

android-[译]掌握CoordinatorLayout



作者 诗不姓唐 (/u/e104f400a701) [+关注](#)

2015.12.30 23:48 字数 1750 阅读 10290 评论 5 喜欢 74

(/u/e104f400a701)

在Google I/O 15 (<https://www.youtube.com/watch?v=7V-flGMDsmE>)上，谷歌发布了一个新的 support library (<http://android-developers.blogspot.com.es/2015/05/android-design-support-library.html>)，里面包含了一些遵循Material Design's spec (<https://www.google.com/design/spec/material-design/introduction.html>)的UI组件，比如，AppBarLayout，CollapsingToolbarLayout 和 CoordinatorLayout。这些组件配合起来使用可以产生强大的效果，那么让我们通过这篇文章来学习如何使用这些组件。

CoordinatorLayout

从名字可以看出，这个ViewGroup是用来协调它的子View的。看下图：



CoordinatorLayout

这个例子中的各个View相互影响，却被和谐的组织在了一起。这就是使用`CoordinatorLayout`最简单的实例：

```

<?xml version="1.0" encoding="utf-8"?>

<android.support.design.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/background_light"
    android:fitsSystemWindows="true"
    >

    <android.support.design.widget.AppBarLayout
        android:id="@+id/main.appbar"
        android:layout_width="match_parent"
        android:layout_height="300dp"
        android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
        android:fitsSystemWindows="true"
        >

        <android.support.design.widget.CollapsingToolbarLayout
            android:id="@+id/main.collapsing"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:layout_scrollFlags="scroll|exitUntilCollapsed"
            android:fitsSystemWindows="true"
            app:contentScrim="?attr/colorPrimary"
            app:expandedTitleMarginStart="48dp"
            app:expandedTitleMarginEnd="64dp"
            >

                <ImageView
                    android:id="@+id/main.backdrop"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:scaleType="centerCrop"
                    android:fitsSystemWindows="true"
                    android:src="@drawable/material_flat"
                    app:layout_collapseMode="parallax"
                    />

                <android.support.v7.widget.Toolbar
                    android:id="@+id/main.toolbar"
                    android:layout_width="match_parent"
                    android:layout_height="?attr/actionBarSize"
                    app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
                    app:layout_collapseMode="pin"
                    />

            </android.support.design.widget.CollapsingToolbarLayout>
        </android.support.design.widget.AppBarLayout>

        <android.support.v4.widget.NestedScrollView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:layout_behavior="@string/appbar_scrolling_view_behavior"
            >

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:textSize="20sp"
                android:lineSpacingExtra="8dp"
                android:text="@string/lorem"
                android:padding="@dimen/activity_horizontal_margin"
                />

        </android.support.v4.widget.NestedScrollView>

        <android.support.design.widget.FloatingActionButton
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:layout_margin="@dimen/activity_horizontal_margin"
            android:src="@drawable/ic_comment_24dp"
            app:layout_anchor="@+id/main.appbar"
            app:layout_anchorGravity="bottom|right|end"
            />

    </android.support.design.widget.CoordinatorLayout>

```

看一下上面Layout的结构，CoordinatorLayout 包含三个子View：
 一个AppBarLayout，一个scrollable View，一个指定了锚点的 FloatingActionButton。

```

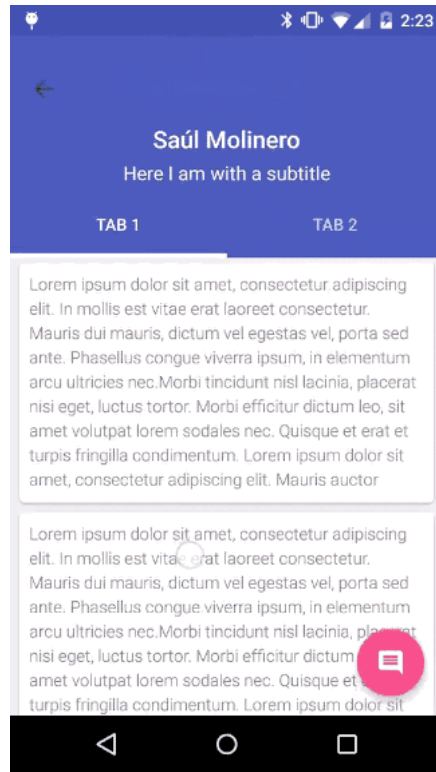
<CoordinatorLayout>
  <AppBarLayout/>
  <scrollView/>
  <FloatingActionButton/>
</CoordinatorLayout>

```

AppBarLayout



首先，AppBarLayout 是一个 LinearLayout，它的子View默认纵向排列，可以通过一些参数控制子View的滑动行为。这么说你还是很理解，所以无图无真相，上GIF：



AppBarLayout

这张图最上面是一个可折叠图片（collapsing image），图片下面的蓝色View就是 AppBarLayout，它包含了一个 Toolbar，一个有标题和子标题的 LinearLayout，一个带有 Tab 的 TabLayout。

```
<AppBarLayout>
  <CollapsingToolbarLayout
    app:layout_scrollFlags="scroll|snap"
  />

  <Toolbar
    app:layout_scrollFlags="scroll|snap"
  />

  <LinearLayout
    android:id="@+id/title_container"
    app:layout_scrollFlags="scroll|enterAlways"
  />

  <TabLayout /> <!-- no flags -->
</AppBarLayout>
```

AppBarLayout 的直接子View的操控行为，可以通过给子View添加 layout_scrollFlags 属性来控制。关于这个属性的值：scroll，在这个例子中用到了这个值。如果一个子View没有赋值 scroll，那么滑动的时候，它就会一直静态显示，而其他 scroll 的View就会被划到它的后面隐藏。

另一个值 snap 的作用是避免一个View停留在动画的中间状态，也就是说滑动结束的时候一个View要么全部显示，要么全部隐藏，不会展示View的部分。

上面的 LinearLayout 指定了 enterAlways，所以下拉的时候，它就会一直出现。TabLayout 没有指定，所以它一直静态显示。

由此，给子View使用不同的 layout_scrollFlags 就会生成不同的 AppBarLayout。全部属性值参照官方文档

(<https://developer.android.com/intl/es/reference/android/support/design/widget/AppBarLayout.LayoutParams.html#CONSTANTS>)，文章最后我也会提供几个放在Github上的实例。

AppBarLayout flags



*SCROLL_FLAG_ENTER_ALWAYS : ((entering) / (scrolling on screen))下拉的时候，这个View也会跟着滑出。

*SCROLL_FLAG_ENTER_ALWAYS_COLLAPSED : 另一种 enterAlways ，但是只显示折叠后的高度。

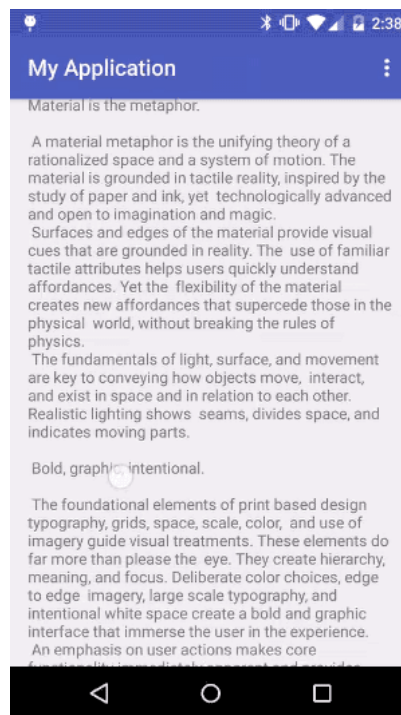
*SCROLL_FLAG_EXIT_UNTIL_COLLAPSED : ((exiting) / (scrolling off screen))上拉的时候，这个View会跟着滑动直到折叠。

*SCROLL_FLAG_SCROLL : 跟着滑动方向滑动。

*SCROLL_FLAG_SNAP : 滑动结束的时候，如果这个View部分显示，它就会滑动到离它最近的上边缘或下边缘。

CoordinatorLayout Behaviors

打开Android Studio (>= 1.4)，用模版 Scrolling Activity 新建一个项目，然后直接编译运行。看到：



Scrolling Activity

查看生成的代码，没发现滑动时候Fab大小变化动画的相关代码，Why？

答案在 FloatingActionButton 的源代码中，感谢Android Studio v1.2自带了反编译源代码的功能，ctrl/cmd + click 我们来 FloatingActionButton 的源码中究竟干了什么。

```

/*
 * Copyright (C) 2015 The Android Open Source Project
 *
 * Floating action buttons are used for a
 * special type of promoted action.
 * They are distinguished by a circled icon
 * floating above the UI and have special motion behaviors
 * related to morphing, launching, and the transferring anchor point.
 *
 * blah.. blah..
 */
@CoordinatorLayout.DefaultBehavior(
    FloatingActionButton.Behavior.class)
public class FloatingActionButton extends ImageButton {
    ...

    public static class Behavior
        extends CoordinatorLayout.Behavior<FloatingActionButton> {

        private boolean updateFabVisibility(
            CoordinatorLayout parent, AppBarLayout appBarLayout,
            FloatingActionButton child {

            if (a long condition) {
                // If the anchor's bottom is below the seam,
                // we'll animate our FAB out
                child.hide();
            } else {
                // Else, we'll animate our FAB back in
                child.show();
            }
        }
    }

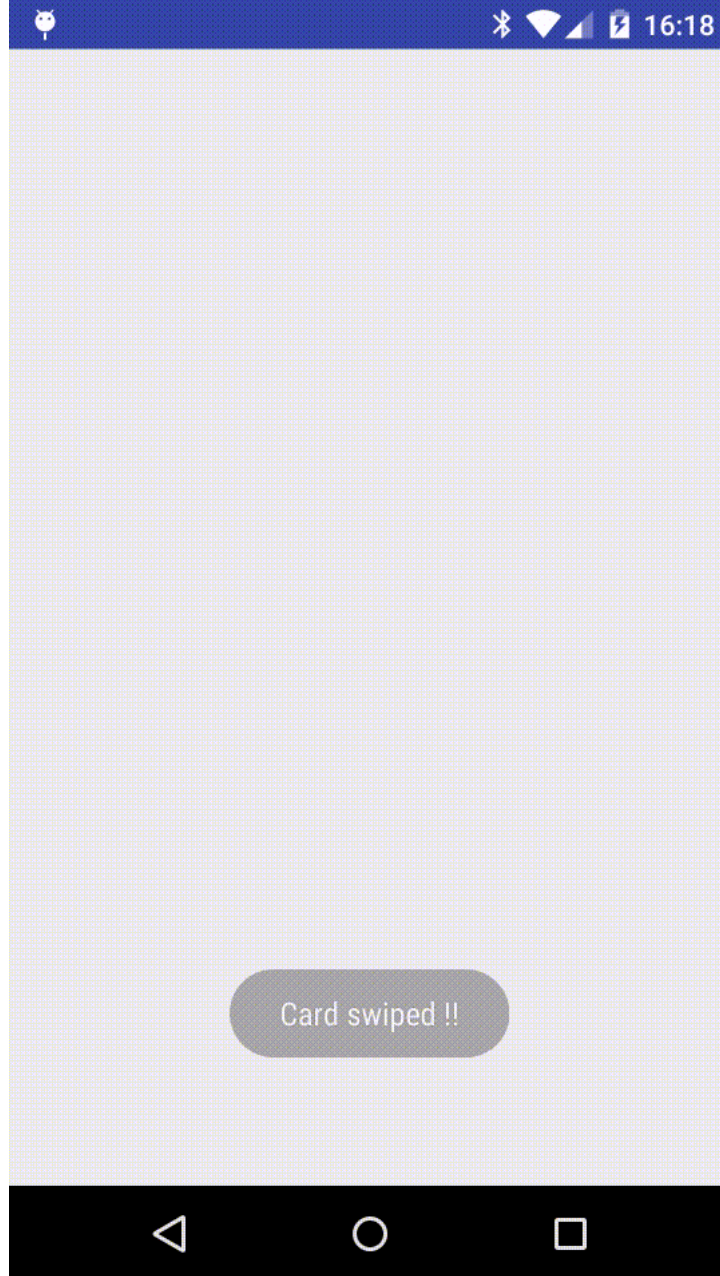
    ...
}

```

其实那个大小变化动画是design包中的 `Behavior` 控制的，上面的 `CoordinatorLayout.Behavior<FloatingActionButton>` 控制显示或隐藏FAB，interesting？

SwipeDismissBehavior

在 design support library 中，我们发现了 `SwipeDismissBehavior`，有了它，我们可以在 `CoordinatorLayout` 中轻松实现滑动删除功能。



swipe

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_swipe_behavior);
    mCardView = (CardView) findViewById(R.id.swype_card);

    final SwipeDismissBehavior<CardView> swipe
        = new SwipeDismissBehavior();

    swipe.setSwipeDirection(
        SwipeDismissBehavior.SWIPE_DIRECTION_ANY);

    swipe.setListener(
        new SwipeDismissBehavior.OnDismisslistener() {
            @Override public void onDismiss(View view) {
                Toast.makeText(SwipeBehaviorExampleActivity.this,
                    "Card swiped !!", Toast.LENGTH_SHORT).show();
            }
        });

    @Override
    public void onDragStateChanged(int state) {}
});

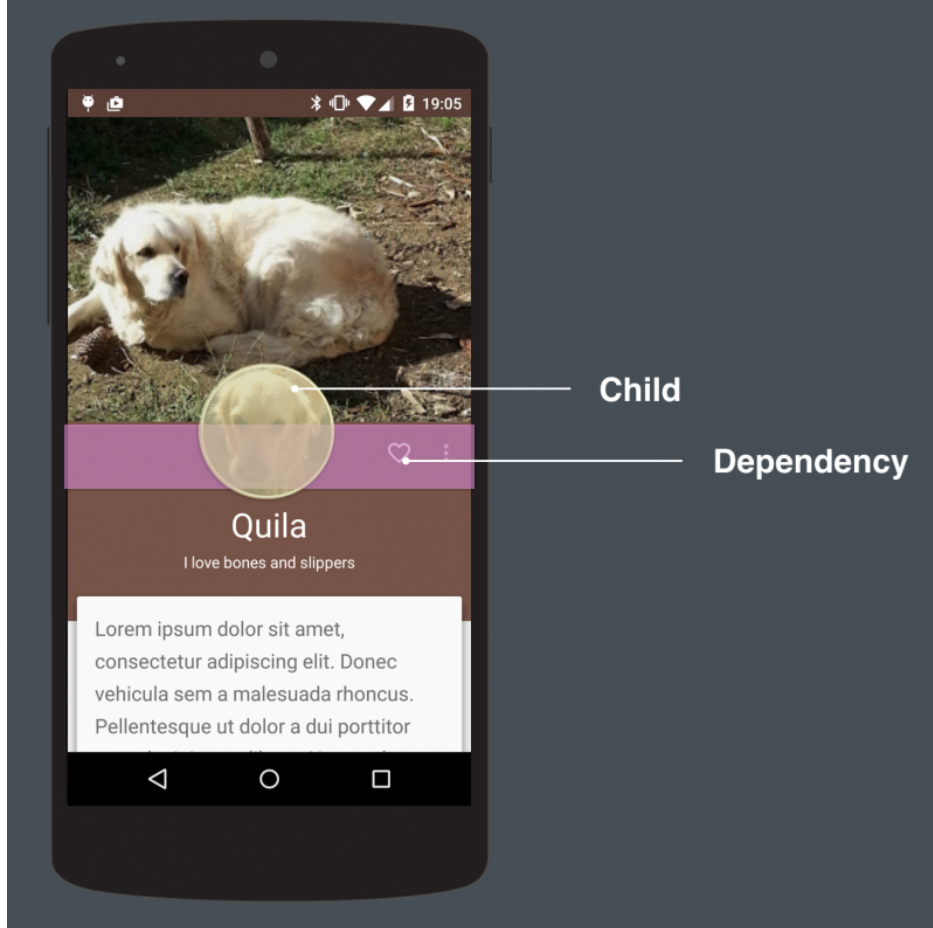
LayoutParams coordinatorParams =
    (LayoutParams) mCardView.getLayoutParams();

coordinatorParams.setBehavior(swipe);
}
```

Custom Behaviors

自定义Behavior并不难，首先介绍两个元素：child 和 dependency。

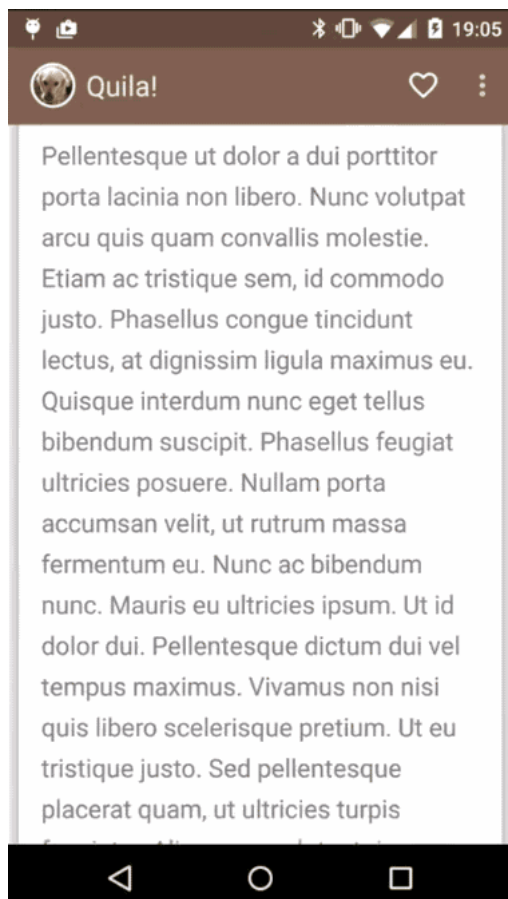




child

Childs and dependencies

要改变行为的那个View就是child，dependency是作为触发器影响child的那个View。这个例子中child是 `ImageView`，dependency是 `Toolbar`，然后，`Toolbar` 滑动的时候，`ImageView` 跟着滑动。



gif



下面开始创建自定义Behavior，首先，继承 `CoordinatorLayout.Behavior<T>`，`T` 就是child的类型，上面例子是那个 `ImageView`，然后我们重写方法

`layoutDependsOn` onDependentViewChanged

每当UI变化的时候就会调用 `layoutDependsOn`，鉴定完dependency后一定要返回true。上面例子中用户一滑动就会自动调用 `layoutDependsOn`，然后开始控制child 的行为。

```
@Override
public boolean layoutDependsOn(
    CoordinatorLayout parent,
    CircleImageView child,
    View dependency) {

    return dependency instanceof Toolbar;
}
```

`layoutDependsOn` 返回true后就开始调用 `onDependentViewChanged`，在这个方法中我们利用 `dependency`来实现动画，转换，动作。

```
public boolean onDependentViewChanged(
    CoordinatorLayout parent,
    CircleImageView avatar,
    View dependency) {

    modifyAvatarDependingDependencyState(avatar, dependency);
}

private void modifyAvatarDependingDependencyState(
    CircleImageView avatar, View dependency) {
    // avatar.setY(dependency.getY());
    // avatar.setBlahBlah(dependency.blah / blah);
}
```

放在一起就是：

```
public static class AvatarImageBehavior
    extends CoordinatorLayout.Behavior<CircleImageView> {

    @Override
    public boolean layoutDependsOn(
        CoordinatorLayout parent,
        CircleImageView child,
        View dependency) {

        return dependency instanceof Toolbar;
    }

    public boolean onDependentViewChanged(
        CoordinatorLayout parent,
        CircleImageView avatar,
        View dependency) {
        modifyAvatarDependingDependencyState(avatar, dependency);
    }

    private void modifyAvatarDependingDependencyState(
        CircleImageView avatar, View dependency) {
        // avatar.setY(dependency.getY());
        // avatar.setBlahBlah(dependency.blah / blah);
    }
}
```

YoungPeanut.github.io (<http://youngpeanut.github.io/>)

博客 (<http://youngpeanut.github.io/2015-12-23/android-%5B%E8%AF%91%5D%E6%8E%8C%E6%8F%A1CoordinatorLayout/>)

原文 (<http://saulmm.github.io/mastering-coordinator/>)


Resources

* Coordinator Behavior Example

(<https://github.com/saulmm/CoordinatorBehaviorExample>)- Github

* Coordinator Examples (<https://github.com/saulmm/CoordinatorExamples>)- Github





诗不姓唐 (/u/e104f400a701)

写了 6480 字，被 76 人关注，获得了 127 个喜欢 (/u/e104f400a701)




+ 关注

好诗不易写，亦不易遇，此乃妙手难，偶得亦难。诗不姓，词无名。有姓不是唐，无名亦诗词。

如果觉得我的文章对您有用，请随意打赏。您的支持将鼓励我继续创作！

赞赏支持

👍 喜欢 (/sign_in) | 74



更多分享

(http://cwb.assets.jianshu.io/notes/images/2201902/weibo



(/sign_in) 后发表评论

5条评论 只看作者 按喜欢排序 按时间正序 按时间倒序



帅的人都叫这个 (/u/2179c66e9f4a)

3楼 · 2016.08.18 08:04 (/u/2179c66e9f4a)

赞一个

👍 2人赞

💬 回复




dongyiLi (/u/343dd47ae9f8)

2楼 · 2016.07.07 10:47 (/u/343dd47ae9f8)

写的详细，学习了

👍 赞

💬 回复




最后的大魔王 (/u/bb3ff9ef07e1)

4楼 · 2016.09.29 11:55 (/u/bb3ff9ef07e1)

写的太棒了，博主翻译的也特别到位，国外大牛确实不同凡响！

👍 赞

💬 回复




xujun9411 (/u/ca9b3e19f454)

5楼 · 2016.10.18 21:06 (/u/ca9b3e19f454)

写的太棒了，学习了

👍 赞

💬 回复



罂粟般的女人 (/u/2ca4c8da6605)

6楼 · 2017.01.12 02:52 (/u/2ca4c8da6605)

大佬写的不太全啊。CollapsingToolbarLayout没写

👍 赞

💬 回复



被以下专题收入，发现更多相似内容

[技术文](#)

[Android.
..](#)

[技术干货](#)

[Android
知识](#)

[Android.
..](#)

[移动开发](#)

[Coordin.
..](#)

[安卓
metr...](#)

[MD控件](#)

[android..
.](#)

[Android.
..](#)