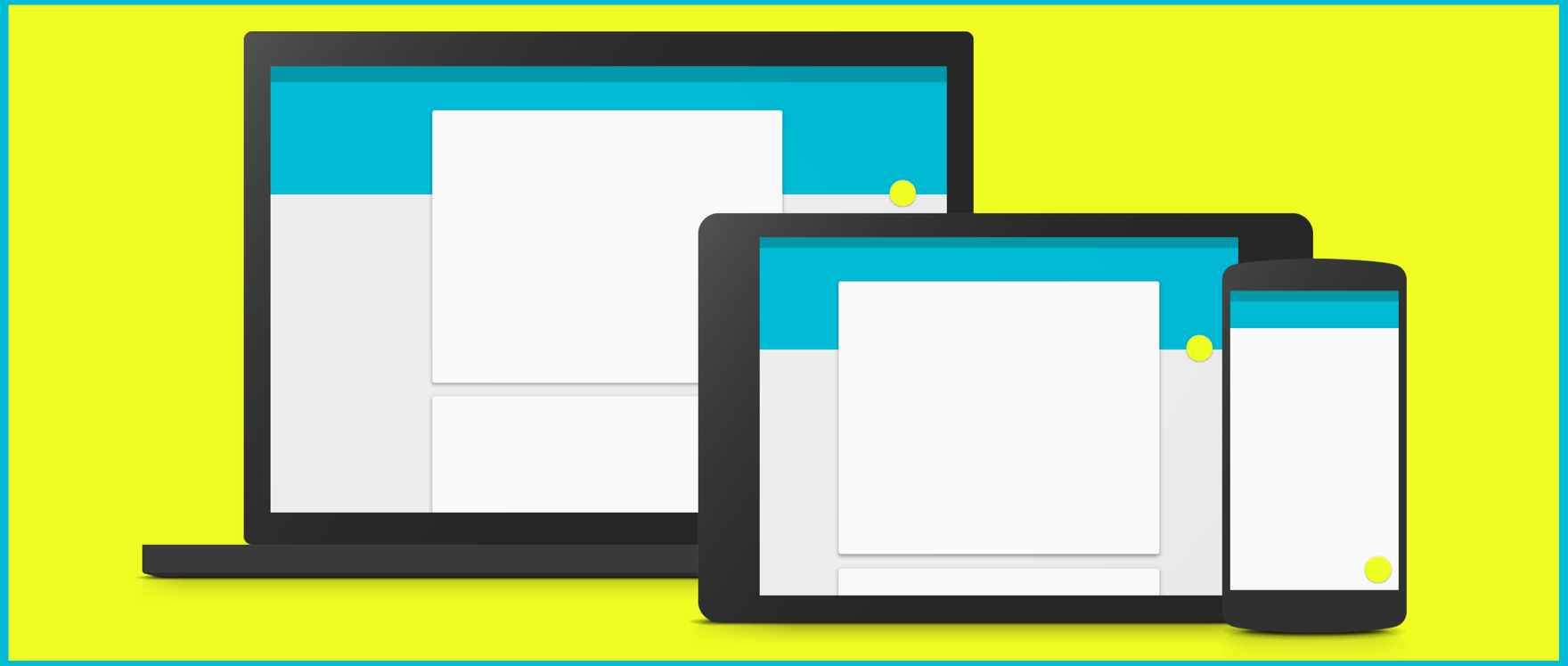


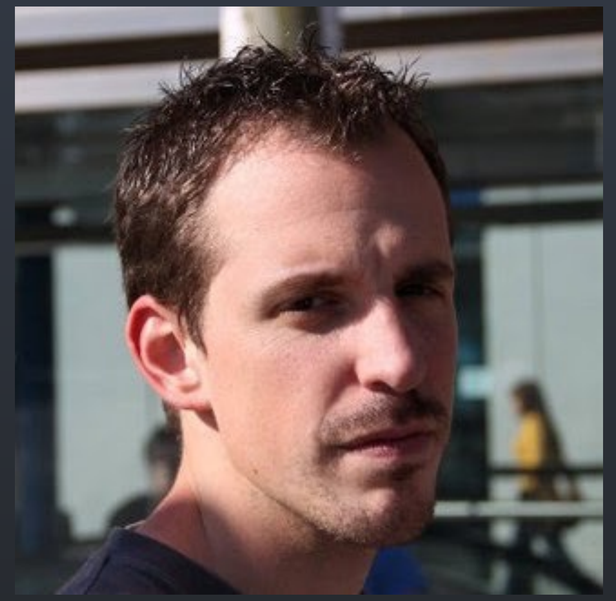
Materialize your App

Put Material Design into practice

ANTONIO LEIVA

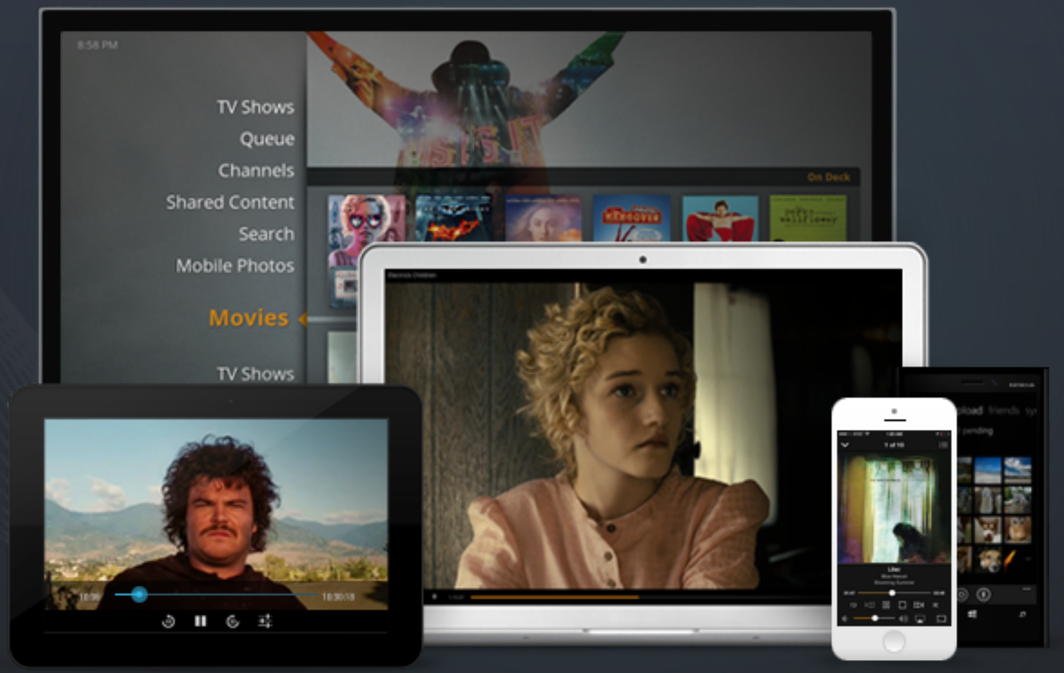
Antonio Leiva

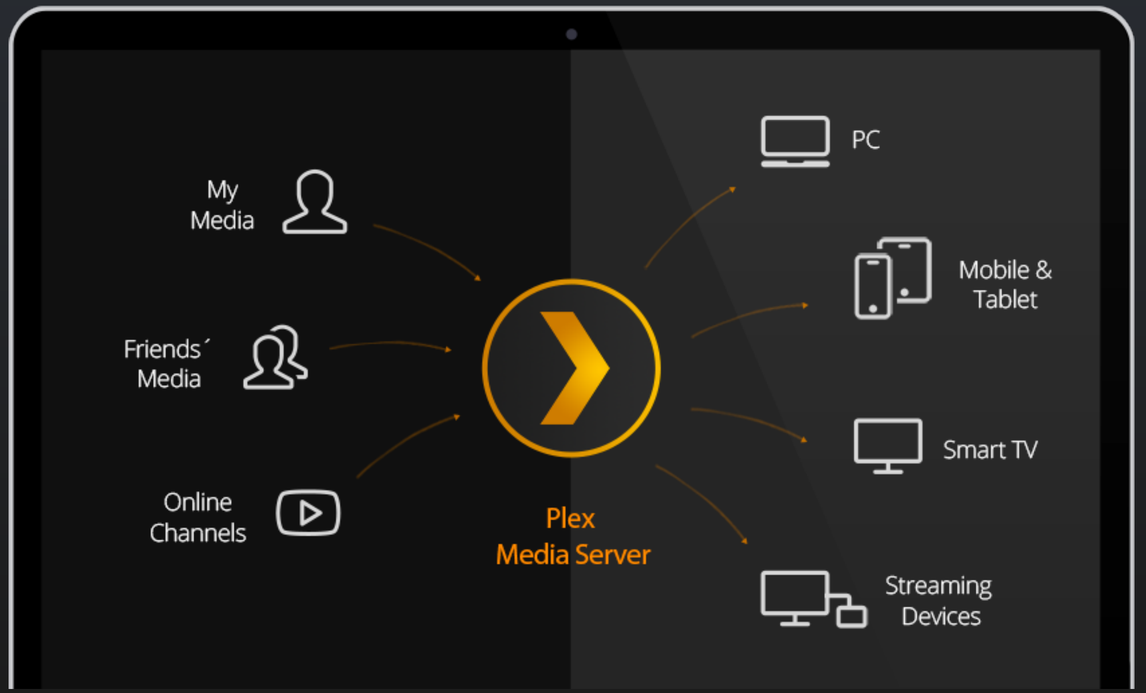
Android Engineer @ Plex, Inc



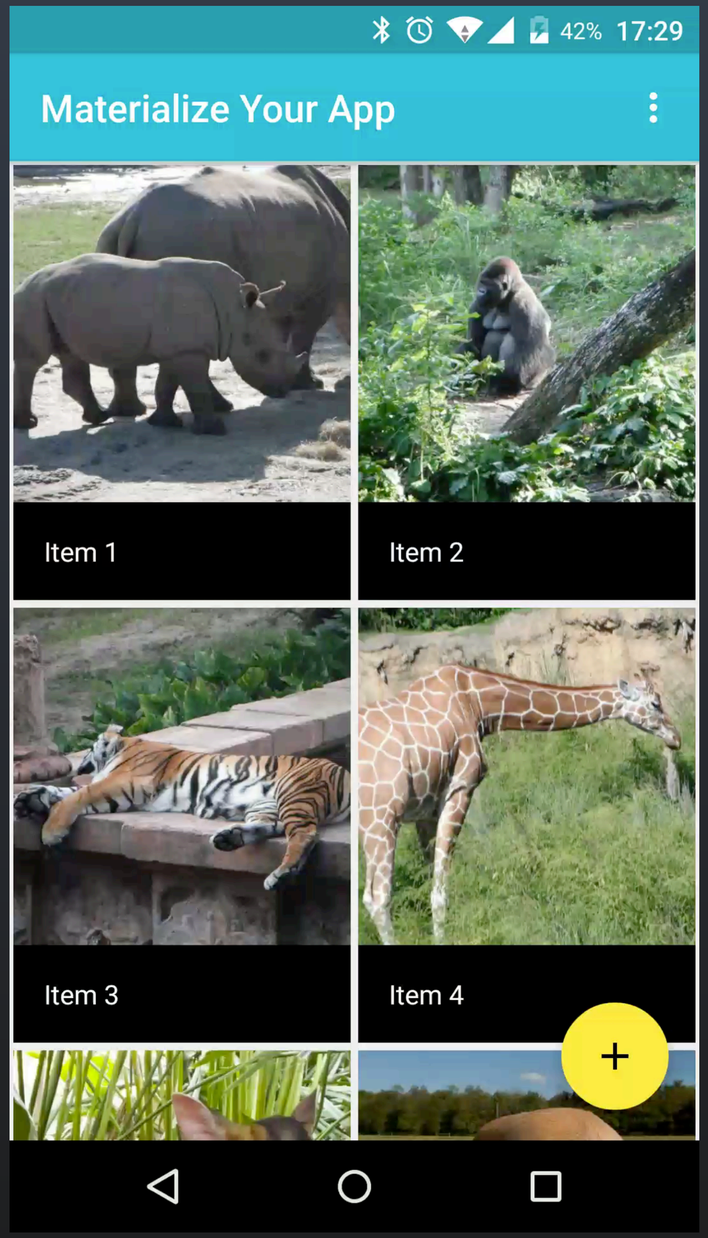
http://antonioleiva.com
http://plus.google.com/+AntonioLeivaGordillo
@lime_cl

PLEX





http://plex.tv



Materialize Your App

http://goo.gl/dXVujh

Source:

http://github.com/antoniolg/MaterializeYourApp

https://play.google.com/store/apps/details?id=com.antonioleiva.materializeyourapp

Index

1. What is Material?
2. Material Everywhere: AppCompat
3. Toolbar
4. Activity transitions
5. Ripples
6. RecyclerView
7. CardView
8. Palette
9. Tips

What is Material?

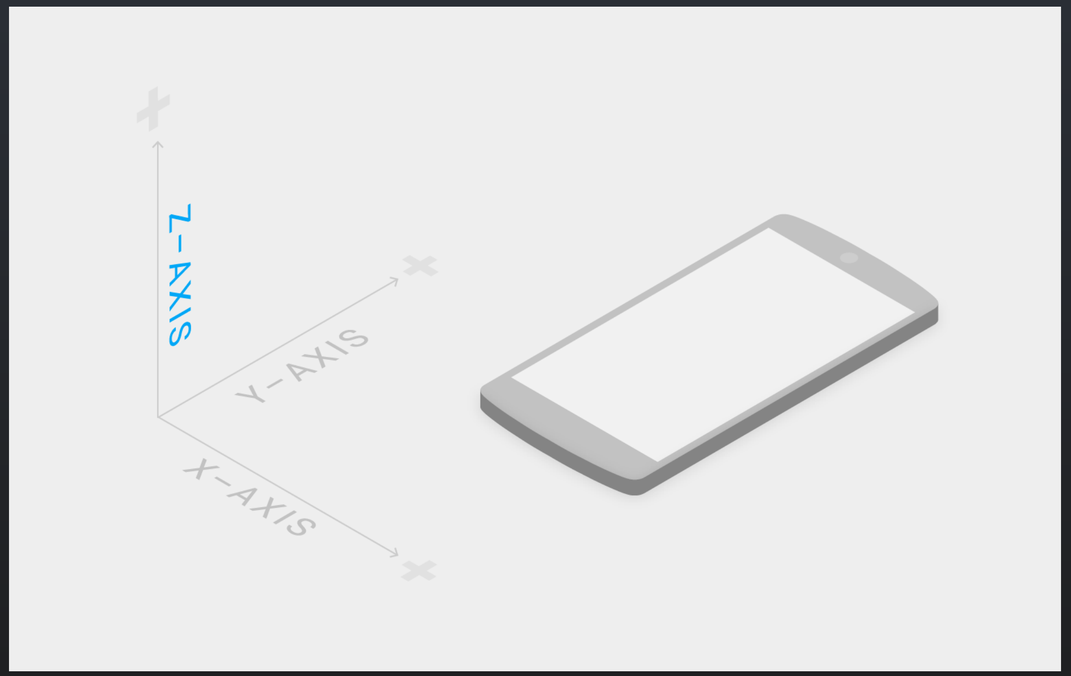
Material is the metaphor

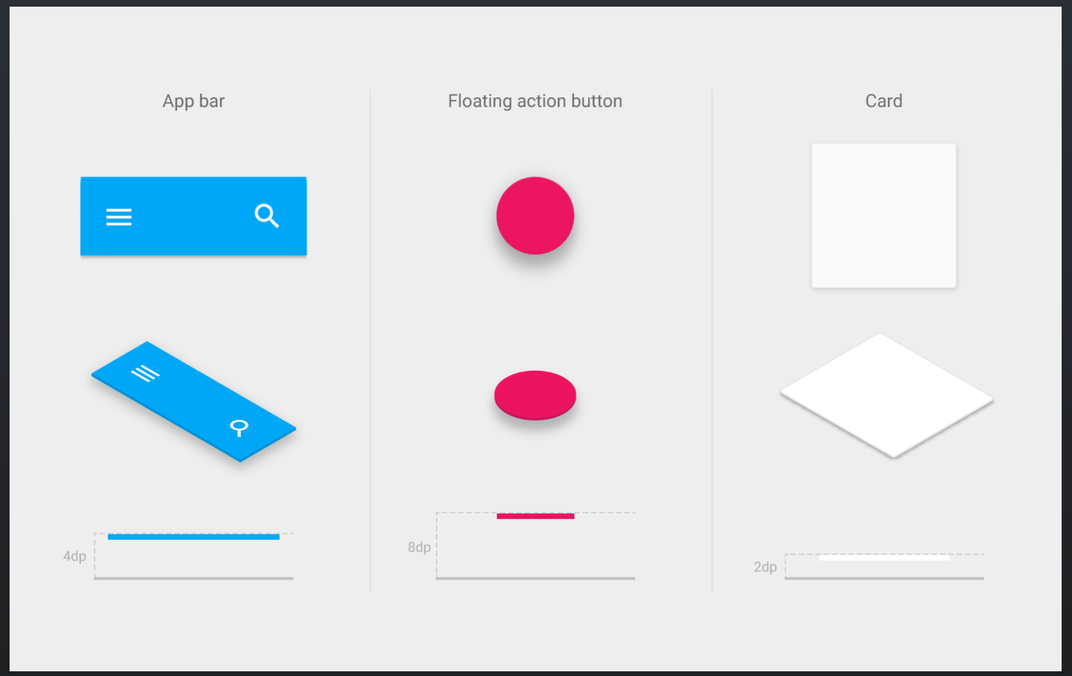
Bold, graphic, intentional

Motion provides meaning

What is Material?

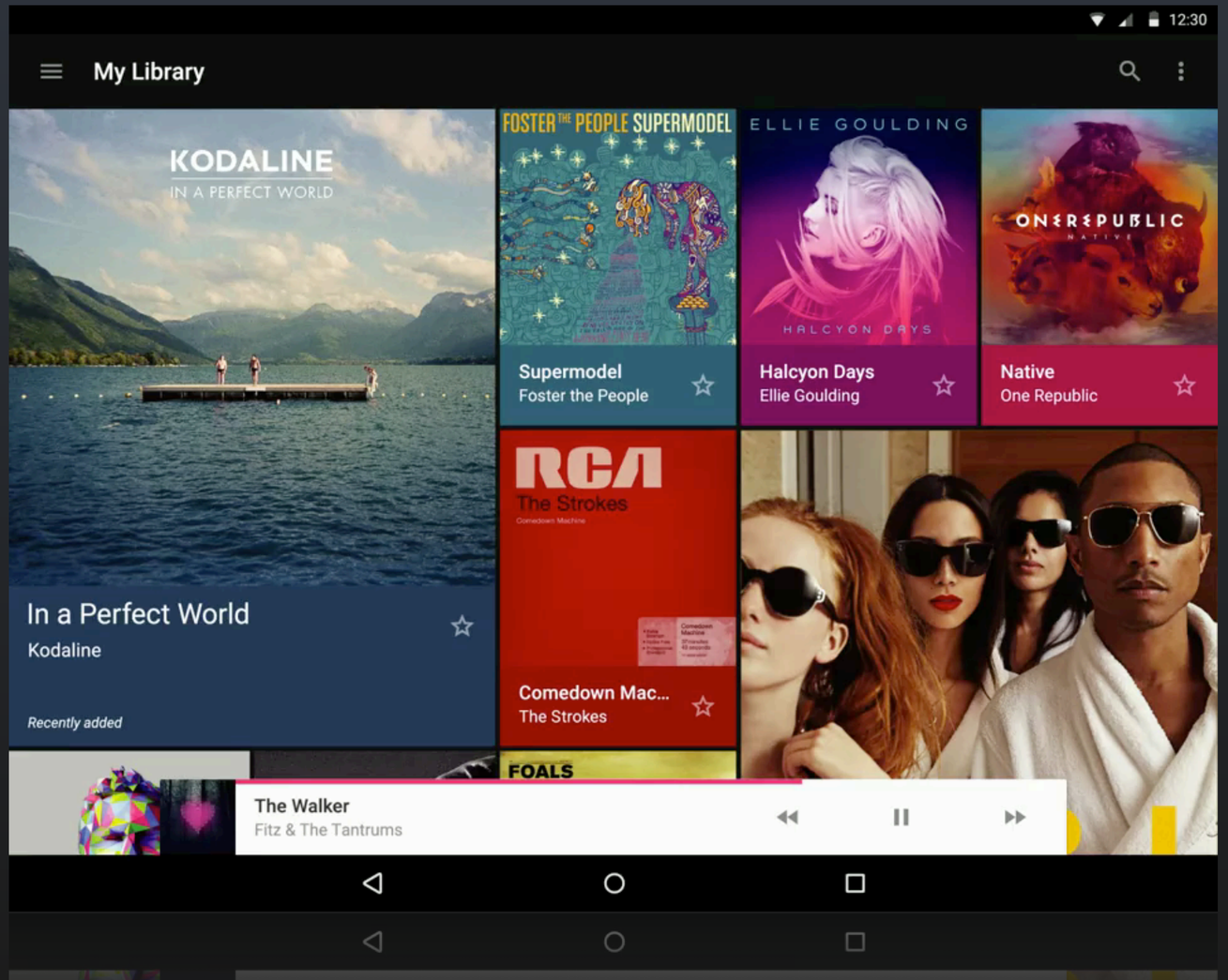
Z - Axis





What is material?

Meaningful transitions



Material Everywhere: AppCompat

1. Add the dependency

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:21.0.3'
}
```

Material Everywhere: AppCompat

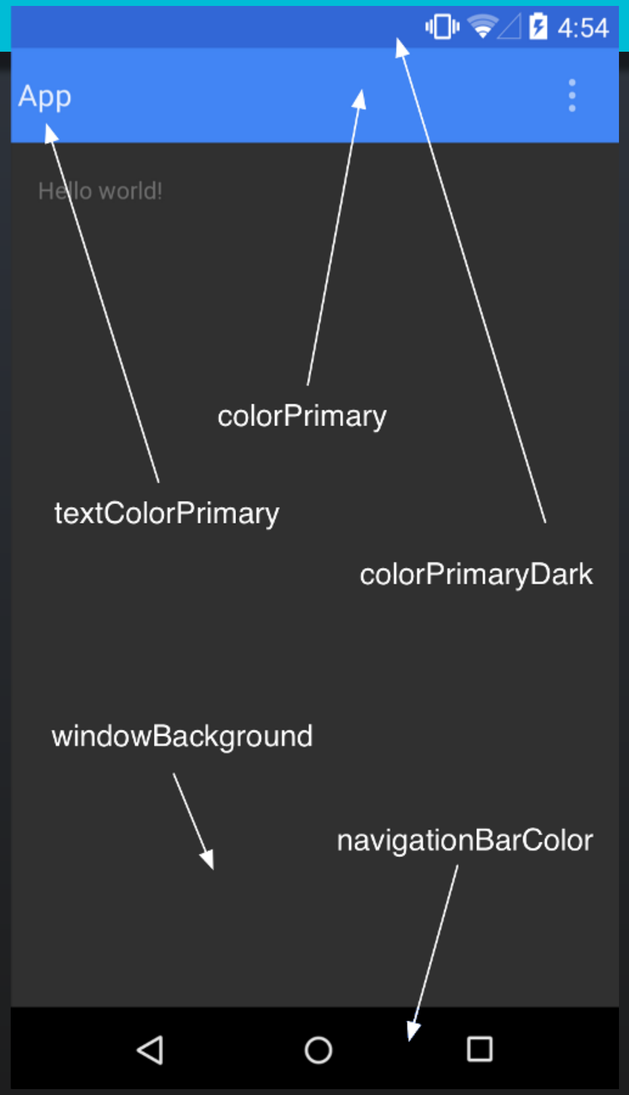
2. Make you theme extend AppCompat

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <item name="colorPrimary">@color/primary</item>
    <item name="colorPrimaryDark">@color/primary_dark</item>
    <item name="colorControlHighlight">@color/primary</item>
    <item name="colorAccent">@color/accent</item>
</style>
```

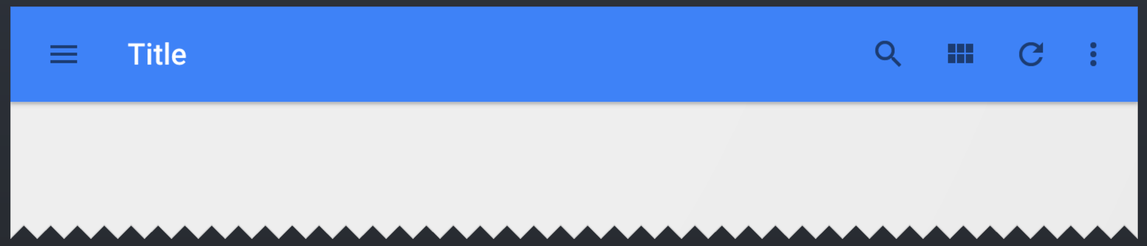
Material Everywhere: AppCompat

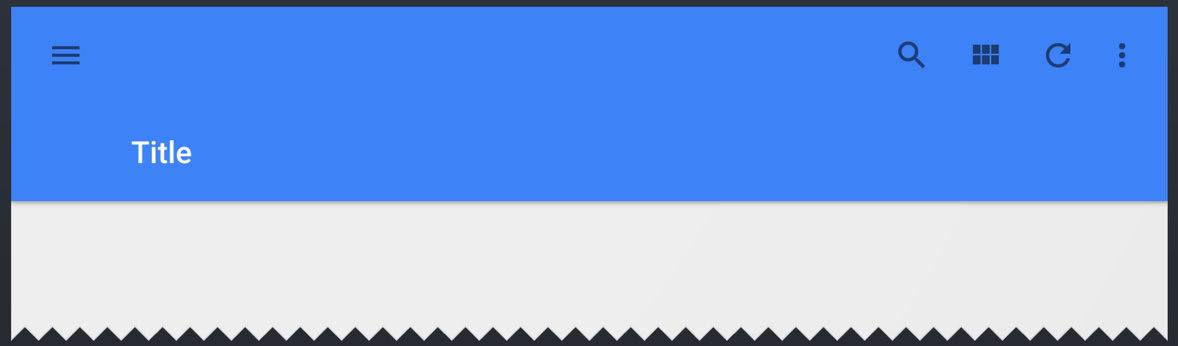
3. Set your primary colors

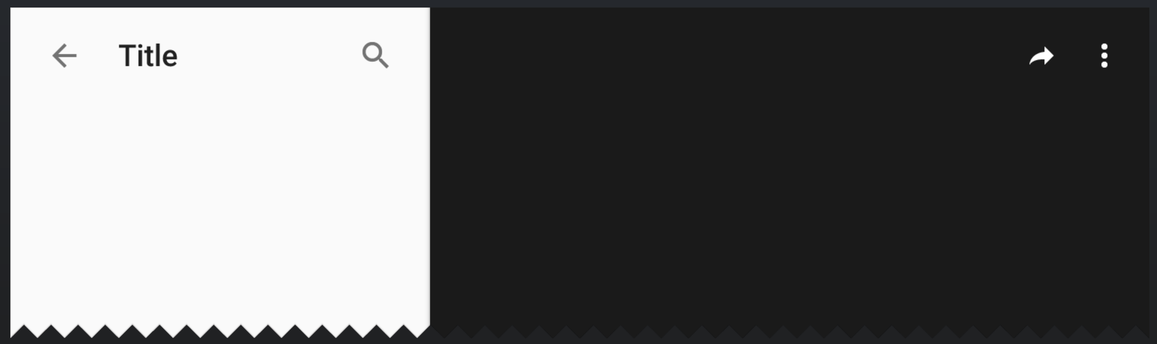
```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <item name="colorPrimary">@color/primary</item>
    <item name="colorPrimaryDark">@color/primary_dark</item>
    <item name=“colorControlHighlight">@color/accent_translucent</item>
    <item name="colorAccent">@color/accent</item>
</style>
```

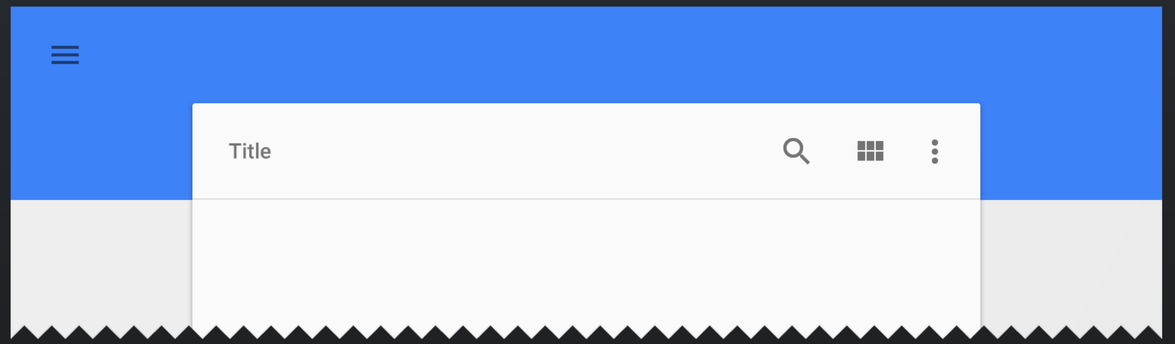


Toolbar









http://www.google.com/design/spec/layout/structure.html#structure-toolbars

Toolbar

1. Add a toolbar to you layouts

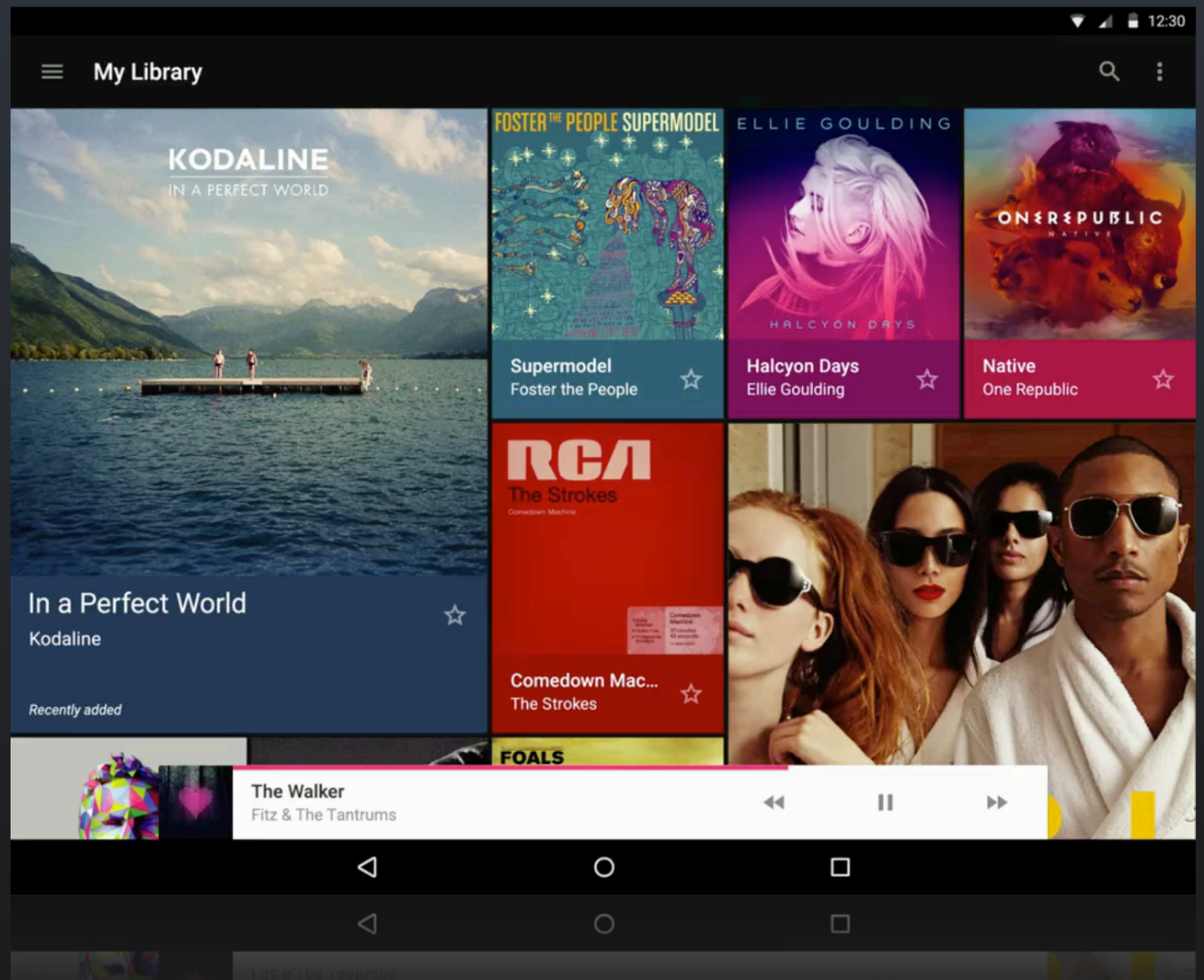
```
<android.support.v7.widget.Toolbar
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/toolbar"
    android:layout_height="?attr/actionBarSize"
    android:layout_width="match_parent"
    android:background="?attr/colorPrimary"
    android:elevation="4dp"
    app:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    app:popupTheme="@style/ThemeOverlay.AppCompat.Light" />
```

Toolbar

2. Assign it to work as the Activity ActionBar

```
@Override protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_home);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    if (toolbar != null) {
        setSupportActionBar(toolbar);
    }
}
```

Activity transitions



Activity transitions

1. Specify the transitions on theme declaration

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    ...
    <item name="android:windowEnterTransition">@android:transition/explode</item>
    <item name="android:windowReturnTransition">@android:transition/explode</item>
    <item name="android:windowSharedElementEnterTransition">@android:transition/move</item>
    <item name="android:windowSharedElementReturnTransition">@android:transition/move</item>
</style>
```

Activity transitions

2. Declare transitions programmatically

```
if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
    Slide transition = new Slide();
    getWindow().setEnterTransition(transition);
    getWindow().setReturnTransition(transition);
}
```

Activity transitions

3. For Shared elements, specify during navigation

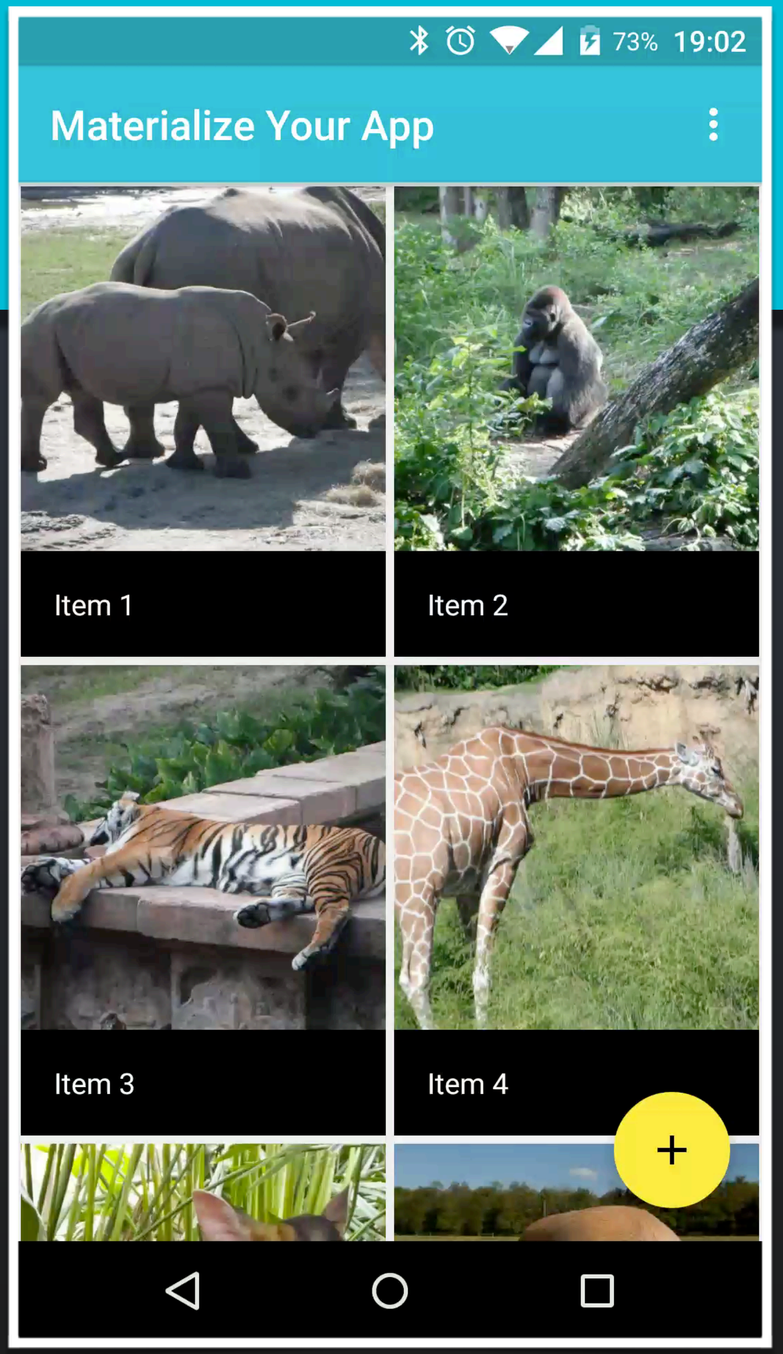
```
View transitionImage = view.findViewById(R.id.image);
ActivityOptionsCompat options =
ActivityOptionsCompat.makeSceneTransitionAnimation(activity, transitionImage,
"transitionImage");
ActivityCompat.startActivity(activity, intent, options.toBundle());
```

```
ViewCompat.setTransitionName(image, "transitionImage");
```

Ripples

res/drawable-v21

```
<ripple
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:color="?attr/colorControlHighlight">
    <item android:id="@android:id/mask"
          android:drawable="@color/accent"/>
</ripple>
```



Ripples

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    ...
    <item name="selectableItemBackground">@drawable/selectable_item_background</item>
    <item name="android:selectableItemBackground">@drawable/selectable_item_background</item>
</style>
```

res/drawable

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@color/accent_focus" android:state_focused="true"/>
    <item android:drawable="@color/accent_pressed" android:state_pressed="true"/>
    <item android:drawable="@android:color/transparent"/>
</selector>
```

RecyclerView

1. Add the dependency

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile ‘com.android.support:appcompat-v7:21.0.3'
    compile 'com.android.support:recyclerview-v7:21.0.3'
}
```

RecyclerView

2. Code

```
RecyclerView recyclerView = (RecyclerView) findViewById(R.id.recycler);
recyclerView.setLayoutManager(new GridLayoutManager(this, 2));
RecyclerViewAdapter adapter = new RecyclerViewAdapter(items);
adapter.setOnItemClickListener(this);
recyclerView.setAdapter(adapter);
```

RecyclerView

```
public class RecyclerViewAdapter extends
RecyclerView.Adapter<RecyclerViewAdapter.ViewHolder> implements View.OnClickListener {

    ...

    @Override public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_recycler,
parent, false);
        v.setOnClickListener(this);
        return new ViewHolder(v);
    }

    @Override public void onBindViewHolder(ViewHolder holder, int position) {
        ViewModel item = items.get(position);
        holder.text.setText(item.getText());
        holder.image.setImageBitmap(null);
    }
}
```

RecyclerView

```
protected static class ViewHolder extends RecyclerView.ViewHolder {
    public ImageView image;
    public TextView text;

    public ViewHolder(View itemView) {
        super(itemView);
        image = (ImageView) itemView.findViewById(R.id.image);
        text = (TextView) itemView.findViewById(R.id.text);
    }
}
```

CardView

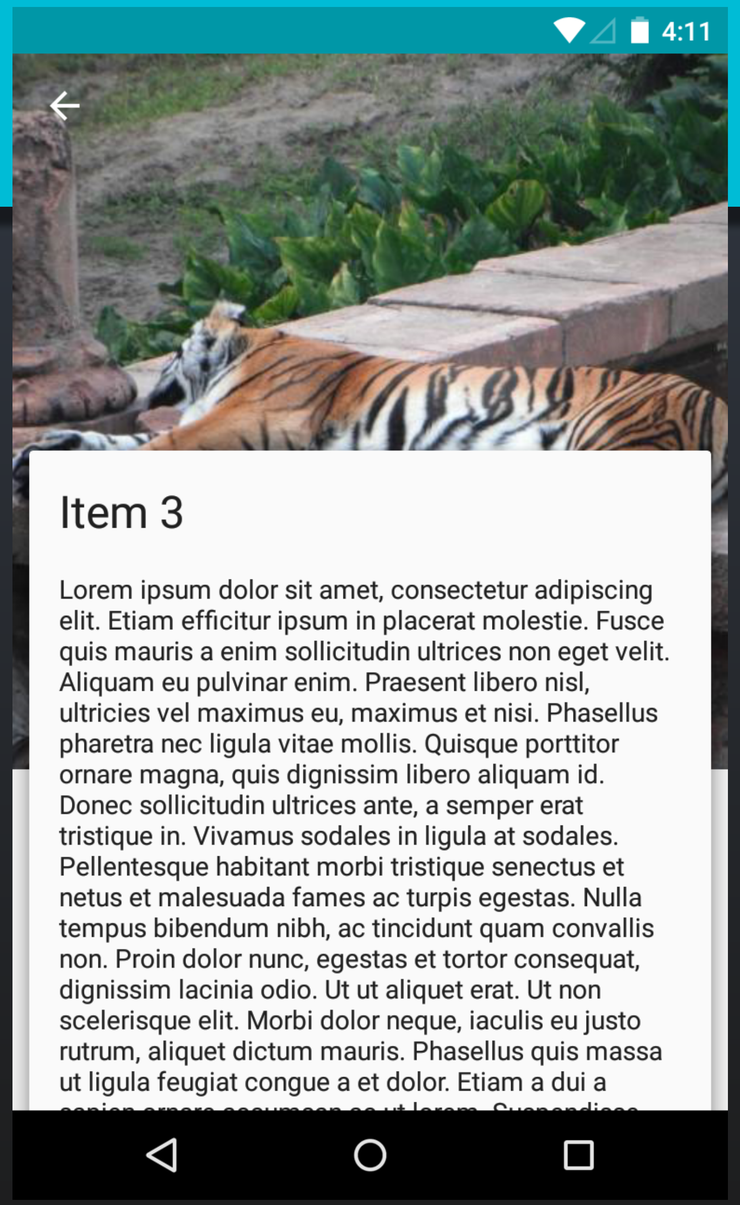
1. Add the dependency

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile ‘com.android.support:appcompat-v7:21.0.3’
    compile ‘com.android.support:recyclerview-v7:21.0.3'
    compile 'com.android.support:cardview-v7:21.0.3'
}
```

CardView

2. Layout

```
<android.support.v7.widget.CardView
android:layout_width="match_parent"
android:layout_height="wrap_content"
app:cardElevation="@dimen/spacing_medium"
app:cardUseCompatPadding="true">

    ...

</android.support.v7.widget.CardView>
```



Palette

1. Add the dependency

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile ‘com.android.support:appcompat-v7:21.0.3’
    compile ‘com.android.support:recyclerview-v7:21.0.3’
    compile ‘com.android.support:cardview-v7:21.0.3'
    compile 'com.android.support:palette-v7:21.0.3'
}
```

Palette

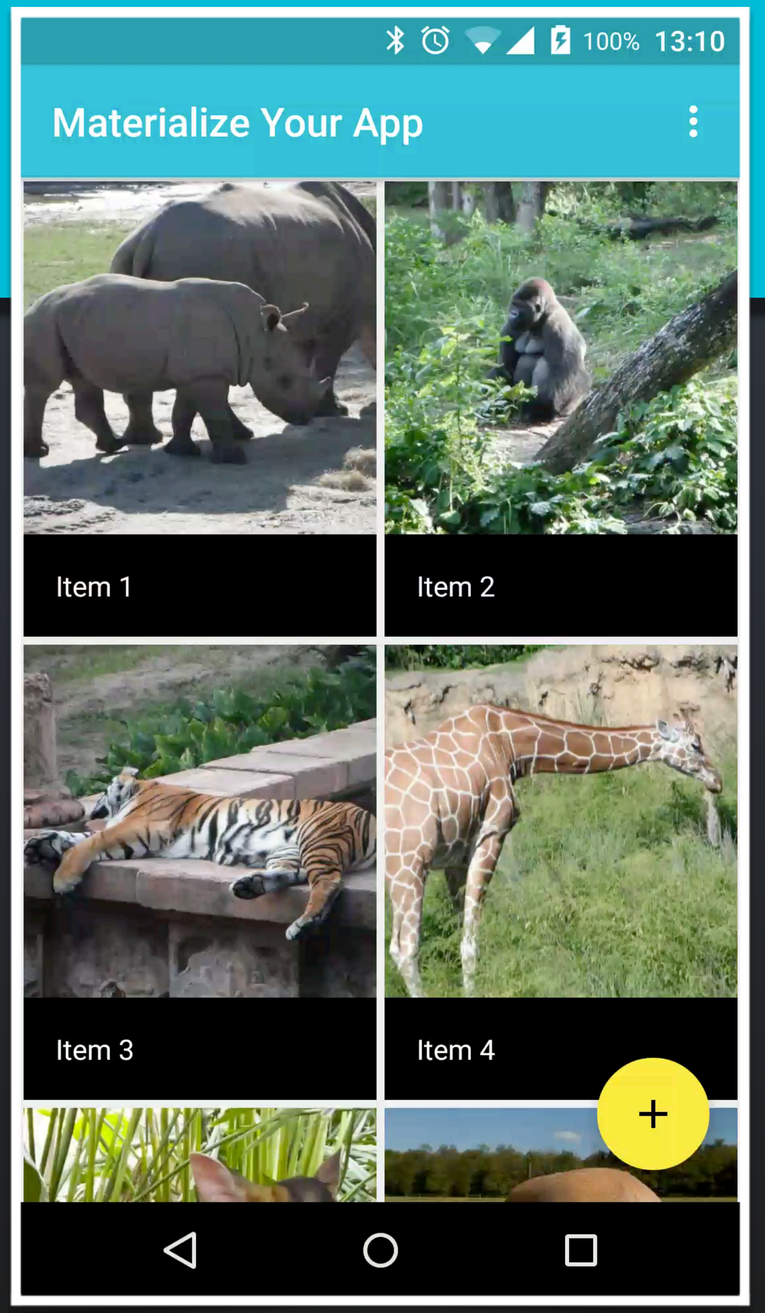
2. Code

```
Palette.generateAsync(bitmap, new Palette.PaletteAsyncListener() {
    public void onGenerated(Palette palette) {
        int primaryDark = getResources().getColor(R.color.primary_dark);
        int primary = getResources().getColor(R.color.primary);
        toolbar.setBackgroundColor(palette.getMutedColor(primary));
        getWindow().setStatusBarColor(palette.getDarkMutedColor(primaryDark));
    }
});
```

Material tips

1. Make elements disappear on scroll

```
recyclerView.setOnScrollListener(

            getSupportActionBar().hide()

            getSupportActionBar().show()

    }
});
```



Material tips

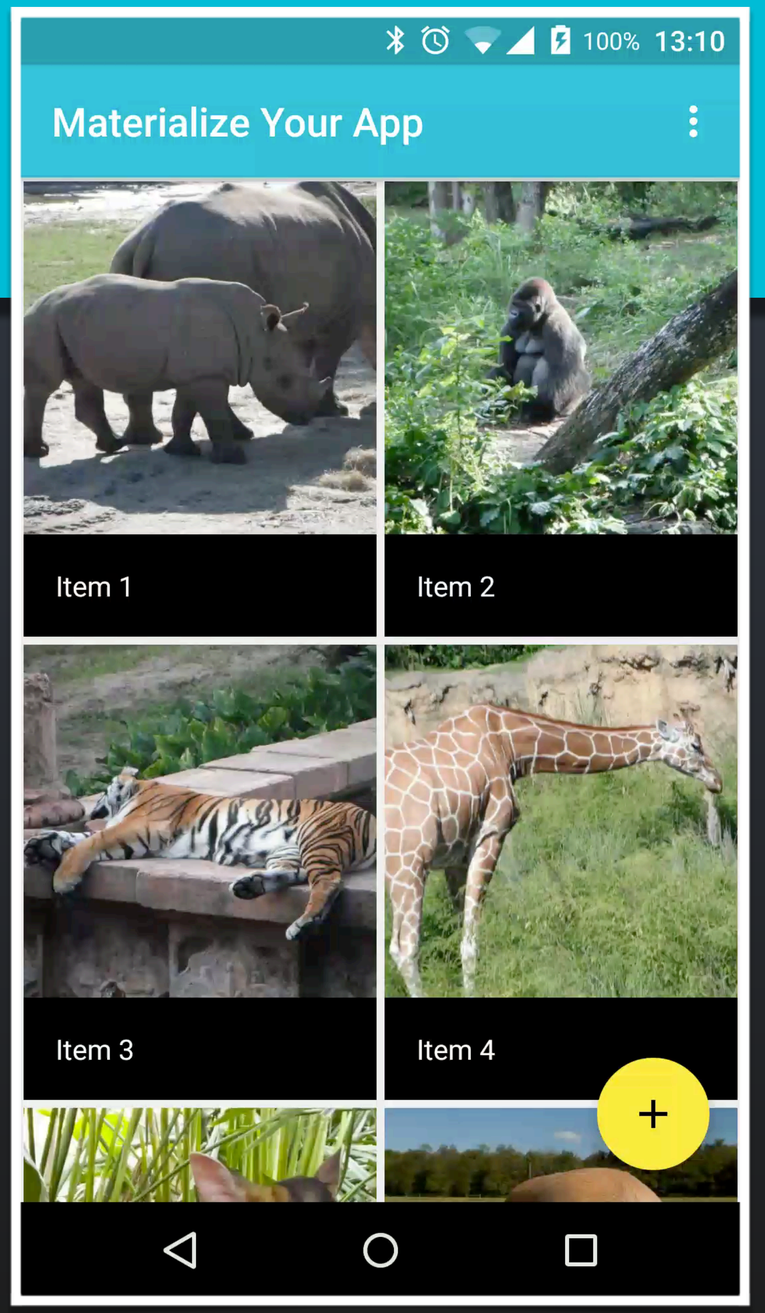
1. Make elements disappear on scroll

```
recyclerView.setOnScrollListener(new RecyclerView.OnScrollListener() {
    @Override public void onScrolled(RecyclerView recyclerView, int dx, int dy) {
        if (dy > 0 && getSupportActionBar().isShowing()){
            getSupportActionBar().hide();
        }else if (dy < 0 && !getSupportActionBar().isShowing()) {
            getSupportActionBar().show();
        }
    }
});
```

Material tips

1. Make elements disappear on scroll

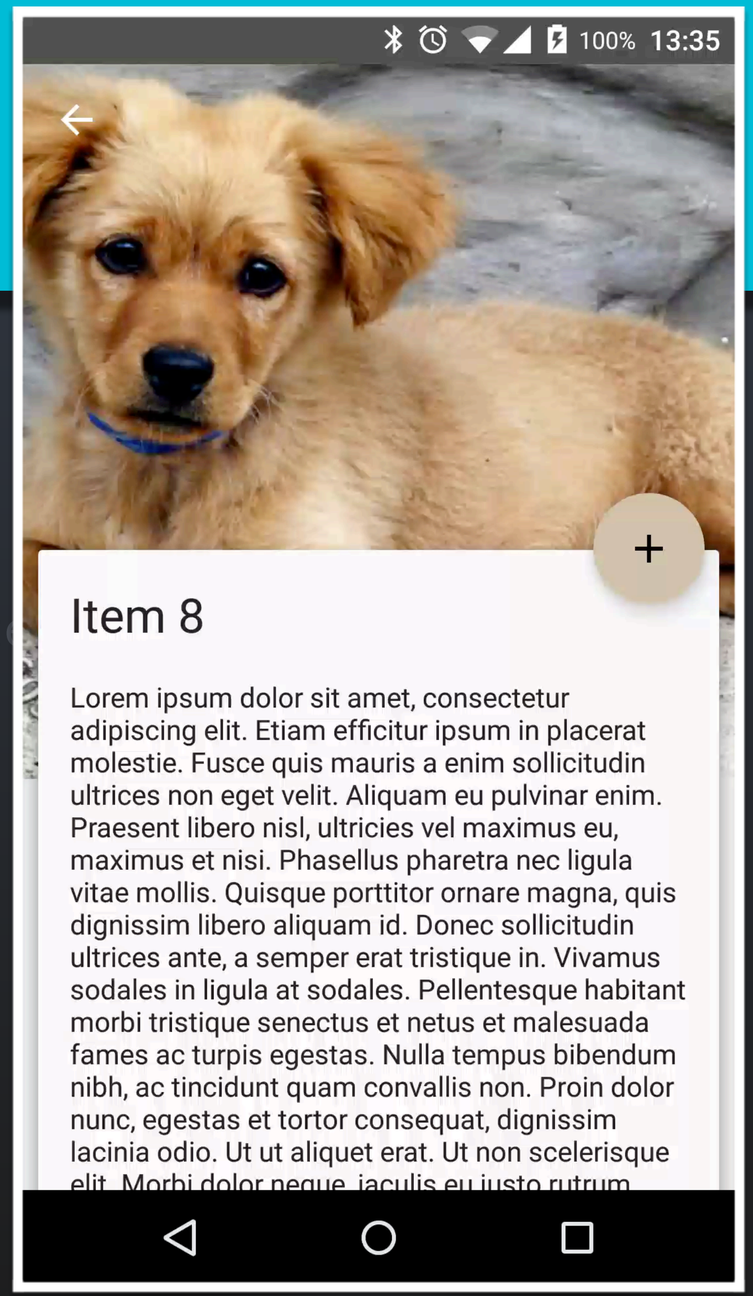
ScrollManager.java



Material tips

2. Parallax + Toolbar fade

```
scrollView.getViewTreeObserver().addOnScrollChangedListe

        })
```



Material tips

2. Parallax + Toolbar fade

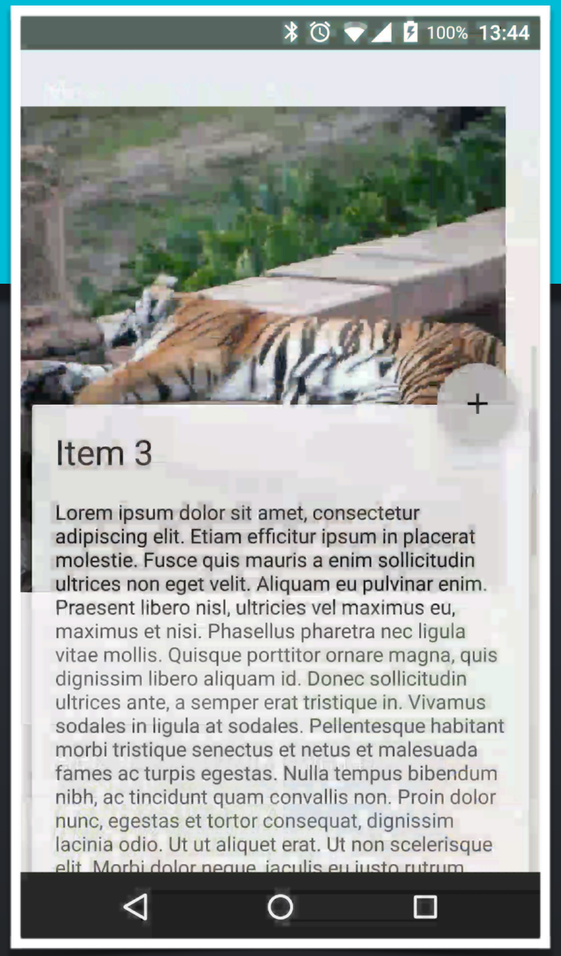
```
scrollView.getViewTreeObserver().addOnScrollChangedListener(
        new ViewTreeObserver.OnScrollChangedListener() {
            @Override
            public void onScrollChanged() {
                int scrollY = scrollView.getScrollY();
                image.setTranslationY(-scrollY / 2);

                ColorDrawable background = (ColorDrawable) toolbar.getBackground();
                background.setAlpha(calculateAlpha(scrollY));
            }
        });
```

Material tips

3. Prevent shared view overlay

By default shared elements are drawn in the decor view's ViewOverlay, on top of the entire window's View hierarchy.

use overlay = true

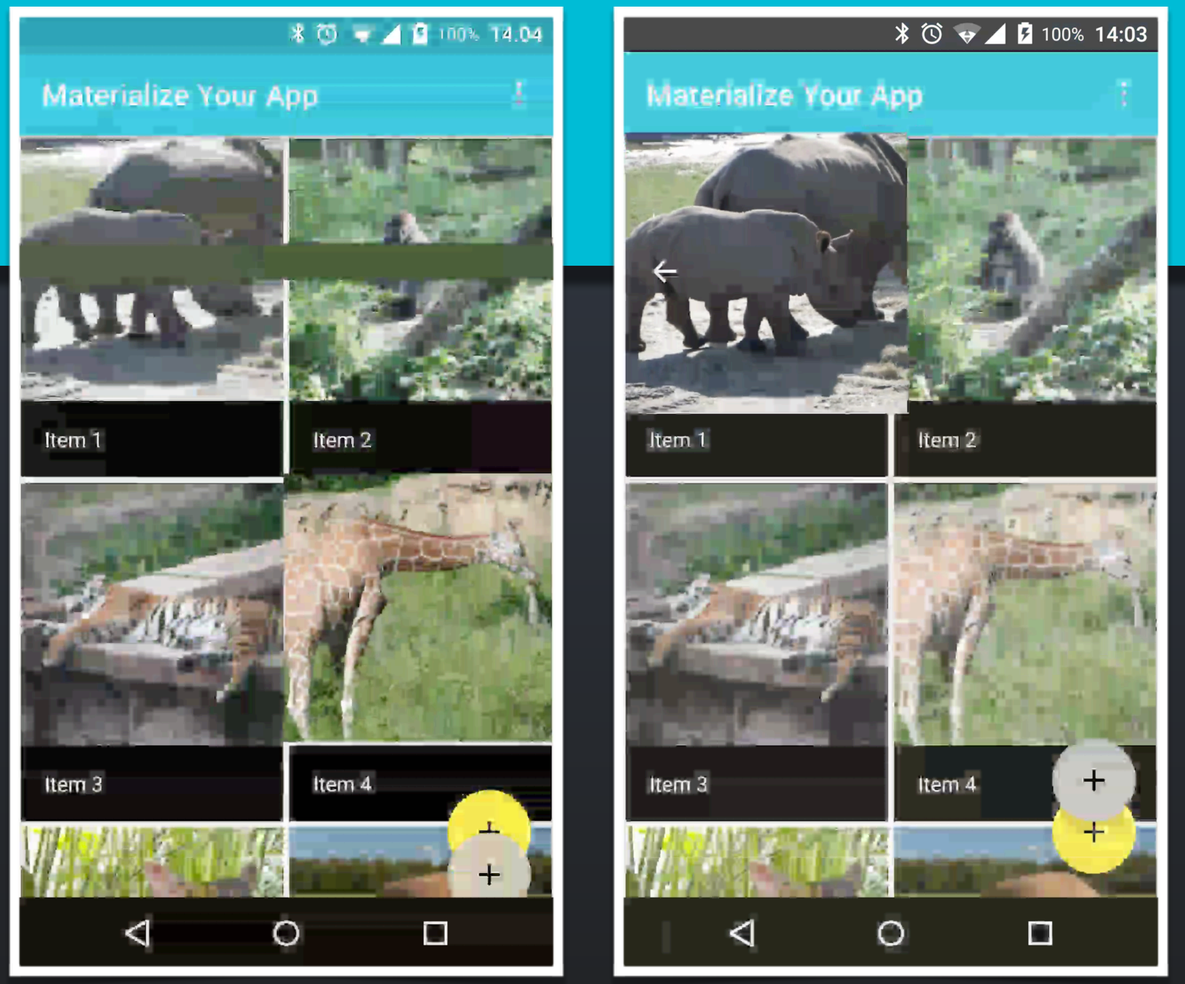


use overlay = false

```
<style name="AppTheme"
    parent="Theme.AppCompat.Light.NoActionBar">
    ...
    <item name="android:windowSharedElementsUseOverlay">false</item>
</style>
```

Material tips

4. Exclude views from transition

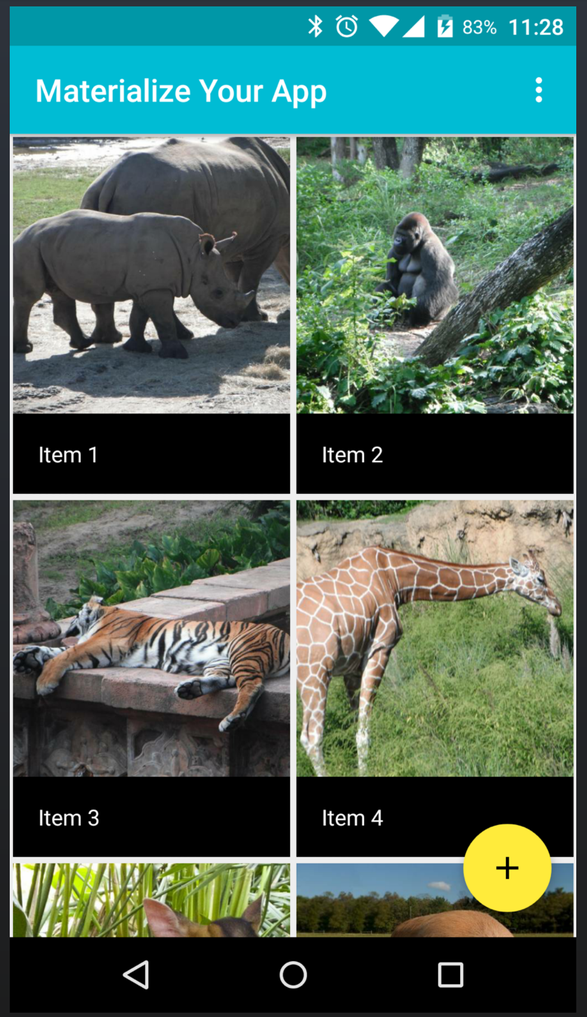


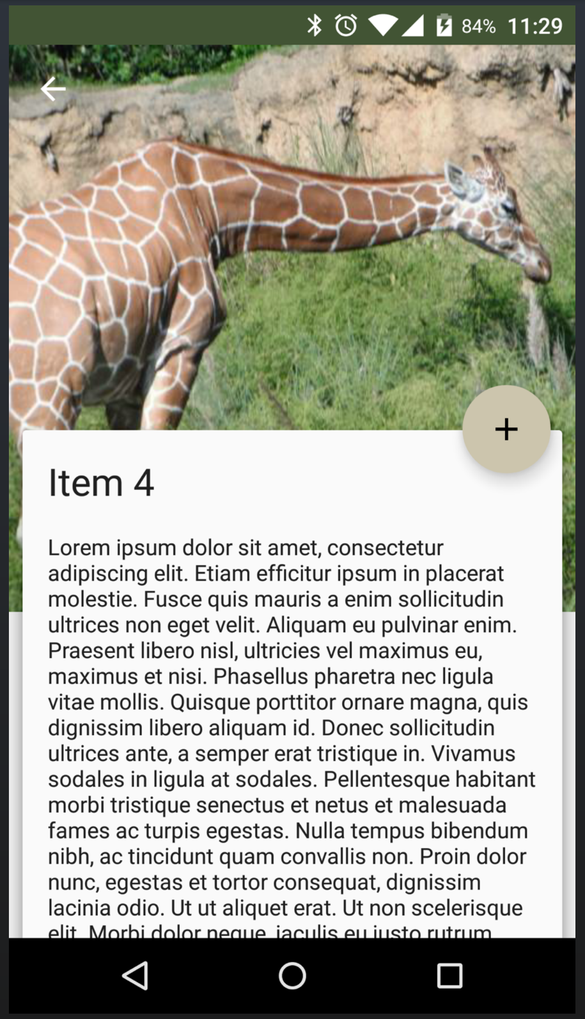

```
Slide transition = new Slide();
transition.excludeTarget(android.R.id.statusBarBackground, true);
getWindow().setEnterTransition(transition);
getWindow().setReturnTransition(transition);
```

Material tips

5. Create a FAB

```
<ImageButton
    android:id="@+id/fab"
    android:layout_gravity="bottom|end"
    style="@style/FabStyle"/>
```





Material tips

5. Create a FAB

```
<style name="FabStyle">
    <item name="android:layout_width">56dp</item>
    <item name="android:layout_height">56dp</item>
    <item name="android:layout_margin">@dimen/spacing_large</item>
    <item name="android:background">@drawable/fab_background</item>
    <item name="android:src">@drawable/ic_add_black</item>
    <item name="android:outlineProvider">background</item>
    <item name="android:stateListAnimator">@anim/fab_elevation</item>
</style>
```

Material tips

5. Create a FAB

```
<selector
xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true">
        <shape
            android:shape="oval">
            <solid android:color="@color/accent_bright"/>
        </shape>
    </item>
    <item>
        <shape
            android:shape="oval">
            <solid android:color="@color/accent"/>

        </shape>
    </item>
</selector>
```

fab_background.xml

```
<ripple
xmlns:android="http://schemas.android.com/apk/res/android"
        android:color="@color/accent_bright">
    <item>
        <shape android:shape="oval">
            <solid android:color="@color/accent"/>
        </shape>
    </item>
</ripple>
```

fab_background.xml (v21)

Material tips

5. Create a FAB

fab_elevation.xml

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true">
        <objectAnimator
            android:propertyName="translationZ"
            android:duration="@android:integer/config_shortAnimTime"
            android:valueFrom="@dimen/elevation_low"
            android:valueTo="@dimen/elevation_high"
            android:valueType="floatType"/>
    </item>
    <item>
        <objectAnimator
            android:propertyName="translationZ"
            android:duration="@android:integer/config_shortAnimTime"
            android:valueFrom="@dimen/elevation_high"
            android:valueTo="@dimen/elevation_low"
            android:valueType="floatType"/>
    </item>
</selector>
```

Questions?

contact@antonioleiva.com

http://antonioleiva.com
http://plus.google.com/+AntonioLeivaGordillo
@lime_cl