Android UI Programming 1 Basic

Prepared by: Vivian Sun

Name	Version	Date	Prepared By
Android UI Programming	1.0	2013/4/3	Vivian
Android UI Programming	2.0	2016/10/26	Vivian

Content

Abstract

This document shows some android user interface programming related information.

UI Design的概念

UI即User Interface（用户界面）的简称。UI设计则是指对软件的人机交互、操作逻辑、界面美观等各个方面的整体设计。


ASUS
ASUS
ePrice

分享的主题：Android UI Programming相关的内容。

Agenda

Android View

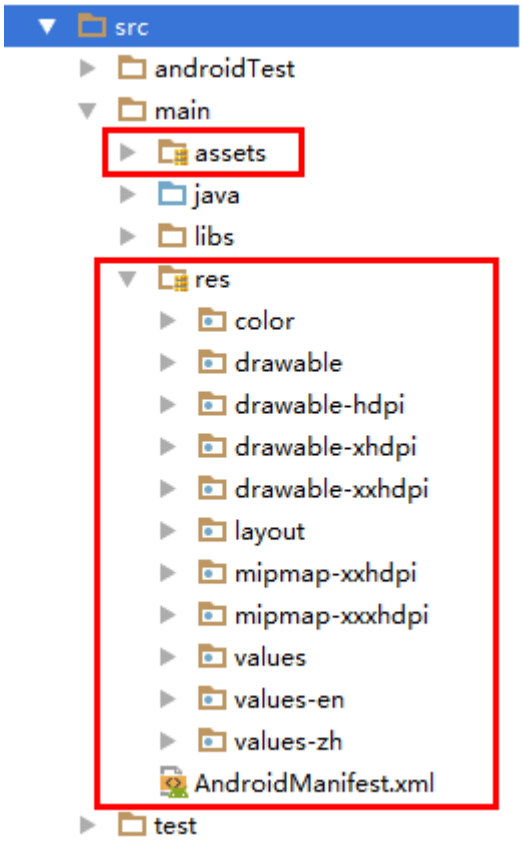
Android Layout

Android Customized View

Android View Event

Android User Interface

UI related Contents in Project



res

存放程序所需要的资源文件（可编译的资源文件），常见的目录有：

Path	Description	Type
res/animator/	定义动画属性	XML 文件
res/anim/	被编译进逐帧动画（frame by frame animation）或补间动画(tweened animation)对象	XML 文件
res/color/	定义颜色状态的列表	XML 文件
res/layout/	存放被编译为屏幕布局(或屏幕的一部分)	XML 文件

res/menu/	用来定义应用的菜单	XML 文件
res/drawable/	.png, .9.png, .jpg, .gif 等	图片文件
res/raw/	直接复制到设备中的任意文件，它们无需编译	Raw 文件
res/values/	存放可以被编译成很多种类型的资源文件 a. array.xml ： 定义数组 b. colors.xml ： 定义 color drawable 和颜色的字符串值。 c. dimens.xml 定义尺寸值(dimension value)。 d. strings.xml 定义字符串(string)值。 e. styles.xml 定义样式(style)对象。	XML 格式
res/xml/	任意的 XML 文件，在运行时可通过调用 Resources.getXML()读取	XML 格式

assets 目录

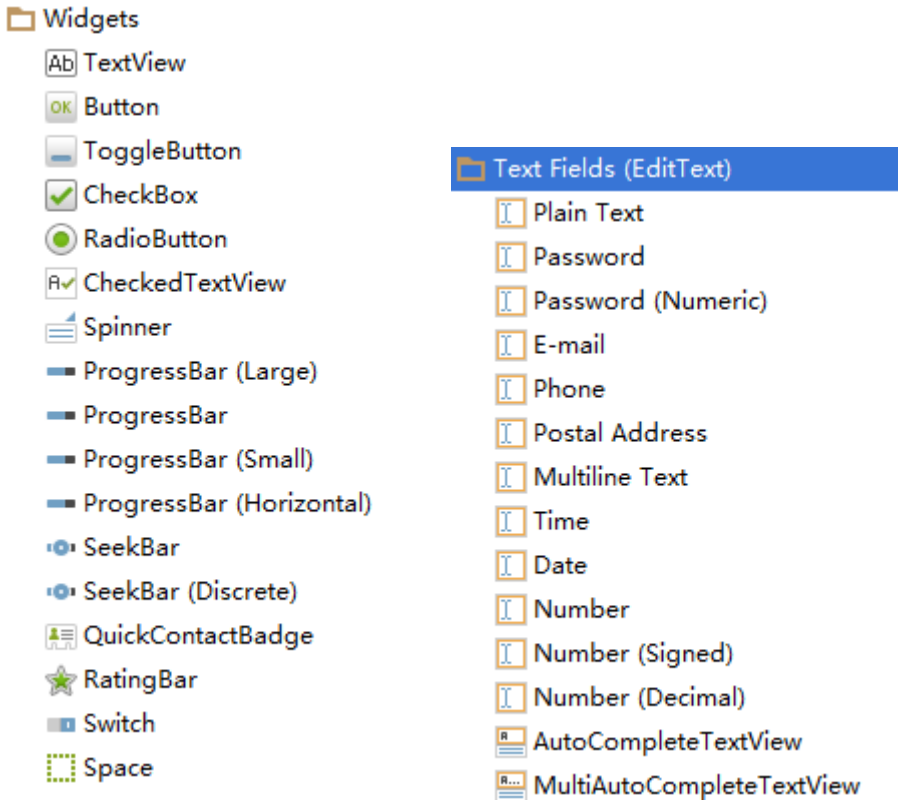
存放的原生资源文件。例如txt, html 等

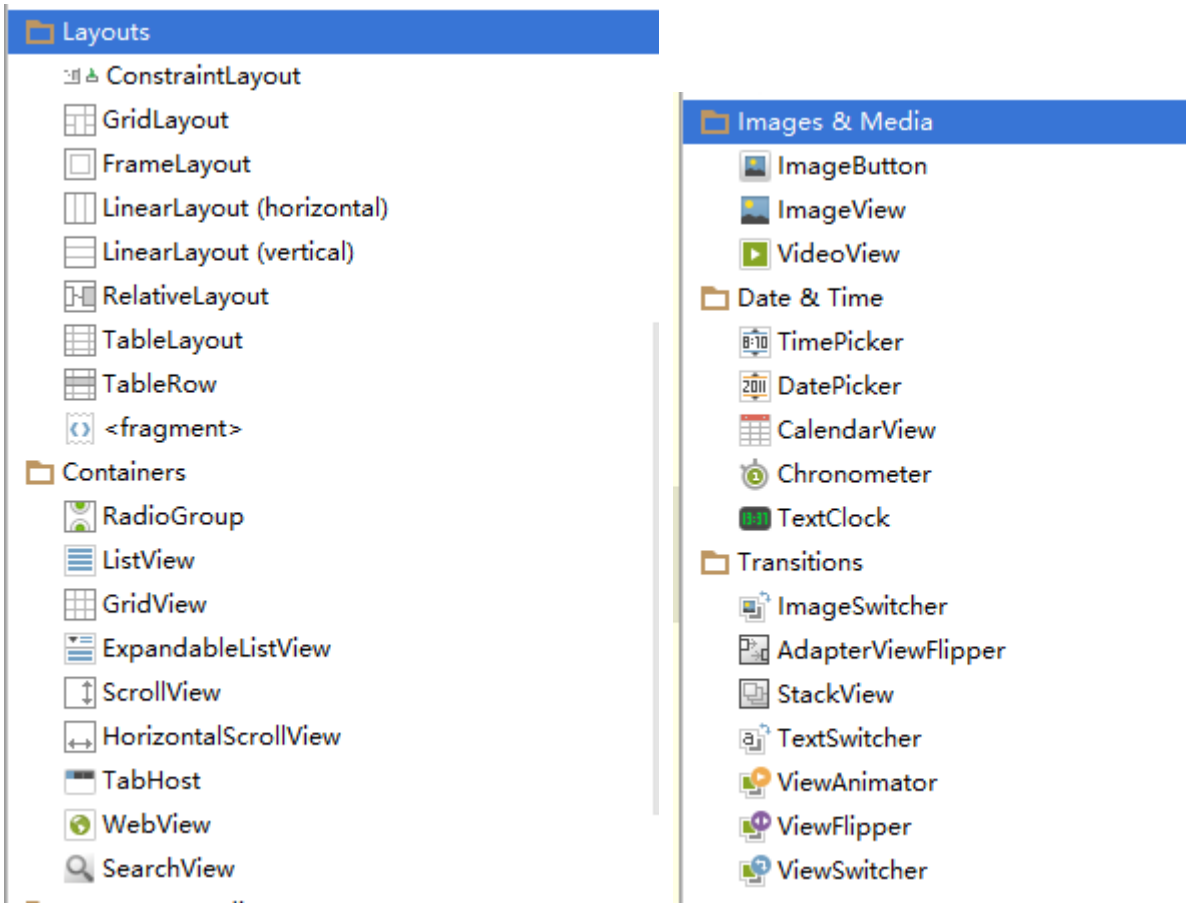
Android View

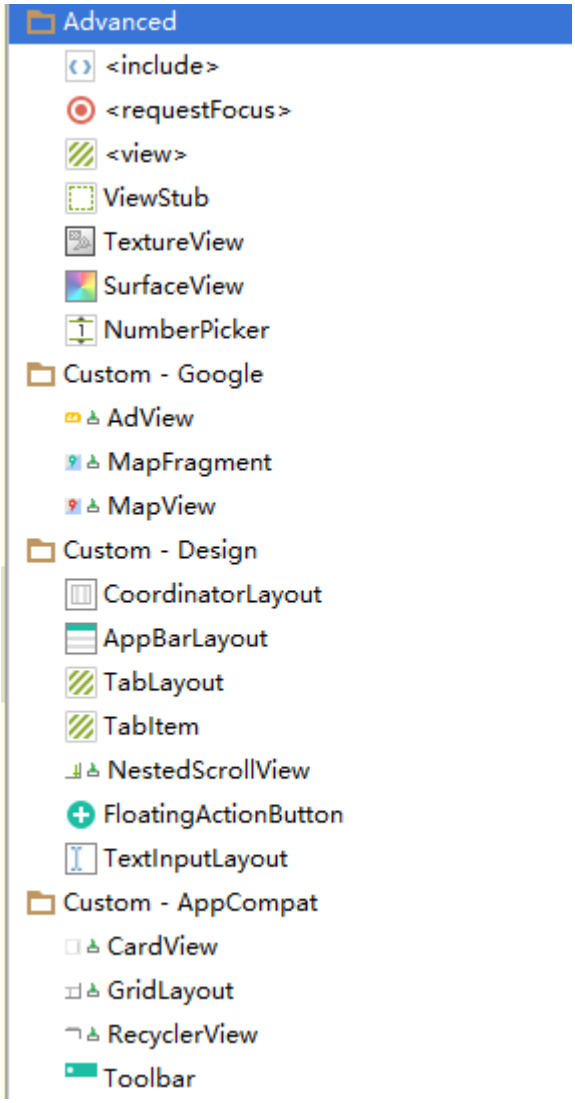
What is Android View?

- ✧ Android UI最基本的单元是View
- ✧ 继承自android.view.View
- ✧ 一个View代表一个控件（widget）
- ✧ 包括button、textview、edittext等。

View类是所有与用户交互的组件的Widgets的基类，Android_View继承体系图：




Layouts
ConstraintLayout
GridLayout
FrameLayout
LinearLayout (horizontal)
LinearLayout (vertical)
RelativeLayout
TableLayout
TableRow
<fragment>
Containers
RadioGroup
ListView
GridView
ExpandableListView
ScrollView
HorizontalScrollView
TabHost
WebView
SearchView
Images & Media
ImageButton
ImageView
VideoView
Date & Time
TimePicker
DatePicker
CalendarView
Chronometer
TextClock
Transitions
ImageSwitcher
AdapterViewFlipper
StackView
TextSwitcher
ViewAnimator
ViewFlipper
ViewSwitcher


Advanced
<include>
<requestFocus>
<view>
ViewStub
TextureView
SurfaceView
NumberPicker
Custom - Google
AdView
MapFragment
MapView
Custom - Design
CoordinatorLayout
AppBarLayout
TabLayout
TabItem
NestedScrollView
FloatingActionButton
TextInputLayout
Custom - AppCompat
CardView
GridLayout
RecyclerView
Toolbar

Android Layout

布局就像容器，里面可以装下很多控件。布局里面还可以套用其他的布局。可以实现界面的多样性与设计的灵活性

LinearLayout 线性布局

所有元素都水平或垂直放置，可以嵌套其它布局


```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFFFFF"
    tools:context=".MainActivity" >

        // 这里第一行显示标签为一个水平布局
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
        <EditText
            android:id="@+id/msg"
            android:inputType="number"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="">
        </EditText>
    </LinearLayout>
```

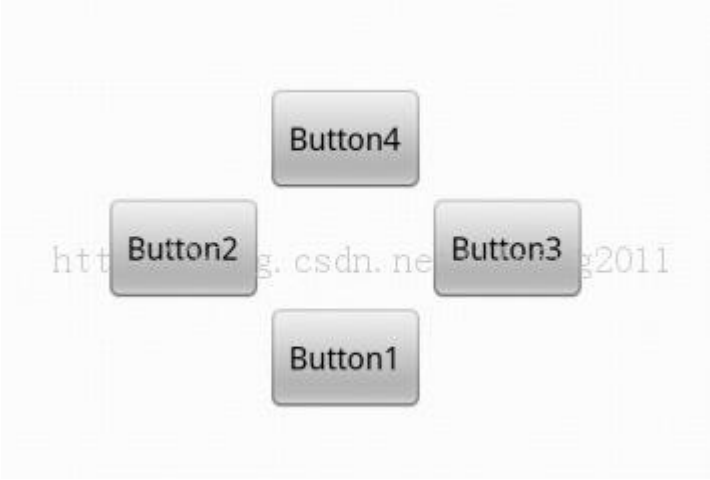
RelativeLayout 相对布局

RelativeLayout就是以相对的方式定位布局，允许子元素指定他们相对于其它元素或父元素的位置（通过ID指定）。因此如果第一个元素在屏幕的中央，那么相对于这个元素的其它元素将以屏幕中央的相对位置来排列。

相对布局的属性比较多，但用起来比较灵活. 关键的属性

android:layout_below 在某元素的下方
android:layout_above 在某元素的的上方
android:layout_toLeftOf 在某元素的左边
android:layout_toRightOf 在某元素的右边

android:layout_alignTop 本元素的上边缘和某元素的的上边缘对齐
android:layout_alignLeft 本元素的左边缘和某元素的的左边缘对齐
android:layout_alignBottom 本元素的下边缘和某元素的的下边缘对齐
android:layout_alignRight 本元素的右边缘和某元素的的右边缘对齐



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <Button android:id="@+id/btn1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:layout_centerHorizontal="true"
        android:text="Button1"
        ></Button>
    <Button android:id="@+id/btn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@id/btn1"
        android:layout_above="@id/btn1"
        android:text="Button2"
        ></Button>
    <Button android:id="@+id/btn3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/btn1"
        android:layout_above="@id/btn1"
        android:text="Button3"
        ></Button>
    <Button android:id="@+id/btn4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/btn2"
        android:layout_toLeftOf="@id/btn3"
        android:layout_above="@id/btn2"
        android:text="Button4"
        ></Button>
</RelativeLayout>
```

TableLayout 表格布局

表格布局，类似于HTML的Table和WPF的Grid。

通过TableRow来定义一行，如果一个控件占用多列可以设置android:layout_span。

默认情况下一个控件是按顺序放置在每一列的(column 0, column 1….)，也可以通过android:layout_column指定放在哪一列。

如果一列内容过长或者过短，可以通过android:stretchColumns和android:shrinkColumns来增加或者减少此列的宽度。

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:shrinkColumns="0,1,2"  // 设置三列都可以收缩
    android:stretchColumns="0,1,2" // 设置三列都可以拉伸 如果不设置这个，那个显
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <TableRow android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <Button android:gravity="center"
            android:padding="10dp"
            android:text="Button1">
        </Button>

        <Button android:gravity="center"
            android:padding="10dp"
            android:text="Button2">
        </Button>
        <Button android:gravity="center"
            android:padding="10dp"
            android:text="Button3">
        </Button>
    </TableRow>
```

FrameLayout 框架布局

将所有的子元素放在整个界面的左上角，后面的子元素直接覆盖前面的子元素，所以用的比较少。

所以用的比较少。

AbsoluteLayout 绝对布局 （已废弃）

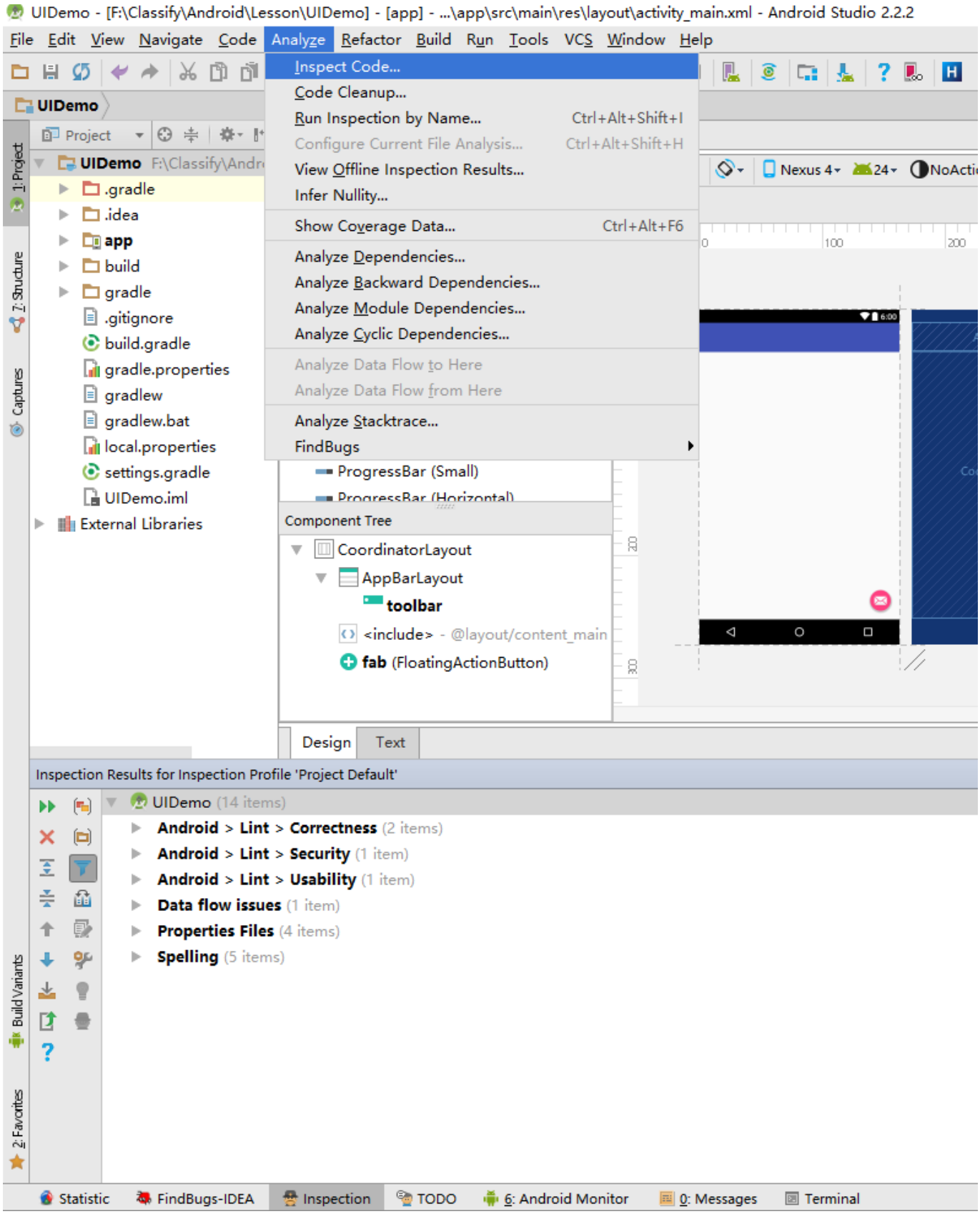
Android SDK 工具布局优化工具

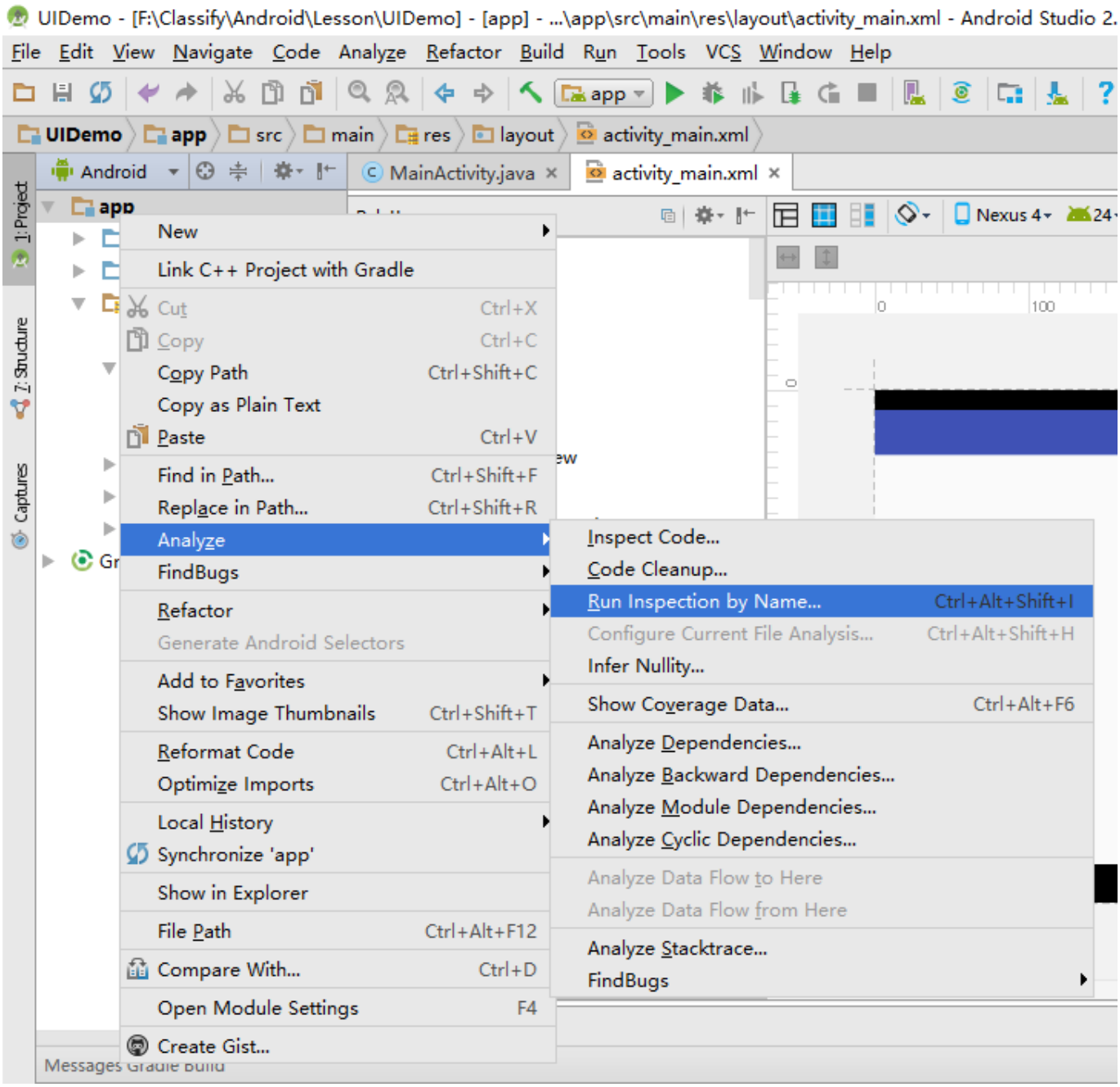
lint （ sdk 16之前是layoutopt）

位置：sdk\tools

Android Studio使用Lint进行代码检查 ： http://blog.csdn.net/u014651216/article/details/50820748

android lint-checks http://tools.android.com/tips/lint-checks


UIDemo - [F:\Classify\Android\Lesson\UIDemo] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 2.2.2
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Inspect Code...
Code Cleanup...
Run Inspection by Name... Ctrl+Alt+Shift+I
Configure Current File Analysis... Ctrl+Alt+Shift+H
View Offline Inspection Results...
Infer Nullity...
Show Coverage Data... Ctrl+Alt+F6
Analyze Dependencies...
Analyze Backward Dependencies...
Analyze Module Dependencies...
Analyze Cyclic Dependencies...
Analyze Data Flow to Here
Analyze Data Flow from Here
Analyze Stacktrace...
FindBugs
UIDemo
Project
UIDemo F:\Classify\Andr
.gradle
.idea
app
build
gradle
.gitignore
build.gradle
gradle.properties
gradlew
gradlew.bat
local.properties
settings.gradle
UIDemo.iml
External Libraries
ProgressBar (Small)
Component Tree
CoordinatorLayout
AppBarLayout
toolbar
<include> - @layout/content_main
fab (FloatingActionButton)
Design
Text
Nexus 4
Inspection Results for Inspection Profile 'Project Default'
UIDemo (14 items)
Android > Lint > Correctness (2 items)
Android > Lint > Security (1 item)
Android > Lint > Usability (1 item)
Data flow issues (1 item)
Properties Files (4 items)
Spelling (5 items)
Statistic
FindBugs-IDEA
Inspection
TODO
6: Android Monitor
0: Messages
Terminal


UIDemo - [F:\Classify\Android\Lesson\UIDemo] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 2.
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
app
UIDemo app src main res layout activity_main.xml
Android
MainActivity.java
activity_main.xml
Nexus 4
1: Project
7: Structure
Captures
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy Ctrl+C
Copy Path Ctrl+Shift+C
Copy as Plain Text
Paste Ctrl+V
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
FindBugs
Refactor
Generate Android Selectors
Add to Favorites
Show Image Thumbnails Ctrl+Shift+T
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Local History
Synchronize 'app'
Show in Explorer
File Path Ctrl+Alt+F12
Compare With... Ctrl+D
Open Module Settings F4
Create Gist...
Inspect Code...
Code Cleanup...
Run Inspection by Name... Ctrl+Alt+Shift+I
Configure Current File Analysis... Ctrl+Alt+Shift+H
Infer Nullity...
Show Coverage Data... Ctrl+Alt+F6
Analyze Dependencies...
Analyze Backward Dependencies...
Analyze Module Dependencies...
Analyze Cyclic Dependencies...
Analyze Data Flow to Here
Analyze Data Flow from Here
Analyze Stacktrace...
FindBugs
Messages Gradle Build

Customized Controls

为什么需要自定义 View

1. 现有的View满足不了你的需求,也没有办法从已有控件派生一个出来;界面元素需要自己绘制。

2. 现有 View 可以满足要求，把它做成自定义 View 只是为了抽象：为这个自定义 View 提供若干方法，方便调用着操纵 View。通常做法是派生一个已有 View，或者结合 xml 文件直接 inflate。

Attention：能够用 Android 基础控件解决的问题就尽量用基础控件，其次是用基础控件的组合。如果是确实有必要自定义才考虑自定义。自定义的控件，既需要耗费较长的开发时间，又不一定能保证有基础控件那么高的效率（基础控件都是谷歌优化过了的）。

应用场景

A. 组合控件：试题控件（TextView+VideoGroup）、下拉刷新、瀑布流控件、带左/右滑功能的控件、视频控件等。通过 Android 的基础控件(TextView、CheckBox、Button、ProgressBar 等)组合而成，

B. 完全自定义控件：继承自 View、TextureView 或 SurfaceView，然后重写核心的回调方法。
   比如：webview + loading 动画（SurfaceView）、
   比如输入法中的手写控件、图文混排控件（现在很多都是通过 webview 加载网页实现了）、词典取词控件、图表控件、个性化进度条、弹幕显示控件、Markdown 控件、IDE 代码编辑控件等。

系统学习 Android 自定义控件

如何系统学习自定义控件 https://zhuanlan.zhihu.com/p/21995633

[Android技术专题]自定义View： https://www.zhihu.com/question/41101031

自定义属性

Android提供了一些基本的控件实现，有时无法满足我们需求。

这时需要自定义控件，在Android基础控件上实现我们想要的功能

下面以RadioButton为例：

我们自定义控件RadioButton，加入自定义属性”key”

attrs.xml 中定义 key 属性

attrs.xml | user_linearlayout.xml | UserRadioButton.jav

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

    <declare-styleable name="RadioButton">
        <attr name="key" format="string" />
    </declare-styleable>

</resources>
```

XML 中引用自定义控件

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res/com.vivian.ui.testcase"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
      ……
        <RadioGroup
            android:id="@+id/radioGroup"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:contentDescription="@string/user_radiobtn"
            android:layout_gravity="center_horizontal">

            <com.user.utils.UserRadioButton
                android:id="@+id/userRadioBtn"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/user_radiobtn"
                app:key="user key" >
            </com.user.utils.UserRadioButton>

            <com.user.utils.UserRadioButton
                android:id="@+id/userRadioBtn2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/user_radiobtn"
                app:key="user key 2" >
            </com.user.utils.UserRadioButton>
        </RadioGroup>
```

创建 UserRadioButton 类继承 RadioButton

把我们自定义的属性跟UserRadioButton类中的变量绑定

```
public class UserRadioButton extends RadioButton {
......
	public UserRadioButton(Context context, AttributeSet attrs) {
		super(context, attrs);

		// bind with property values/attrs.xml
		try {
			TypedArray a = context.obtainStyledAttributes(attrs,
					R.styleable.RadioButton);
			this.mUserKey = a.getString(R.styleable.RadioButton_key);
			a.recycle();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}

	public String getKey() {
		return mUserKey;
	}

	public void setKey(String value) {
		mUserKey = value;
	}
```

Code 中修改自定义属性

```
mUserRadioButton = (UserRadioButton) findViewById(R.id.userRadioBtn);
mUserRadioButton.setKey("modified key");
```

优质开源库

awesome-android-ui : https://github.com/wasabeef/awesome-android-ui

android-open-project : https://github.com/Trinea/android-open-project

View 事件传递

基础知识

A. 所有 Touch 事件都被封装成了 MotionEvent 对象，包括 Touch 的位置、时间、历史记录以及第几个手指(多指触摸)等。

B. 事件类型分为

ACTION_DOWN, ACTION_UP, ACTION_MOVE,

ACTION_POINTER_DOWN, ACTION_POINTER_UP,

ACTION_CANCEL，

每个事件都是以 ACTION_DOWN 开始 ACTION_UP 结束。

C. 对事件的处理包括三类

传递——dispatchTouchEvent()函数

拦截——onInterceptTouchEvent()函数

消费——onTouchEvent()函数和 OnTouchListener

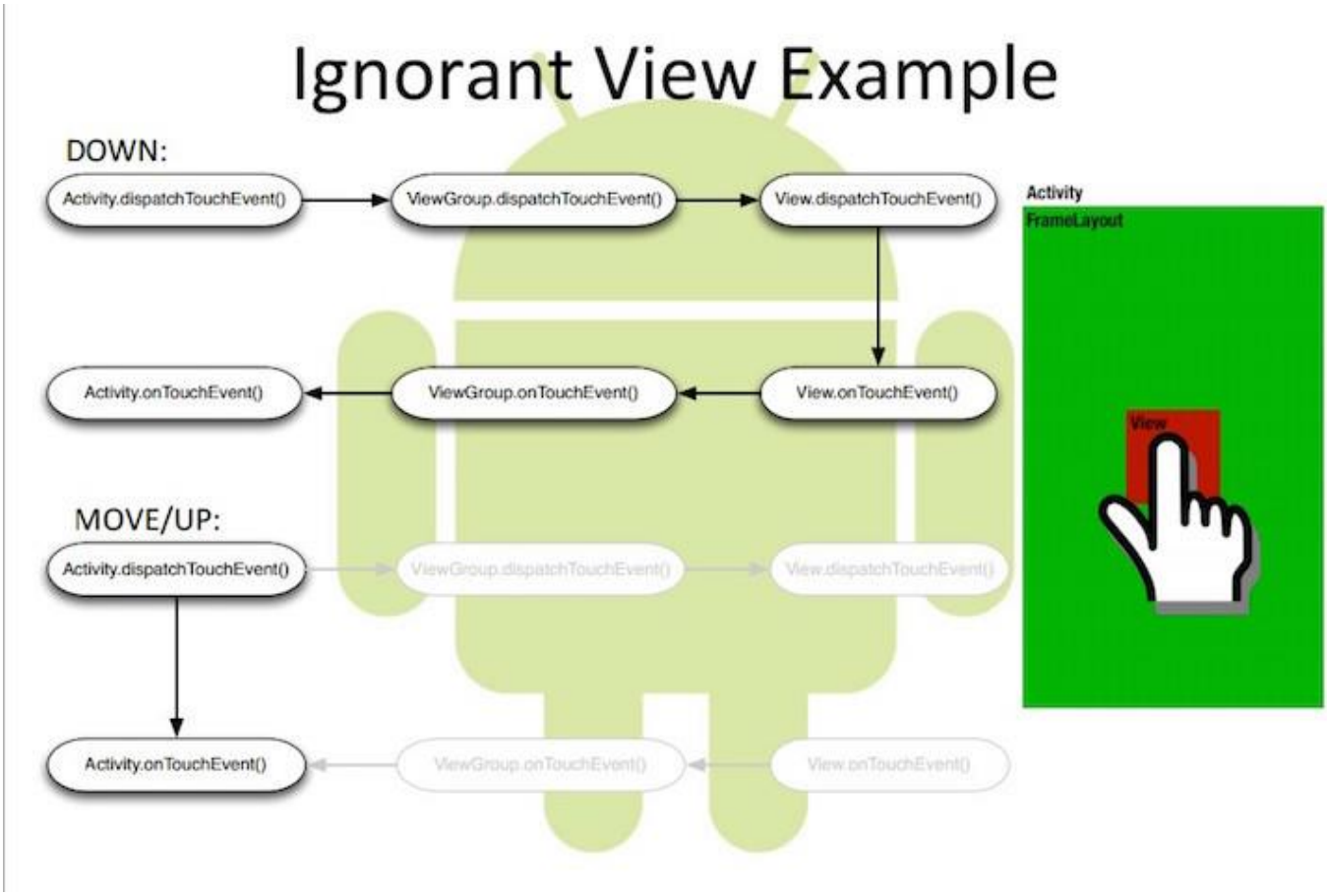
传递流程

1. 事件从 Activity.dispatchTouchEvent()开始传递。只要没有被停止或拦截，从最上层的 View(ViewGroup)开始一直往下(子 View)传递。子 View 可以通过 onTouchEvent()对事件进行处理。
2. 事件由父 View(ViewGroup)传递给子 View，ViewGroup 可以通过 onInterceptTouchEvent()对事件做拦截，停止其往下传递。

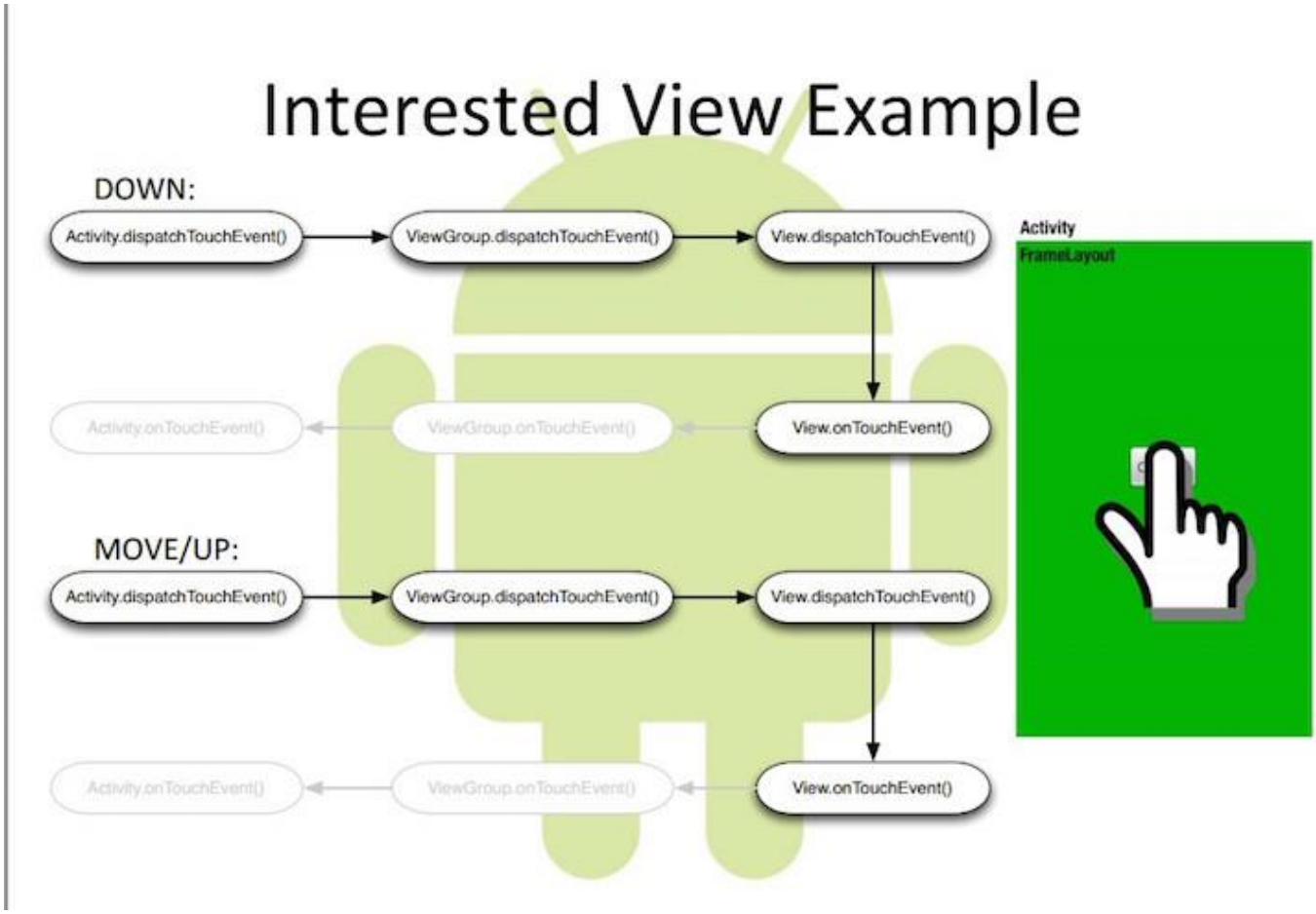
3. 如果事件从上往下传递过程中一直没有被停止，且最底层子 View 没有消费事件，事件会反向往上传递，这时父 View(ViewGroup)可以进行消费，如果还是没有被消费的话，最后会到 Activity 的 onTouchEvent()函数。
4. 如果 View 没有对 ACTION_DOWN 进行消费，之后的其他事件不会传递过来。
5. OnTouchListener 优先于 onTouchEvent()对事件进行消费。

上面的消费即表示相应函数返回值为 true。

View 不处理事件流程图



View 处理事件流程图



详见：https://github.com/android-cn/android-open-project-analysis 公共技术点之事件传递

UserInterface

Creating Menus

Notifying the User

Handling UI Events

Creating Menus

类型

Android系统里面有3种类型的菜单：options menu，context menu，sub menu。

1. Options Menu

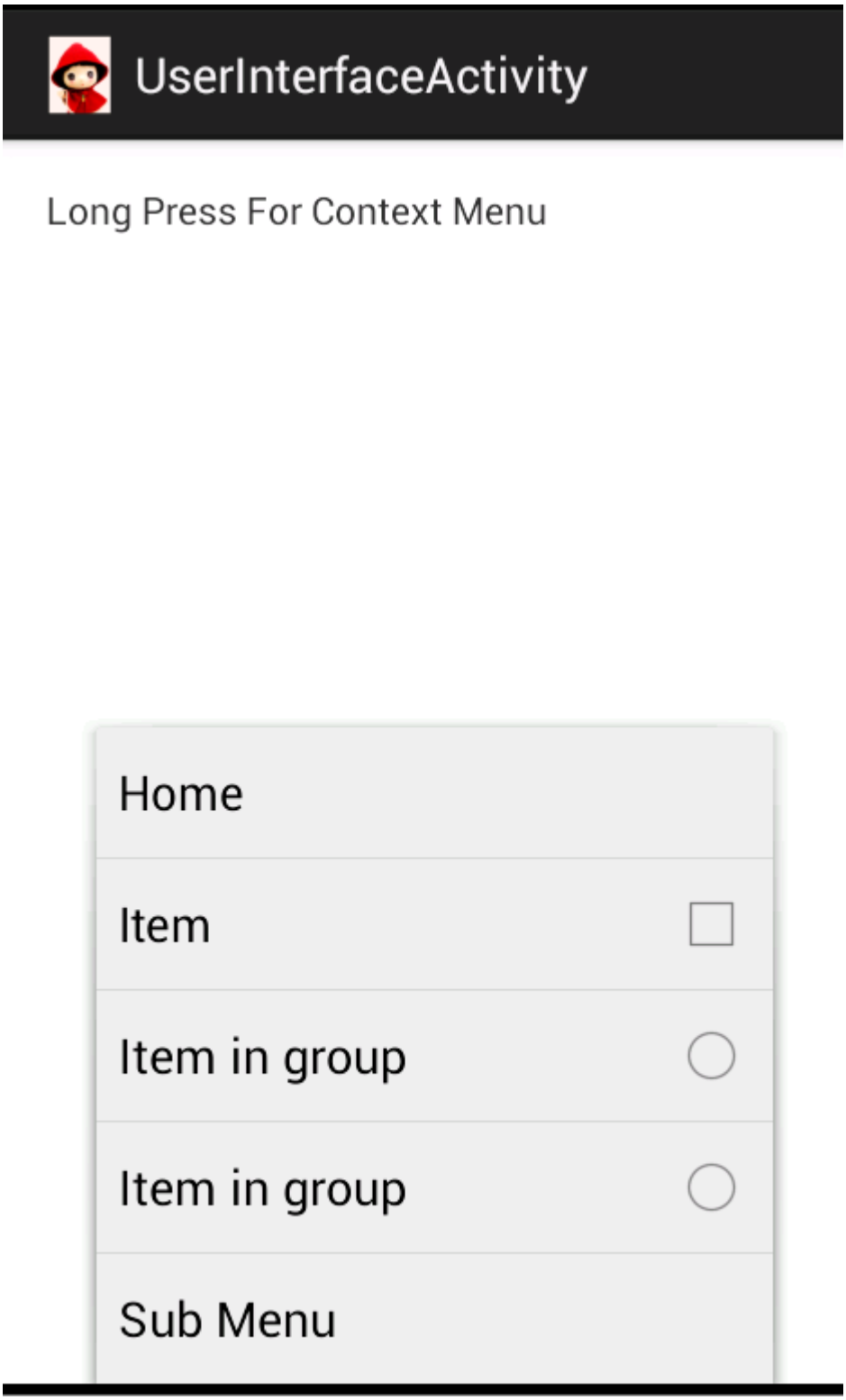
在Activity里面，一般通过以下函数来使用options menu：

Activity::onCreateOptionsMenu (Menu menu) 创建options menu，这个函数只会在menu第一次显示时调用。

Activity::onPrepareOptionsMenu (Menu menu) 更新改变options menu的内容，这个函数会在menu每次显示时调用。

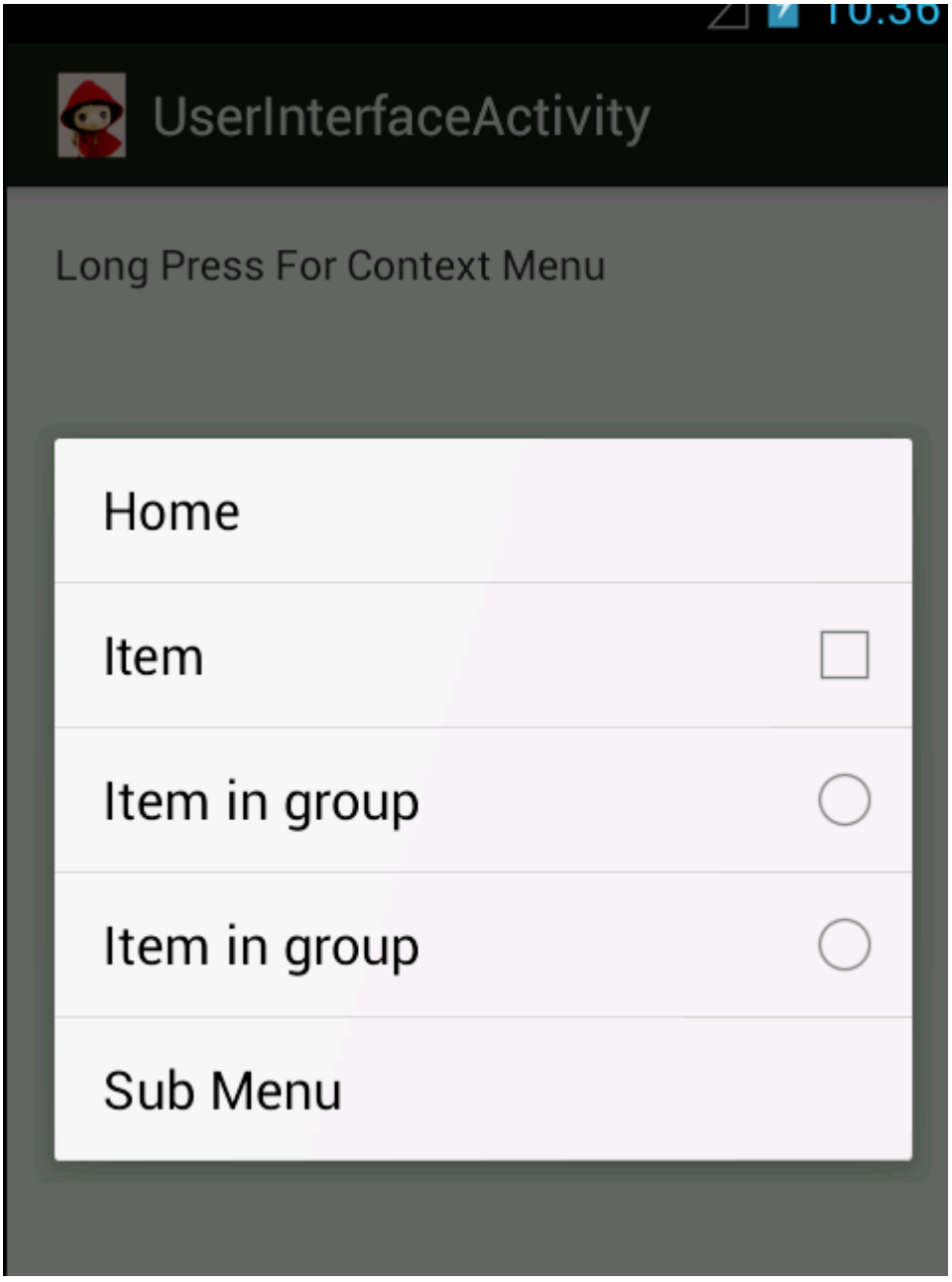
Activity::onOptionsItemSelected (MenuItem item) 处理选中的菜单项。

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.option_menu, menu);
    return true;
}

@Override
public boolean onMenuItemSelected(int featureId, MenuItem item) {
    // TODO Auto-generated method stub
    switch (item.getItemId()) {
    case R.id.action_home:
        goHome();
        break;

    default:
        Toast.makeText(this, item.getTitle() + " selected",
                Toast.LENGTH_LONG).show();
        break;
    }

    return super.onMenuItemSelected(featureId, item);
}
```



2. Context Menu

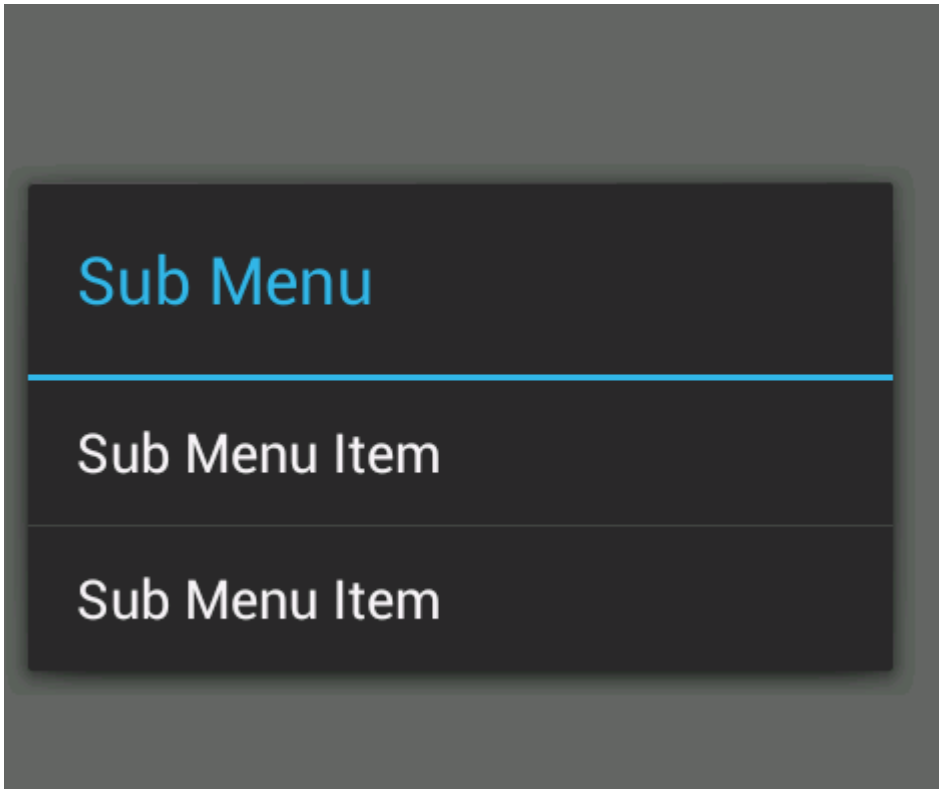
要在相应的view上按几秒后才显示的，用于view，跟某个具体的view绑定在一起。这类型的菜单不支持icon和快捷键



3. Submenu

以上两种menu都可以加入子菜单，但子菜单不能嵌套子菜单。同时子菜单不支持icon。

按下menu item后弹出的menu



xml 形式的 menu 定义及应用

接下来介绍相关的节点和属性(所有的属性都定义为android空间内，例如android:icon="@drawable/icon")：

<menu> 根节点，没有属性。

<group> 表示在它里面的<item>在同一group。相关属性包括：

id：group id

menuCategory：对应 常量Menu CATEGORY_* — 定义了一组的优先权，有效值：container，system，secondary，和alternative

orderInCategory：定义这组菜单在菜单中的默认次序，int值

checkableBehavior：这组菜单项是否checkable。有效值：none，all(单选/单选按钮radio button)，single(非单选/复选类型checkboxes)

visible：这组菜单是否可见 true or false

enabled：这组菜单是否可用，true or false

<item> 菜单项，可以嵌入<menu>作为子菜单。相关属性包括：

id： item id

menuCategory： 用来定义menu类别

orderInCategory： 用来定义次序，与一个组在一起(Used to define the order of the item, within a group)

title： 标题

titleCondensed：标题摘要， 当原标题太长的时候，需要用简短的字符串来代替title

icon： icon 图标

alphabeticShortcut： 字母快捷键

numericShortcut：数学快捷键

checkable：是否为checkbox， true or false

checked：是否设置为checked状态，true or false

visible： 是否可见， true or false

enabled：是否可用，true or false

Code 中使用

上述的三种类型的menu都能够定义为xml资源，但需要手动地使用MenuInflater来得到Menu对象的引用。

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.option_menu, menu);
    return true;
}

@Override
public boolean onMenuItemSelected(int featureId, MenuItem item) {
    // TODO Auto-generated method stub
    switch (item.getItemId()) {
    case R.id.action_home:
        goHome();
        break;

    default:
        Toast.makeText(this, item.getTitle() + " selected",
                Toast.LENGTH_LONG).show();
        break;
    }

    return super.onMenuItemSelected(featureId, item);
}

@Override
public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenuInfo menuInfo) {
    switch (v.getId()) {
    case R.id.menuTV:
        getMenuInflater().inflate(R.menu.option_menu, menu);

        break;

    default:
        break;
    }

    super.onCreateContextMenu(menu, v, menuInfo);
}
```

Notifying the User

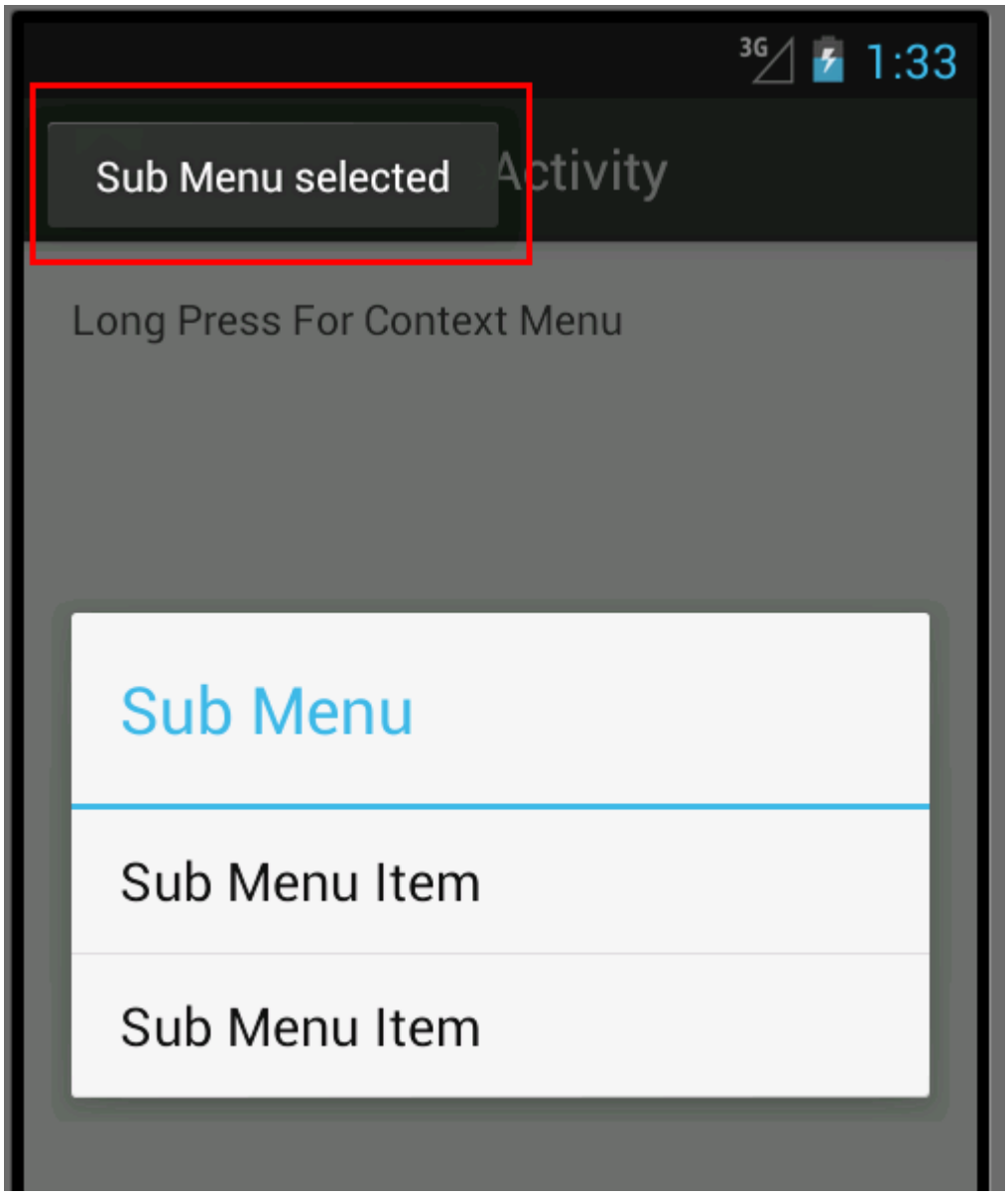
Creating Toast Notifications

Creating Status Bar Notifications

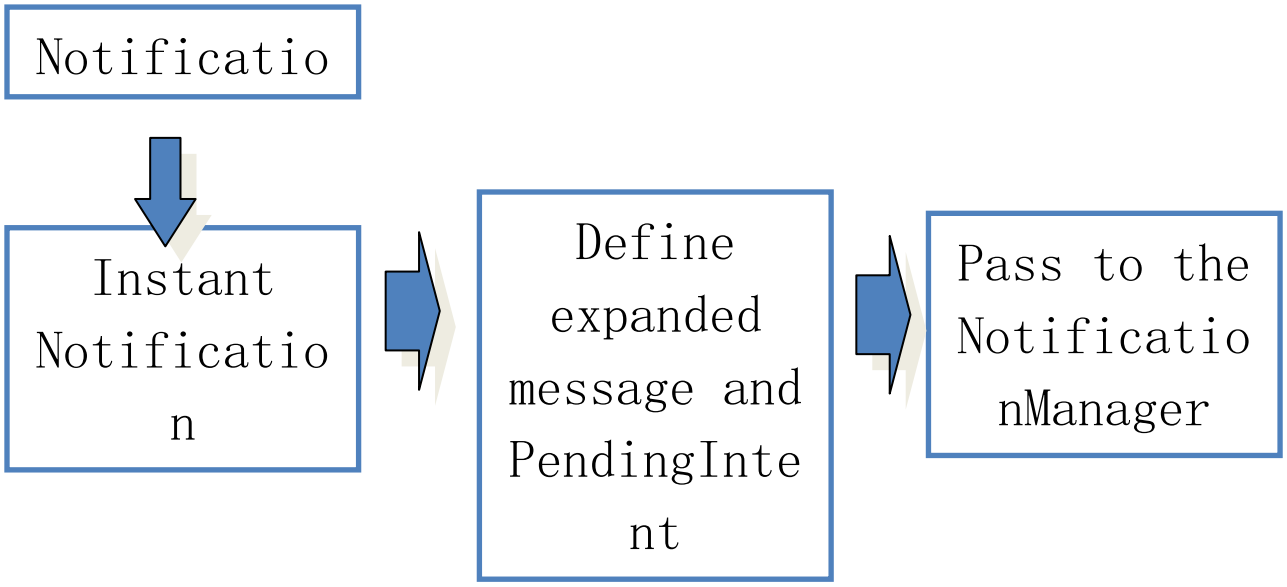
Creating Dialogs

Toast Notifications

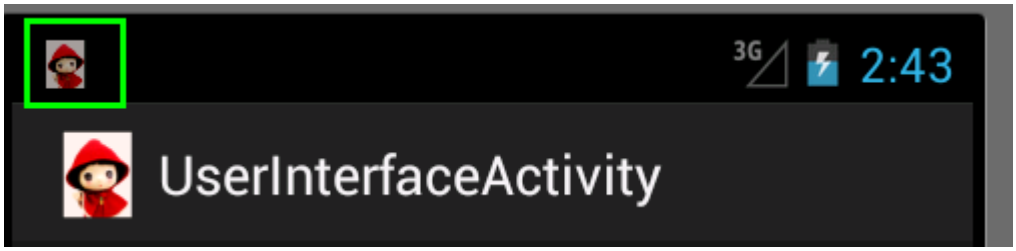
```
Toast ts = Toast.makeText(this, item.getTitle() + " selected",
        Toast.LENGTH_LONG);
ts.setGravity(Gravity.TOP | Gravity.LEFT, 0, 0);
ts.show();
```

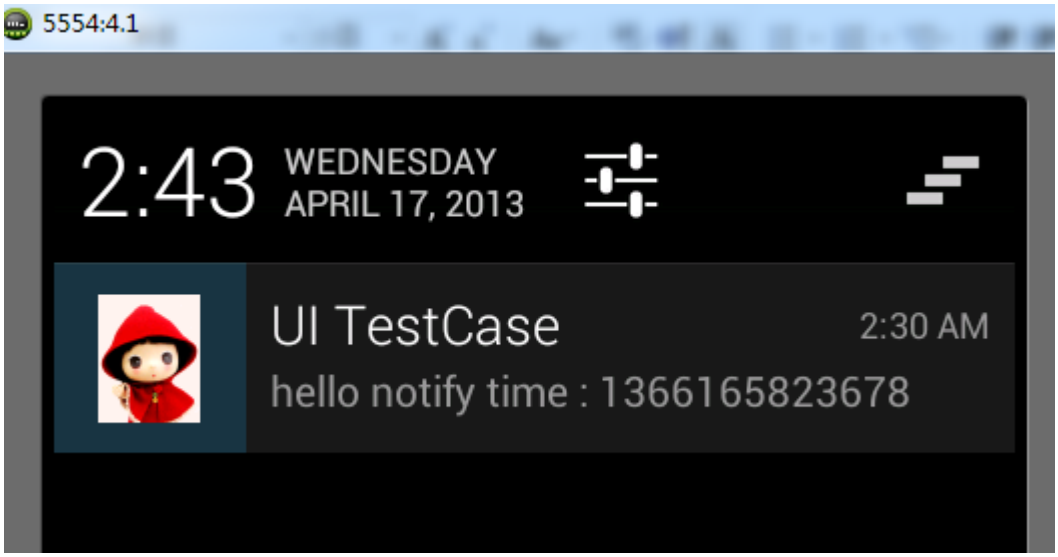


Status Bar Notifications




```
private void showStatusBarNotification(String notifyContent) {
    // get NotificationManager
    mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);

    // instant Notification
    Notification notification = new Notification();
    notification.icon = R.drawable.ic_launcher;
    notification.tickerText = getString(R.string.app_name);
    notification.when = System.currentTimeMillis();

    notification.flags |= Notification.FLAG_NO_CLEAR;
    notification.flags |= Notification.FLAG_SHOW_LIGHTS;
    notification.ledARGB = Color.BLUE;
    notification.ledOnMS =5000;


    // define expanded message and PendingIntent
    PendingIntent contentIntent = PendingIntent.getActivity(this, 0,
            new Intent(this, UserInterfaceActivity.class), PendingIntent.FLAG_CANCEL_CURRENT);
    notification.setLatestEventInfo(this, getString(R.string.app_name),
            notifyContent, contentIntent);

    // Pass to the NotificationManager
    mNotificationManager.notify(STATUSBAR_FIRST_ID, notification);
}
```



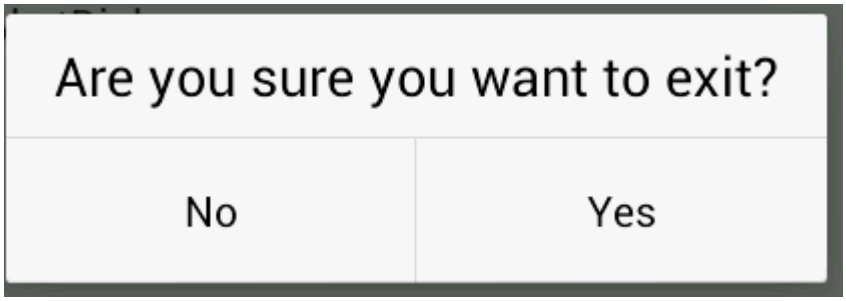


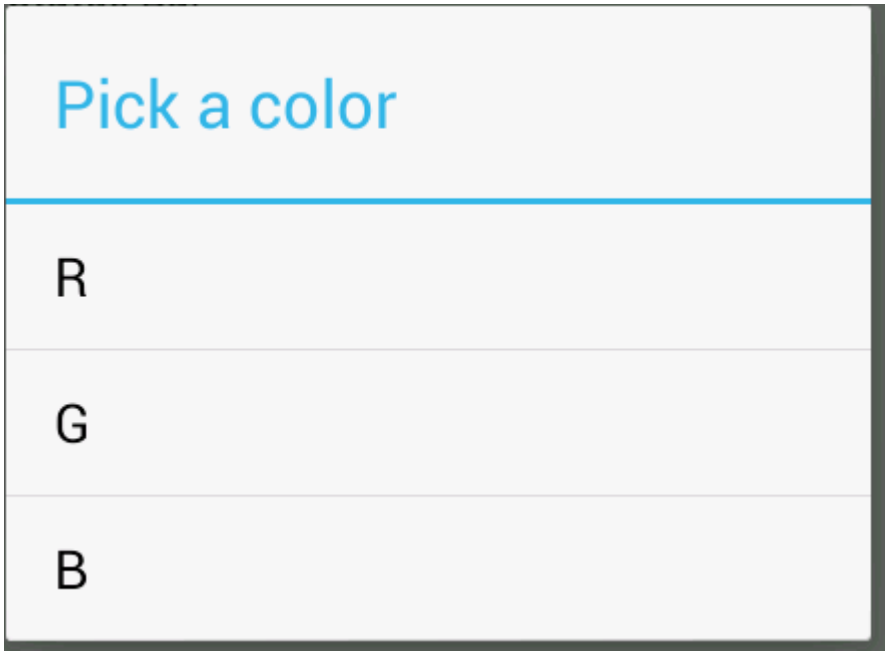
Dialogs

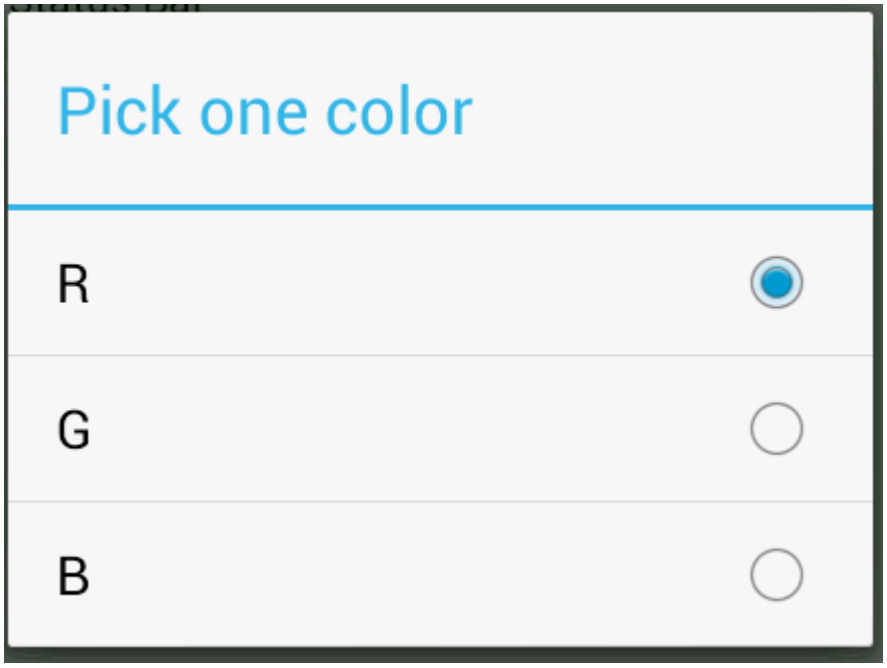
- AlertDialog
  - A dialog that can manage zero, one, two, or three buttons, and/or a list of selectable items that can include checkboxes or radio buttons.
- ProgressDialog
  - Because it's an extension of the AlertDialog, it also supports buttons.
- DatePickerDialog
- TimePickerDialog

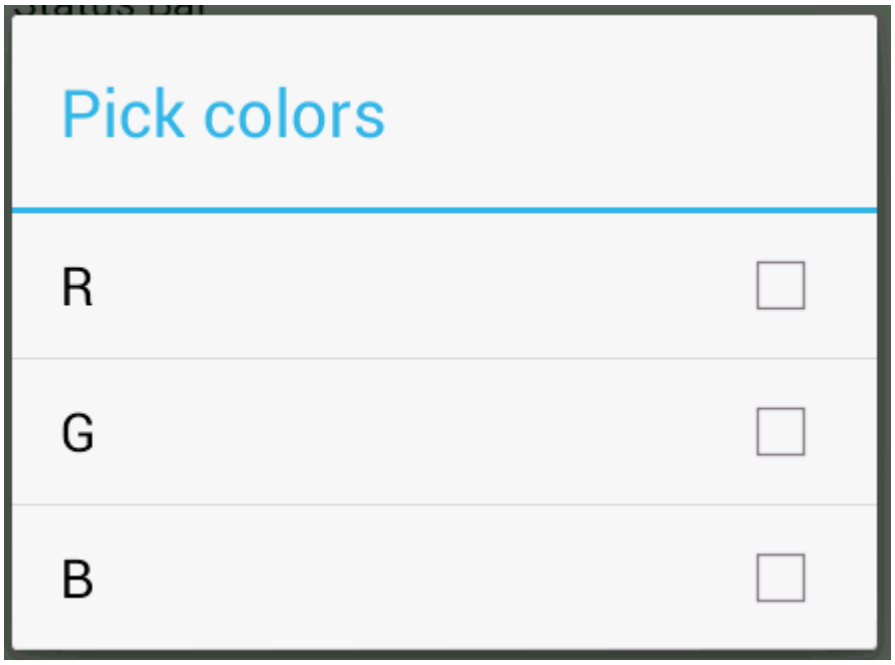
AlertDialogue

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setPositiveButton()
builder.setNeutralButton()
builder.setNegativeButton()

builder.setItems();

builder.setMultiChoiceItems()

builder.setSingleChoiceItems()
```



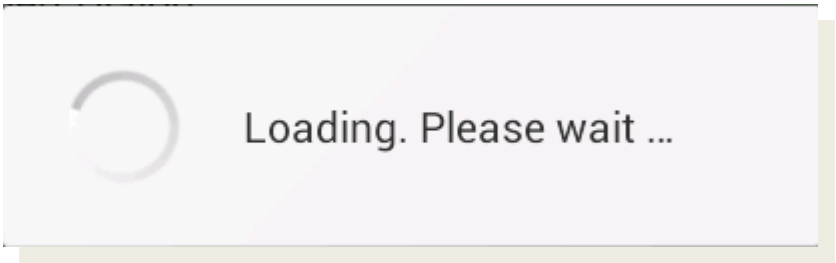


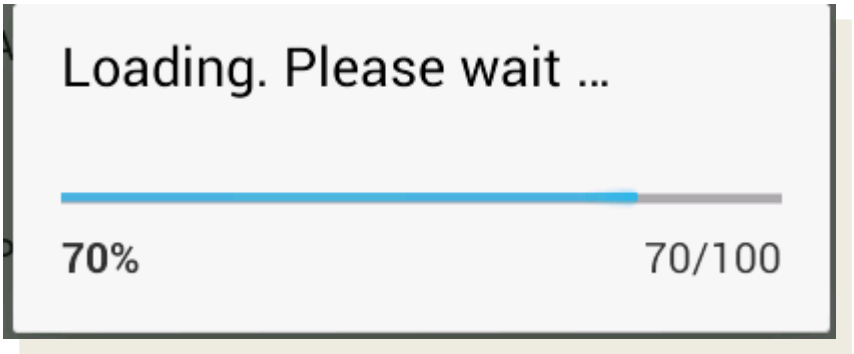




Progress dialogue

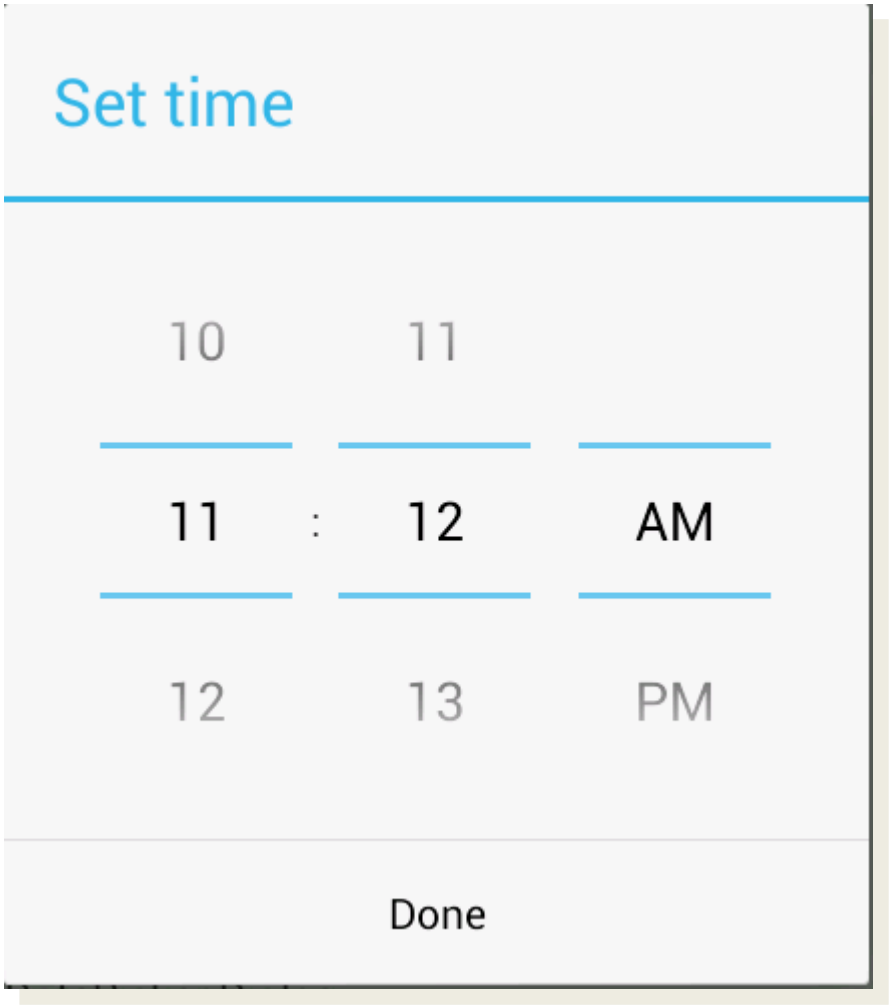
```
case R.id.progressDlgTV:
    mDialog = ProgressDialog.show(UserInterfaceActivity.this, "",
            getString(R.string.loading), true);
    startConsumeThread();
    break;
case R.id.progressDlg2TV:
    mDialog2 = new ProgressDialog(this);
    mDialog2.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
    mDialog2.setMessage(getString(R.string.loading));
    mDialog2.setCancelable(false);
    mDialog2.show();
    startConsumeThread2();
    break;
```

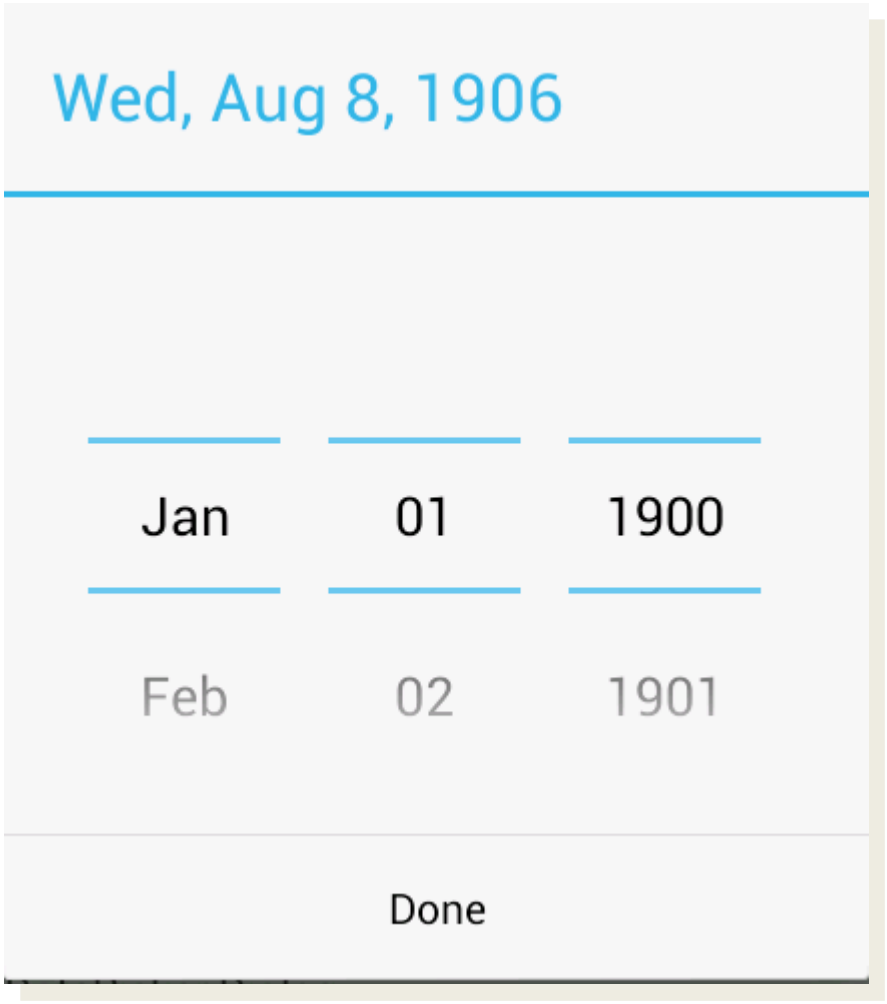




Pick Dialogue

```
case R.id.dataPickDlgTV:
    DatePickerDialog dialog = new DatePickerDialog(this,
            new OnDateSetListener() {
                @Override
                public void onDateSet(DatePicker view, int year,
                        int monthOfYear, int dayOfMonth) {
                    Toast.makeText(
                            mContext,
                            year + " 年 " + monthOfYear + " 月 "
                                    + dayOfMonth + " 日 ",
                            Toast.LENGTH_LONG).show();
                }
            }, Calendar.YEAR, Calendar.MONTH, Calendar.DAY_OF_MONTH);
    dialog.show();
    break;
case R.id.timePickDlgTV:
    TimePickerDialog tdialog = new TimePickerDialog(this,
            new OnTimeSetListener() {

                @Override
                public void onTimeSet(TimePicker view, int hourOfDay,
                        int minute) {
                    Toast.makeText(mContext,
                            hourOfDay + " 点 " + minute + " 分 ",
                            Toast.LENGTH_LONG).show();

                }
            }, Calendar.HOUR_OF_DAY, Calendar.MINUTE, false);
    tdialog.show();
```


Set time
10
11
11
:
12
AM
12
13
PM
Done


Wed, Aug 8, 1906
Jan
01
1900
Feb
02
1901
Done

Reference

[1] 如何系统学习自定义控件
https://zhuanlan.zhihu.com/p/21995633

[2] [Android技术专题]自定义View
https://www.zhihu.com/question/41101031

[3] 推翻自己和过往，重学自定义View

[4] 深入了解View

[5] Android 深入理解Android中的自定义属性

[6] 公共技术点之 View 事件传递

[7] Android 触摸及手势操作GestureDetector

[8] Android视图SurfaceView的实现原理分析

[9] 五、优质开源项目

- awesome-android-ui
- Android 开源项目分类汇总
- Android 平台下的原生 Markdown 解析器
- Android 开源弹幕引擎
- Android 图文混排控件 MixtureTextView