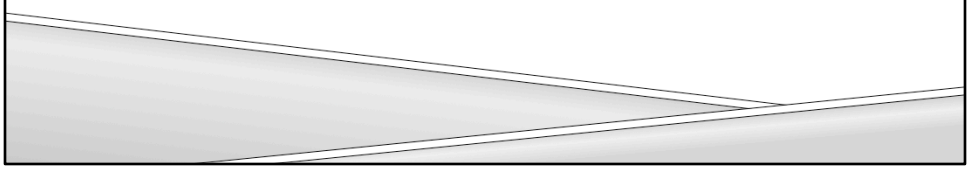
USING CUSTOM LAYOUTPARAMS

- Extend a LayoutParams class
- Instantiate LayoutParams
- Do Something with LayoutParams



TASK #7

- Uncomment layout params section at bottom of ProfileLayout
- Create default constructor LayoutParams(Context c, AttributeSet attrs)



READING ATTRIBUTES

```
public static class MyAwesomeParams extends ViewGroup.MarginLayoutParams {
        int corner = 0;

        public MyAwesomeParams (Context context, AttributeSet attrs) {
            super(context, attrs);
            TypedArray a = context.obtainStyledAttributes(
                    attrs,
                    R.styleable.MyAwesome);
            corner = a.getInt(R.styleable.MyAwesome_layout_badge_corner, 0);
            …
            a.recycle();
        }
}
```

TASK #8

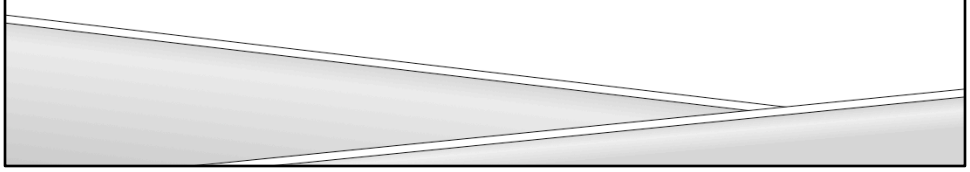
- Read attribute for the portrait, store it in a public member variable

GENERATE CUSTOM LAYOUTPARAMS

```
public ViewGroup.LayoutParams generateLayoutParams(AttributeSet attrs) {
        return new MyAwesomeLayout.LayoutParams(getContext(), attrs);
 }

boolean checkLayoutParams(ViewGroup.LayoutParams)

ViewGroup.LayoutParams generateLayoutParams(ViewGroup.LayoutParams p)

ViewGroup.LayoutParams generateDefaultLayoutParams() {
        return new LayoutParams();
}
```

USING CUSTOM LAYOUTPARAMS

- OnMeasure
- OnLayout
- DispatchDraw
- Other places

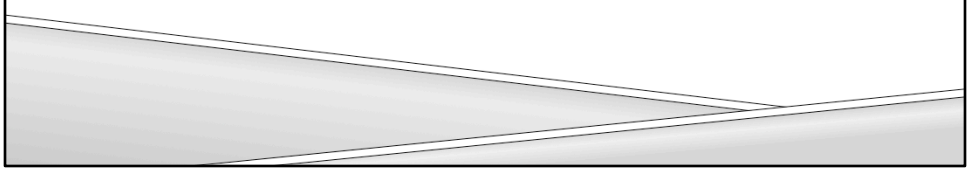


ON MEASURE

- Responsibilities(children, and self)
- Must call setMeasuredDimension(int rawPixelWidth, int rawPixelHeight)
- Takes in measure specs(*2x)
- MeasureSpec modes:
  1. Unspecified
  2. AT Most
  3. Exactly
- MeasureSpec size in (raw)pixels

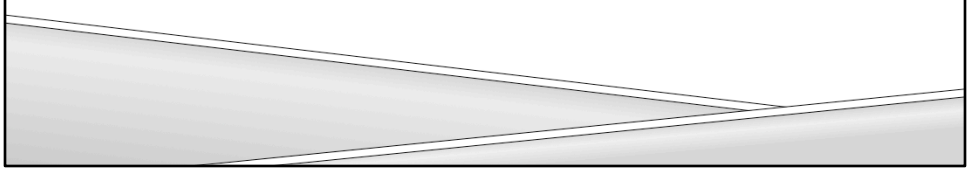
RAW PIXELS

- Actual pixels on a given device
- Convert dp to px



TASK #9

- Change on measure to a completely empty function
- Run the app, what happens?



MEASURE

- Final method
- setMeasuredDimension(int, int)!!!
- View.java: http://goo.gl/wfHQLi

SETMEASUREDDIMENSION

```
protected final void ...setMeasuredDimension(int measuredWidth, int
measuredHeight) {

    mMeasuredWidth = measuredWidth;
    mMeasuredHeight = measuredHeight;
   mPrivateFlags |= MEASURED_DIMENSION_SET;
}
```

MEASURING CHILDREN

- getChildCount()
- getChildAt(int)
- child.getLayoutParams()
- measureChild*(…) or child.measure(int, int)
- Multipass but don't go crazy
- getMeasuredWidth, getMeasuredHeight

TASK #10

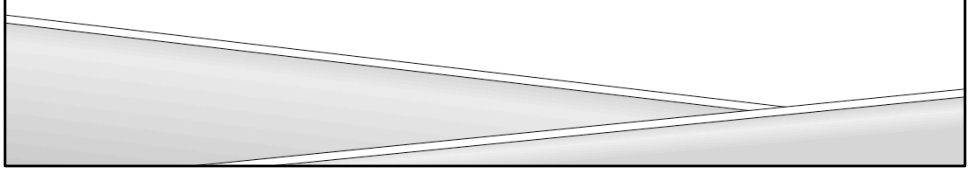
- Override onMeasure to measure all of the children
- Use measureChild

MEASURE CHILD HELPER FUNCTION

```
protected void measureChild(View child, int parentWidthMeasureSpec,
    int parentHeightMeasureSpec) {

    final LayoutParams lp = child.getLayoutParams();

    final int childWidthMeasureSpec = getChildMeasureSpec(parentWidthMeasureSpec,
                        mPaddingLeft + mPaddingRight, lp.width);

    final int childHeightMeasureSpec = getChildMeasureSpec(parentHeightMeasureSpec,
                        mPaddingTop + mPaddingBottom, lp.height);

    child.measure(childWidthMeasureSpec, childHeightMeasureSpec);
}
```

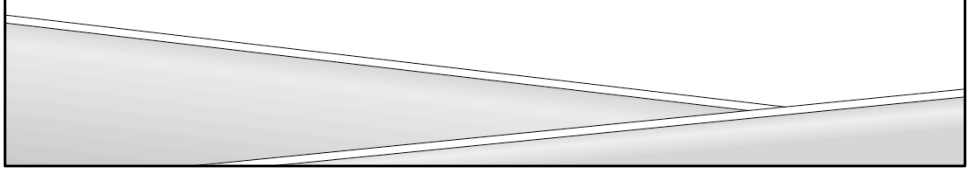
GETCHILDMEASURESPEC

- Hard worker function
- Not always what you want



ROLL YOUR OWN MEASURESPEC

- Int = MeasureSpec.getMode(int)
- Int = MeasureSpec.getSize(int)
- Int = MeasureSpec.makeMeasureSpec(int, int)



ON LAYOUT

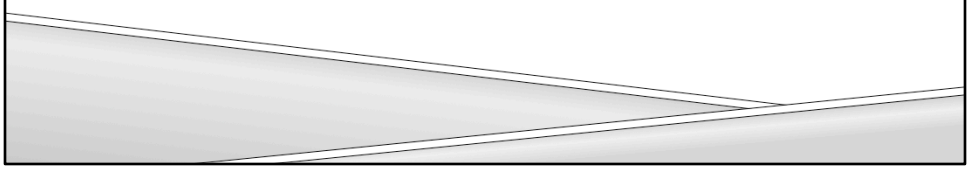
- Where to place each child
- getChildCount()
- getChildAt(int index)
- child.getLayoutParams()
- child.layout(left int, top int, right int, bottom int)
- child.getWidth(), child.getHeight()

GETMEASUREDWIDTH VS. GETWIDTH

- onMeasure
- onLayout

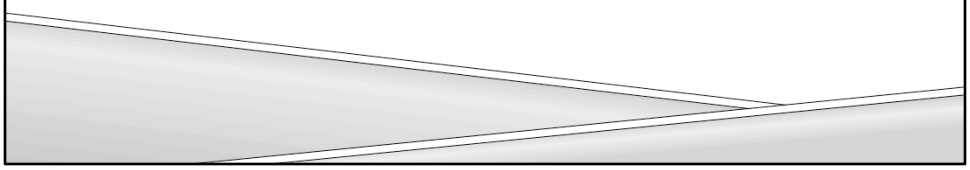
TASK #11

- Override onLayout and place all of the children vertically



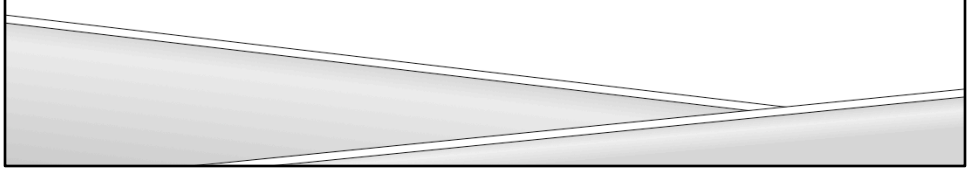
DISPATCH DRAW

- Draw each child



CUSTOM VIEWGROUP STRATEGERY

- Group by proximity
- Order by display



TASK #12

- If the portrait attribute is present on the view center the view horizontally.

TASK #13

- Create an attribute for the frame
- If the frame attribute is present display it on top of the portrait

TASK #14

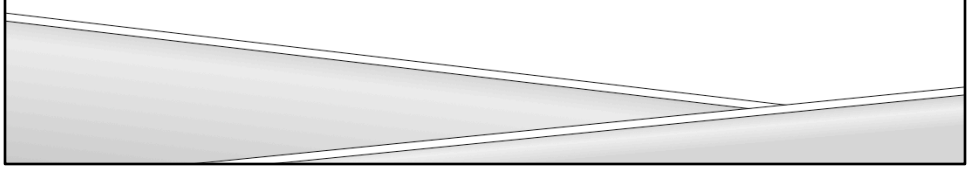
- Add custom attributes that define the amount of spacing outside the frames inner bounds area.
- Use these custom attributes to align the portrait and the frame so that:
  1. The frame is the perfect size for the portrait
  2. The portrait is displayed in the portrait
- Use measurechild for the frame
  - Why doesn't measure child work? Hint use the debugger to step into ViewGroup.java

TASK #15

- Create an attribute for identifying the placement of the items that go on top/around of the frame
- Hint look at the android attrs.xml for some ideas how to accomplish this
- Use this attribute to display the crown in the top left of the frame

TASK #16

- Display one of the symbols on the top right

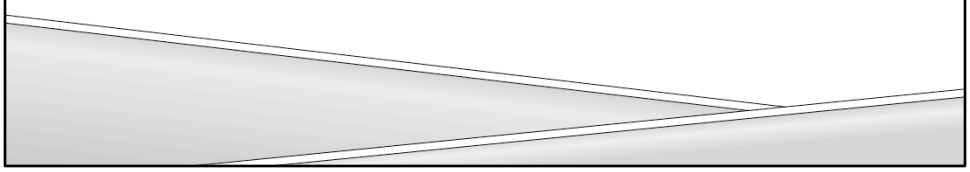


TASK #17

- Display the name scroll using the same custom attribute

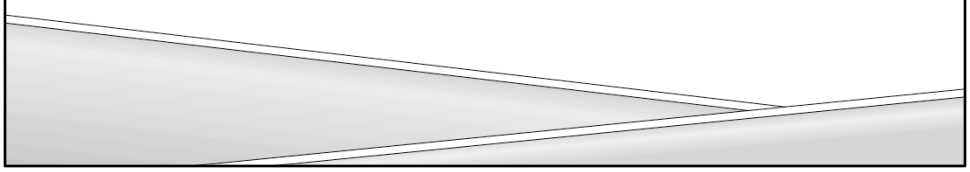
TASK #18

- Display the quote TextView under the picture
- Make the TextView sides match the left and right sides
- Make the TextView appropriately sized(hint MeasureSpec.UNSPECIFIED)



TASK #19

- Display the chains between the picture of the quote TextView
- Make sure that the frame is on top of the chains
- Make sure that the quote box is on top of the chains



TASK #20

- Swap only portrait source image for a landscape one
  - Make sure that it still displays correctly

FINAL THOUGHTS

- Look at the source code for built in layouts
- Hierarchy Viewer is invaluable
- Reminder about my developers email list
- Office hours

- Thank you!