

## Andriod学习

### 前置

- 一、常用辅助网站
  - 1、RGB拾色器网站
  - 2、图标网站
  - 3、视频源码网址

### 第一章 文件内容

- 一、后端 (Java)
  - 1、主文件
- 二、前端 (页面)
  - 1、主布局
- 三、资源文件
  - 1、drawable
  - 2、mipmap
  - 3、values
- 四、配置文件
  - 1、AndroidManifest.xml
  - 2、build.gradle

### 第二章 布局

- 一、线性布局LinearLayout
  - 1、常用属性
    - 1.1 android:id
    - 1.2 android:layout\_width
    - 1.3 android:layout\_height
    - 1.4 android:background
    - 1.5 android:layout\_margin
    - 1.6 android:layout\_padding
    - 1.7 android:orientation
    - 1.8 android:gravity
    - 1.9 android:layout\_weight
  - 2、View
- 二、相对布局RelativeLayout
  - 1、最常用属性
    - 1.1 android:layout\_toLeftOf
    - 1.2 android:layout\_toRightOf
    - 1.3 android:layout\_alignBottom
    - 1.4 android:layout\_alignParentBottom
    - 1.5 android:layout\_below
- 三、文字布局TextView
  - 1、最常用属性
    - 1.1 文字大小、颜色
    - 1.2 显示不下使用...
    - 1.3 文字+icon
    - 1.4 中划线、下划线
    - 1.5 跑马灯
  - 2、Button
    - 2.1 最主要属性
    - 2.2 在activity文件中声明button
    - 2.3 在activity文件中找到button
    - 2.4 在activity文件中设置button点击事件
    - 2.5 新建一个activity
    - 2.6 跳转到新创建的activity
    - 2.7 添加背景，使button显示为圆角
    - 2.8 添加背景，使button显示描边
    - 2.9 添加背景，是button出现按压效果

- 2.9 设置点击事件
- 3、EditText
  - 3.1 主要属性
  - 3.2 用户名输入框
  - 3.3 密码输入框
  - 3.4 登录按钮
  - 3.5 点击事件
  - 3.6 监听输入文字改变事件
- 4、RadioButton
  - 4.1 实现button选择跳转
  - 4.2 实现单选
  - 4.3 更换样式
  - 4.4 监听事件
- 5、CheckBox
  - 5.1 实现多选
  - 5.2 自定义选择框
  - 5.3 监听事件
- 四、ImageView
  - 1、主要属性
  - 2、加载网络图片
- 五、ListView
  - 1、创建ListViewActivity
  - 2、添加展示页面
  - 3、创建Adapter类
  - 4、展示列表
  - 5、构造布局
  - 6、构造静态类保存布局控件
  - 7、getView函数中设置列表块内容
  - 8、ListViewActivity类中获取展示内容
  - 9、设置网格线
  - 10、点击事件和长点击事件
- 六、GridView
  - 1、主要属性
  - 2、设置展示内容
  - 3、点击事件和长点击事件
- 七、滚动视图ScrollView
  - 1、属性
  - 2、水平滚动
- 八、RecyclerView
  - 1、主要特点
    - 1.1 需要的库
  - 2、实现Adapter
  - 3、获取展示内容
  - 4、添加分割线
    - 4.1 添加dimen
    - 4.2 实现类
  - 5、监听点击事件
    - 5.1 可以通过回调方法实现与其他视图一样的监听方式
  - 6、水平展示
  - 7、网格展示
  - 8、瀑布流
  - 9、ViewHolder
- 九、WebView网络视图
  - 1、加载网页
    - 1.1 加载url
    - 1.2 加载html代码
    - 1.3 Native和JavaScript互相调用
  - 2、网页的前进后退

- 3、使用WebView
  - 3.1 加载本地html
  - 3.2 加载网络url

# Andriod学习

---

## 前置

---

### 一、常用辅助网站

#### 1、RGB拾色器网站

福步：<https://link.fobshanghai.com/rgbcolor.htm>

#### 2、图标网站

阿里巴巴免费的icon网站iconfont：<https://www.iconfont.cn/>

#### 3、视频源码网址

github：[https://github.com/taifus/Android\\_Learning](https://github.com/taifus/Android_Learning)

## 第一章 文件内容

---

### 一、后端（Java）

#### 1、主文件

路径：app/src/main/java/(具体的包)/MainActivity

重要生命周期方法onCreate通过setContentView方法声明了布局即Activity\_main

### 二、前端（页面）

#### 1、主布局

路径：app/src/main/res/layout/activity\_main.xml

### 三、资源文件

#### 1、drawable

路径：app/src/main/res/drawable

内容：图片和自定义的xml文件

#### 2、mipmap

路径：app/src/main/res/mipmap-hdpi、app/src/main/res/mipmap-mdpi、  
app/src/main/res/mipmap-xhdpi、app/src/main/res/mipmap-xxhdpi、  
app/src/main/res/mipmap-xxxhdpi

内容：logo

### 3、values

路径: app/src/main/res/values

内容: 颜色 (colors) 、文字 (strings) 、样式 (styles)

## 四、配置文件

### 1、AndroidManifest.xml

路径: app/src/main/AndroidManifest.xml

内容: 应用中所有用到的activitys都需要在这个文件中声明

将Mainactivity设置为启动的activity

```
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

### 2、build.gradle

路径: app/src/build.gradle

内容: android{…}

```
compileSdkVersion 编译sdk的版本
buildToolsVersion 编译工具的版本
defaultConfig{
    versionCode 版本号
    versionName 版本名称
}
dependencies{…}
    compile 'xxx' 使用到的包
}
```

## 第二章 布局

### 一、线性布局LinearLayout

#### 1、常用属性

## 2-1-1 LinearLayout

天哥在奔跑

最常用属性



## 1.1 android:id

线性控件的标志，用来寻找这个控件

@+id创建一个id

**android:id="@+id/l1\_1"**

## 1.2 android:layout\_width

线性控件的宽度，单位使用dp，不用px（适配不同机型）

wrap\_content 包含内容，内容有多少，宽度就为多少

match parent 匹配父控件，父控件有多少，宽度就为多少

## 1.3 android:layout\_height

线性控件的高度，单位使用dp（字体使用sp）不用px（适配不同机型）

wrap\_content 包含内容，内容有多少，宽度就为多少

match parent 匹配父控件，父控件有多少，宽度就为多少

## 1.4 android:background

线性控件的背景、颜色、图片和自定义的xml文件

#000000 黑色 //fobshanghai.com/rgbcolor.htm取色

## 1.5 android:layout\_margin

线性控件的外边距，单位使用dp（字体使用sp）不用px（适配不同机型）

## 1.6 android:layout padding

线性控件的内边距，单位使用dp（字体使用sp）不用px（适配不同机型）

```
    android:paddingLeft="20dp"
    android:paddingRight="20dp"
    android:paddingTop="50dp"
    android:paddingBottom="10dp">>
```

### **1.7 android:orientation**

线性控件的方向， 默认为水平

vertical 垂直方向

horizontal 水平方向

### **1.8 android:gravity**

对齐方式 (内部元素排列方式)

bottom 底部对齐

right 右部对齐

left 左部对齐

top 上部对齐

center 居中对齐

center\_horizontal 水平居中

center\_vertical 垂直居中

### **1.9 android:layout\_weight**

减掉父控件被占用的空间后剩余部分根据权重划分

## **2、View**

所有控件的父类

## **二、相对布局RelativeLayout**

### **1、最常用属性**



### **1.1 android:layout\_toLeftOf**

在某个控件的左边

### **1.2 android:layout\_toRightOf**

在某个控件的右边

### **1.3 android:layout\_alignBottom**

跟某个控件底部对齐

### **1.4 android:layout\_alignParentBottom**

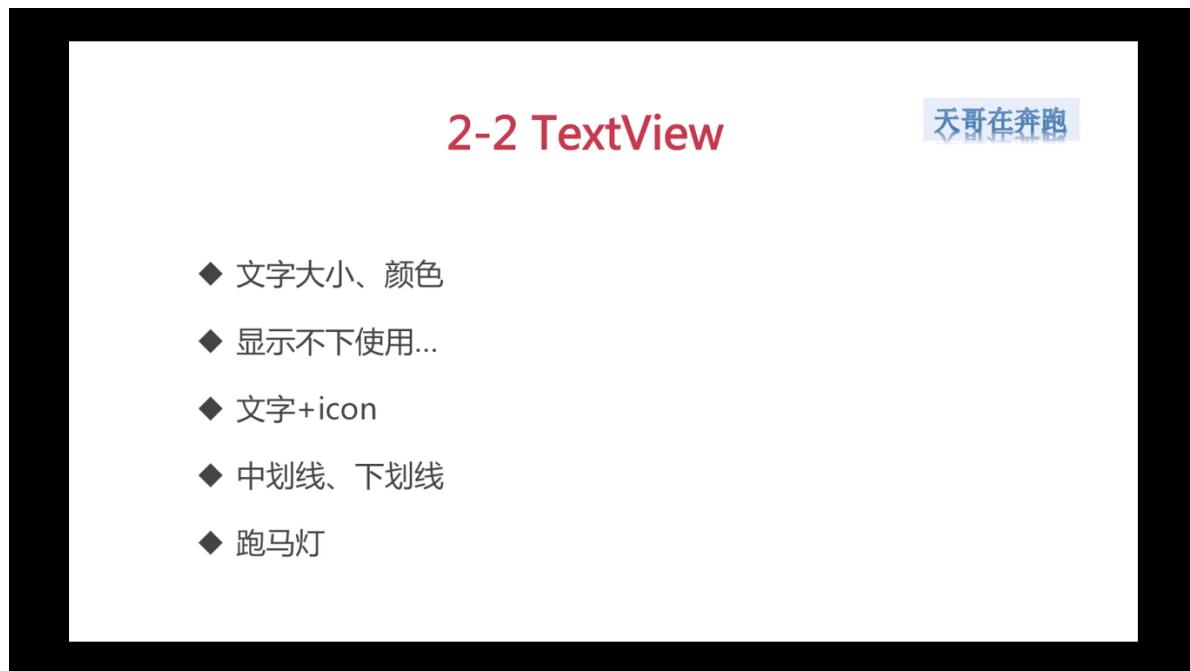
跟父控件底部对齐

### **1.5 android:layout\_below**

在某个控件的下边

## **三、文字布局TextView**

### **1、最常用属性**



#### **1.1 文字大小、颜色**

文字内容: android:text //可以引用字符串, 也可以直接写文字

文字显示为大写: android:textAllCaps="true" //默认为true

文字颜色: android:textColor

文字大小: android:textSize //字体大小使用sp

#### **1.2 显示不下使用...**

文字行数: android:maxLines

使用...: android:ellipsize="end"

### 1.3 文字+icon

图片放在drawable文件夹下



### 1.4 中划线、下划线

中划线

```
mTv4 = (TextView) findViewById(R.id.tv_4);
mTv4.getPaint().setFlags(Paint.STRIKE_THRU_TEXT_FLAG); //中划线 ![image]
mTv4.getPaint().setAntiAlias(true); //去除锯齿
```

20210103

下划线

```
mTv5 = (TextView) findViewById(R.id.tv_5);
mTv5.getPaint().setFlags(Paint.UNDERLINE_TEXT_FLAG); //下划线
```

//通过html设置下划线

```
mTv6 = (TextView) findViewById(R.id.tv_6);
mTv6.setText(Html.fromHtml("<u>天哥在奔跑</u>"));
```

### 1.5 跑马灯

文字行数: android:singleLine="true"

使用跑马灯: android:ellipsize="marquee"

跑马灯重复次数: android:marqueeRepeatLimit="marquee\_forever" //一直重复, 也可以写成-1

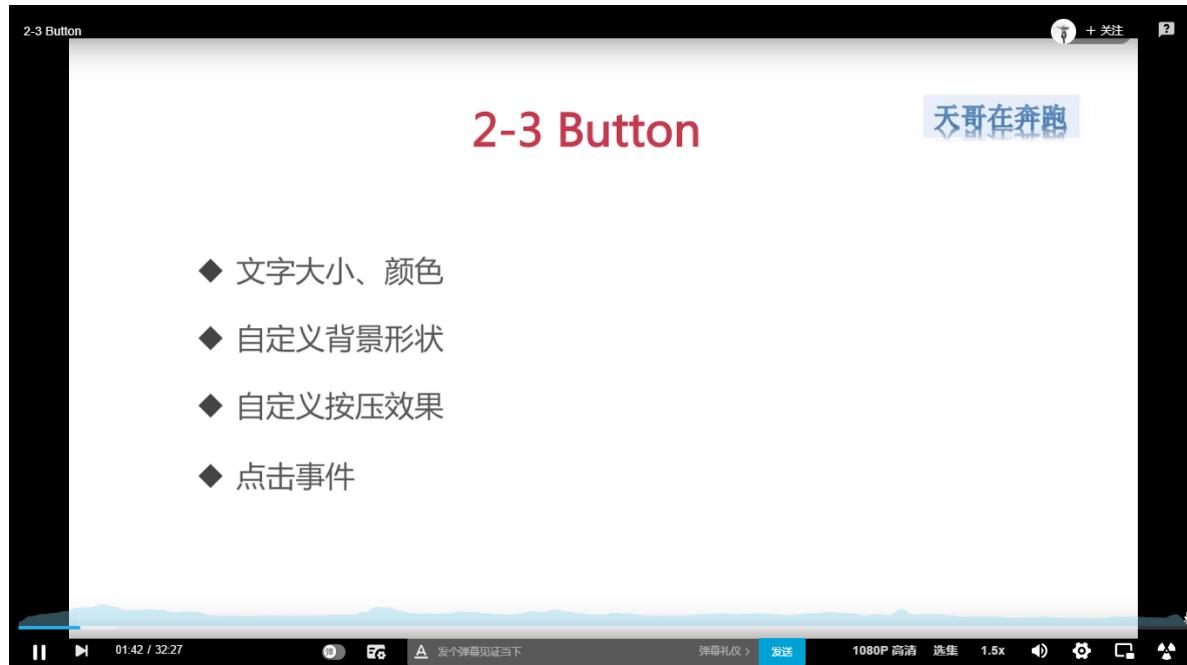
获取焦点: android:focusable="true"

焦点被触及: android:focusableInTouchMode="true"

## 2、Button

为TextView的子类

## 2.1 最主要属性



```
<Button  
    android:id="@+id	btn_textview"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="TextView"/>
```

## 2.2 在activity文件中声明button

```
? android.widget.Button? ↴  
private Button mBtnTextView;
```

## 2.3 在activity文件中找到button

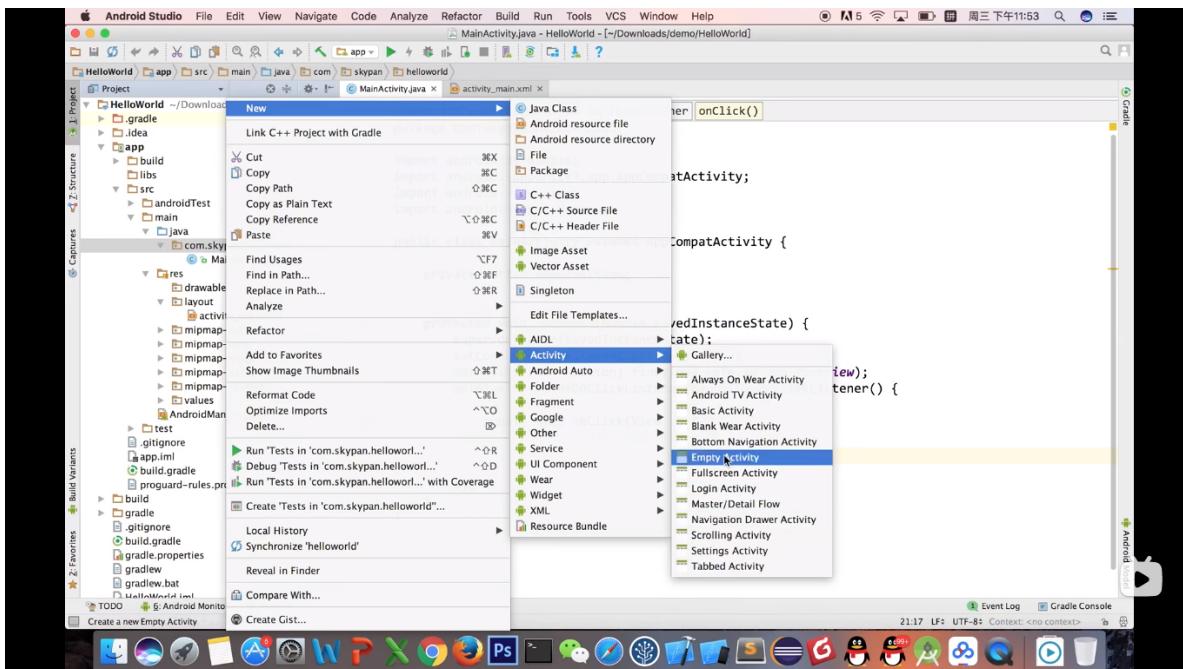
findViewById返回值为View，需要强制类型转换为子类Button

```
mBtnTextView = (Button) findViewById(R.id.btn_textview);
```

## 2.4 在activity文件中设置button点击事件

```
mBtnTextView.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        //跳转到TextView演示界面  
    }  
});
```

## 2.5 新建一个activity



在AndroidManifest.xml中声明activity, '.'代表包名

```
<activity android:name=".TextViewActivity"></activity>
```

## 2.6 跳转到新创建的activity

```
//跳转到TextView演示界面
Intent intent = new Intent(MainActivity.this, TextViewActivity.class);
startActivity(intent);
```

## 2.7 添加背景，使button显示为圆角

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid
        android:color="#FF9900"/>

    <corners
        android:radius="5dp"/>
</shape>
```

## 2.8 添加背景，使button显示描边

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <stroke
        android:width="1dp"
        android:color="#FF9900"/>

    <corners
        android:radius="5dp"/>
</shape>
```

## 2.9 添加背景，是button出现按压效果

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true">
        <shape>
            <solid android:color="#CC7A00"/>
            <corners android:radius="5dp"/>
        </shape>
    </item>
    <item android:state_pressed="false">
        <shape>
            <solid android:color="#FF9900"/>
            <corners android:radius="5dp"/>
        </shape>
    </item>
</selector>
```

## 2.9 设置点击事件

1、 android:onClick="showToast"

在activity中设置函数showToast

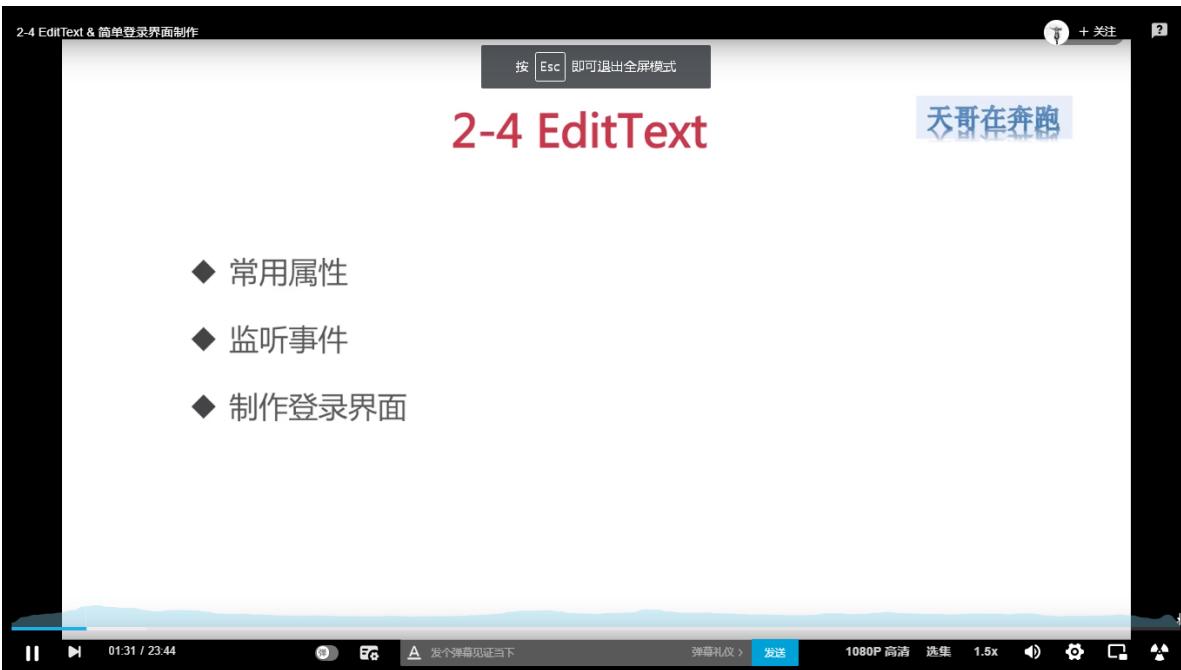
```
public void showToast(View view) {
    Toast.makeText(this,"我被点击了",Toast.LENGTH_SHORT).show();
}
```

2、直接在activity中设置点击事件

```
mBtn3 = (Button) findViewById(R.id.btn_3);
mBtn3.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(ButtonActivity.this,"btn3被点击了",Toast.LENGTH_SHORT).show();
    }
});
```

## 3、EditText

TextView的子类



- ◆ 常用属性
- ◆ 监听事件
- ◆ 制作登录界面

### 3.1 主要属性

提示: android:hint //默认提示

密码不显示: android:inputType="testPassword"

数字: android:inputType="number"

### 3.2 用户名输入框

```
<EditText  
    android:id="@+id/et_1"  
    android:layout_width="match_parent"  
    android:layout_height="50dp"  
    android:hint="用户名"  
    android:textColor="#FFAD33"  
    android:textSize="16sp"  
    android:background="@drawable/bg_username"  
    android:paddingLeft="10dp"  
    android:paddingRight="10dp"  
    android:drawablePadding="5dp"  
    android:maxLines="1"  
    android:drawableLeft="@drawable/icon_user"  
    android:layout_marginTop="50dp"/>/>
```

### 3.3 密码输入框

```
<EditText  
    android:id="@+id/et_2"  
    android:layout_width="match_parent"  
    android:layout_height="50dp"  
    android:layout_below="@id/et_1"  
    android:layout_marginTop="15dp"  
    android:hint="密码"  
    android:inputType="textPassword"  
    android:textColor="#FFAD33"  
    android:textSize="16sp"  
    android:background="@drawable/bg_username"  
    android:paddingLeft="10dp"  
    android:paddingRight="10dp"  
    android:maxLines="1"  
    android:drawablePadding="5dp"  
    android:drawableLeft="@drawable/icon_password"/>
```

### 3.4 登录按钮

```
<Button  
    android:id="@+id	btn_login"  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_below="@id/et_2"  
    android:layout_marginTop="40dp"  
    android:text="登录"  
    android:textColor="#fff"  
    android:textSize="20sp"  
    android:background="@drawable/bg_btn4"/>
```

### 3.5 点击事件

```
mBtnLogin = (Button) findViewById(R.id.btn_login);  
mBtnLogin.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        Toast.makeText(EditTextActivity.this, "登录成功!", Toast.LENGTH_SHORT).show();  
    }  
});
```

### 3.6 监听输入文字改变事件

addTextChangedListener函数的参数TextWacher包含三个方法，分别是文字改变前，文字改变时和文字改变后

```
mEtUserName = (EditText) findViewById(R.id.et_1);
mEtUserName.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
        Log.d("edittext",s.toString());
    }

    @Override
    public void afterTextChanged(Editable s) {

    }
});
```

## 4、RadioButton



### 4.1 实现button选择跳转

```
private class OnClick implements View.OnClickListener{
    @Override
    public void onClick(View v) {
        Intent intent = null;
        switch (v.getId()){
            case R.id.btn_textview:
                //跳转到TextView演示界面
                intent = new Intent(MainActivity.this, TextViewActivity.class);
                break;
            case R.id.btn_button:
                //跳转到Button演示界面
                intent = new Intent(MainActivity.this, ButtonActivity.class);
                break;
            case R.id.btn_edittext:
                //跳转到EditText演示界面
                intent = new Intent(MainActivity.this, EditTextActivity.class);
                break;
            case R.id.btn_radiobutton:
                //跳转到RadioButton演示界面
                intent = new Intent(MainActivity.this, RadioButtonActivity.class);
                break;
        }
        startActivity(intent);
    }
}
```

## 4.2 实现单选

在RadioGroup中放置RadioButton， 默认选中android:checked="true"， 设置checked属性必须给每个RadioButton一个id属性

```
<RadioGroup
    android:id="@+id/rg_1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton
        android:id="@+id/rb_1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="男"
        android:checked="true"
        android:textSize="18sp"
        android:textColor="#FF6600"/>
    <RadioButton
        android:id="@+id/rb_2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="女"
        android:textSize="18sp"
        android:textColor="#FF6600"/>
</RadioGroup>
```

#### 4.3 更换样式

去掉圆圈，使用`android:button="@null"`

更换背景，根据`state_checked`设置不同的样式，选中填充，未选中描边

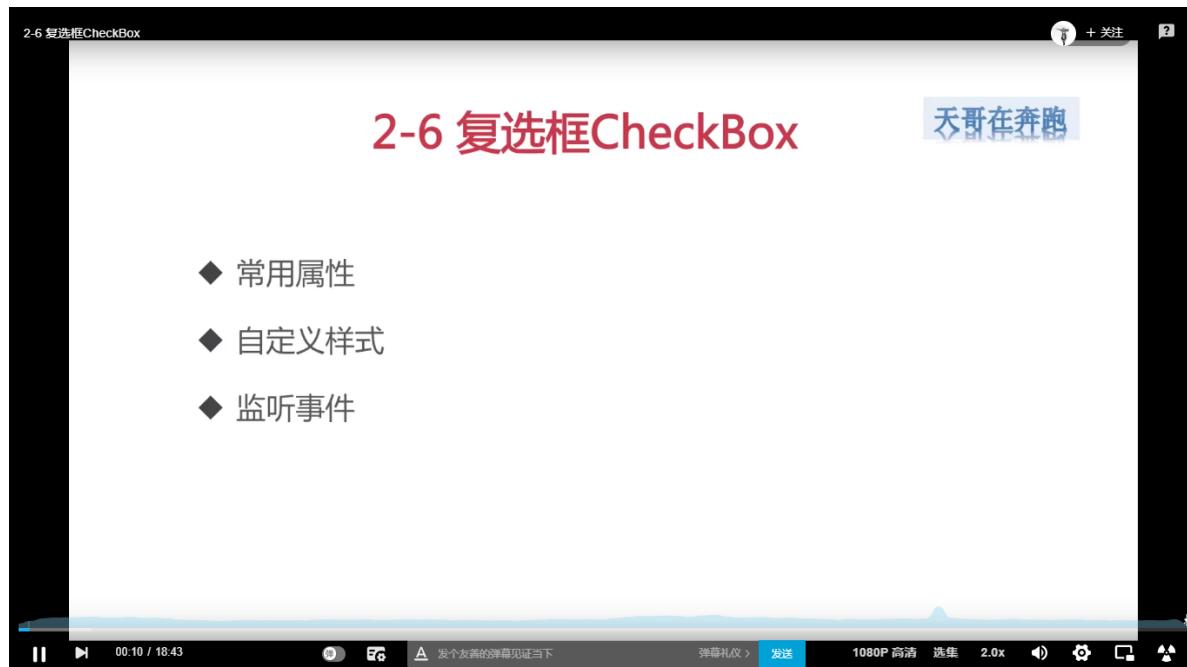
```
<selector xmlns:android="http://schemas.android.com/apk/r
    <item android:state_checked="true">
        <shape>
            <solid android:color="#AA6600"/>
            <corners android:radius="5dp"/>
        </shape>
    </item>
    <item android:state_checked="false">
        <shape>
            <solid android:color="#FF9900"/>
            <corners android:radius="5dp"/>
        </shape>
    </item>
</selector>
```

#### 4.4 监听事件

监听选中事件，在`onCheckedChanged`函数中编写选中后执行的内容

```
mRg1 = (RadioGroup) findViewById(R.id.rg_1);
mRg1.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup group, @IdRes int checkedId) {
        RadioButton radioButton = (RadioButton) group.findViewById(checkedId);
        Toast.makeText(RadioButtonActivity.this,radioButton.getText(),Toast.LENGTH_SHORT).show();
    }
});
```

### 5、CheckBox



## 5.1 实现多选

```
<CheckBox  
    android:id="@+id/cb_3"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="H5"  
    android:textSize="20sp"  
    android:layout_below="@id/cb_2"  
/>  
  
<CheckBox  
    android:id="@+id/cb_4"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="其他"  
    android:textSize="20sp"  
    android:layout_below="@id/cb_3"  
/>>
```

## 5.2 自定义选择框

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">  
    <item android:state_checked="false" android:drawable="@drawable/icon_checkbox_false"/>  
    <item android:state_checked="true" android:drawable="@drawable/icon_checkbox_true"/>  
</selector>  
  
<CheckBox  
    android:id="@+id/cb_5"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="编程"  
    android:button="@drawable/bg_checkbox"  
    android:drawableLeft="10dp"  
    android:textSize="20sp"  
    android:layout_marginTop="10dp"/>
```

## 5.3 监听事件

```
mCb5 = (CheckBox) findViewById(R.id.cb_5);  
mCb6 = (CheckBox) findViewById(R.id.cb_6);  
  
mCb5.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
    @Override  
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
        Toast.makeText(CheckBoxActivity.this,isChecked?"5选中":"5未选中",Toast.LENGTH_SHORT).show();  
    }  
});
```

# 四、 ImageView

## 2-7 ImageView

天哥在奔跑

- ◆ Button的其他衍生控件：ToggleButton、Switch
- ◆ 常用属性
- ◆ 加载网络图片

### 1、主要属性

src 图片属性

scaleType 缩放类型 fitXY撑满控件

- ◆ fitXY：撑满控件，宽高比可能发生改变
- ◆ fitCenter：保持宽高比缩放，直至能够完全显示
- ◆ centerCrop：保持宽高比缩放，直至完全覆盖控件，裁剪显示

### 2、加载网络图片

第三方库glide

Or use Gradle:

```
repositories {  
    mavenCentral() // jcenter() works as well because it pulls from Maven Central  
}  
  
dependencies {  
    compile 'com.github.bumptech.glide:glide:4.0.0'  
    compile 'com.android.support:support-v4:25.3.1'  
    annotationProcessor 'com.github.bumptech.glide:compiler:4.0.0'  
}
```

Or Maven:

```
repositories {
    mavenCentral() // jcenter() works as well because it pulls from Maven Central
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    testCompile 'junit:junit:4.12'
    compile 'com.github.bumptech.glide:glide:4.0.0'
    annotationProcessor 'com.github.bumptech.glide:compiler:4.0.0'
}
```

具体使用

```
mIv4 = (ImageView) findViewById(R.id.iv_4);
Glide.with(this).load("https://ss0.bdstatic.com/5aV1bjqh_Q23odCf/static/superman/img/logo/bd_logo1_31bdc765.png").into(mIv4);
```

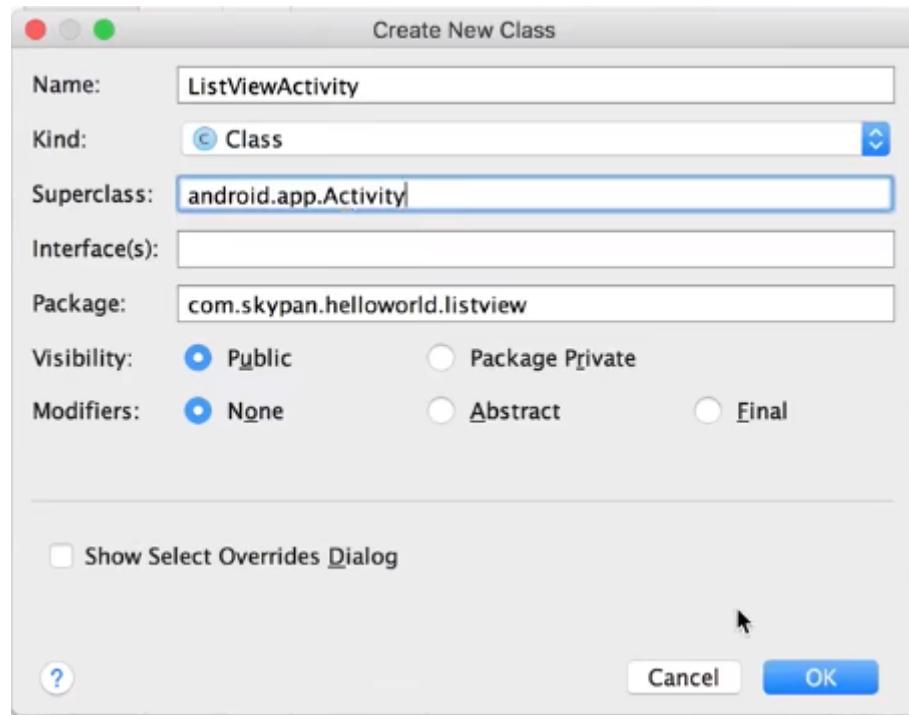
申请网络权限 AndroidManifest.xml

```
<uses-permission android:name="android.permission.INTERNET"/>
```

## 五、ListView

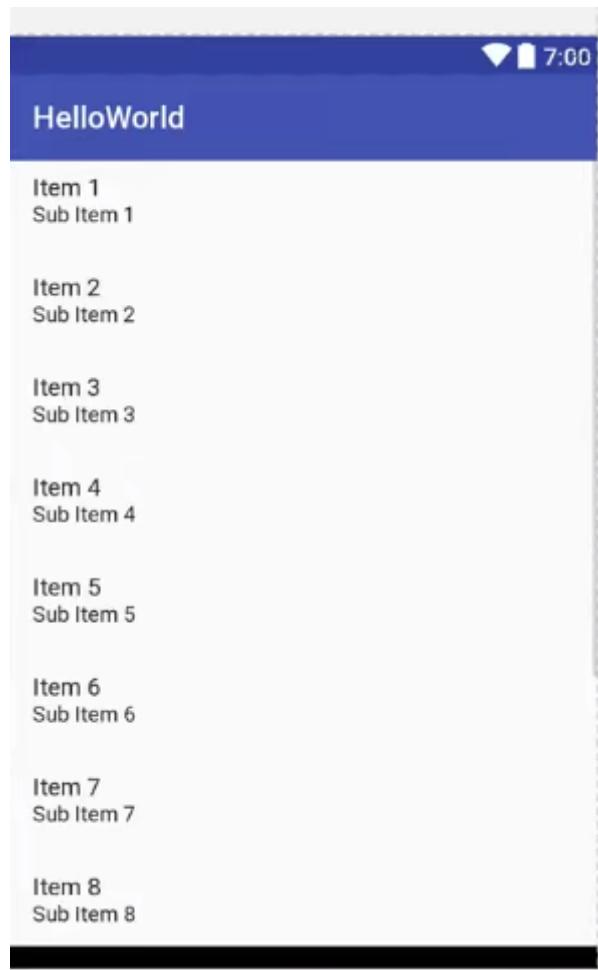


### 1、创建ListViewActivity

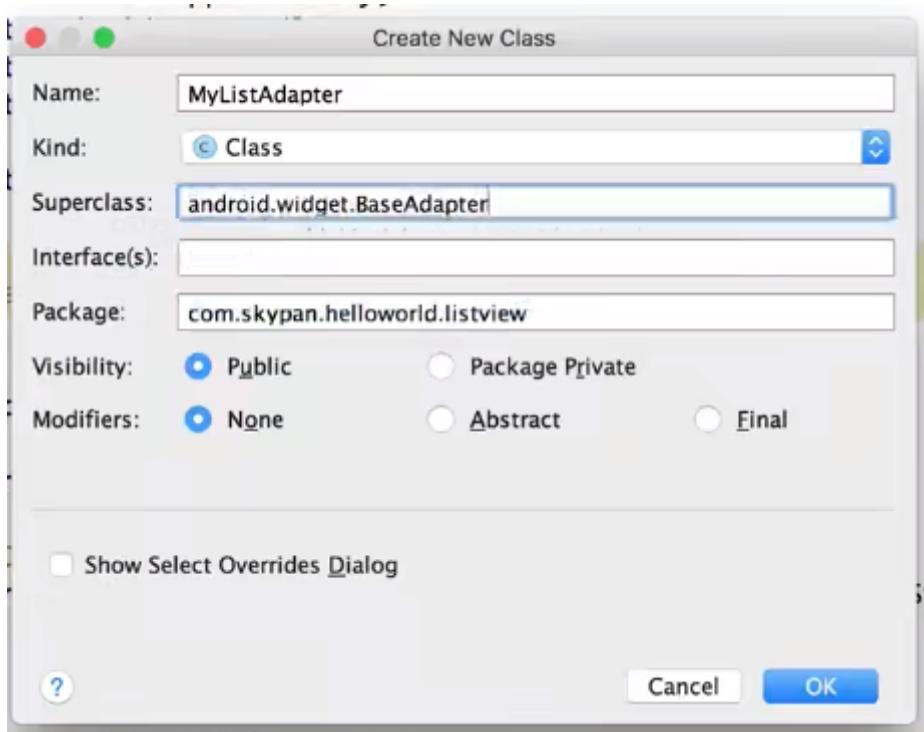


## 2、添加展示页面

```
<ListView  
    android:id="@+id/lv_1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"/>
```



### 3、创建Adapter类



### 4、展示列表

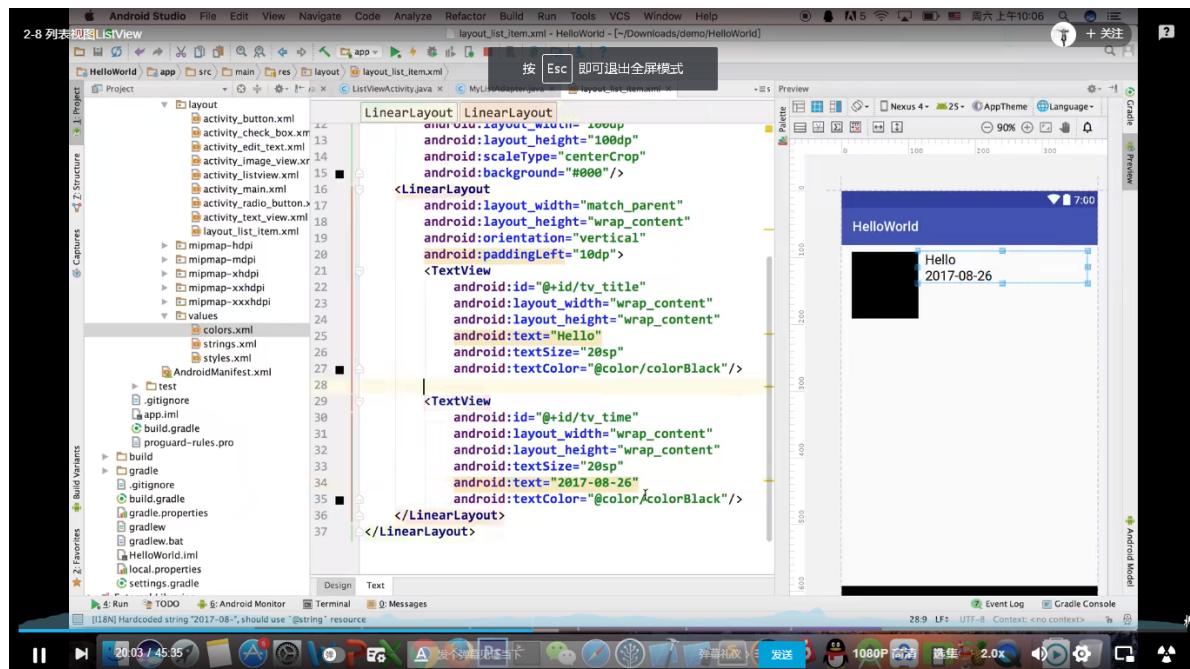
getCount返回值为列表的数量

```
private Context mContext;
private LayoutInflator mLayoutInflater;

public MyListAdapter(Context context){
    this.mContext = context;
    mLayoutInflater = LayoutInflater.from(context);
}

@Override
public int getCount() {
    return 10;
}
```

### 5、构造布局



## 6、构造静态类保存布局控件

```
static class ViewHolder{
    public ImageView imageView;
    public TextView tvTitle, tvTime, tvContent;
}
```

## 7、getView函数中设置列表块内容

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    ViewHolder holder = null;
    if(convertView == null){
        convertView = mLayoutInflater.inflate(R.layout.layout_list_item, null);
        holder = new ViewHolder();
        holder.imageView = (ImageView) convertView.findViewById(R.id.iv);
        holder.tvTitle = (TextView) convertView.findViewById(R.id.tv_title);
        holder.tvTime = (TextView) convertView.findViewById(R.id.tv_time);
        holder.tvContent = (TextView) convertView.findViewById(R.id.tv_content);
        convertView.setTag(holder);
    }else{
        holder = (ViewHolder) convertView.getTag();
    }
    //给控件赋值
    holder.tvTitle.setText("这是标题");
    holder.tvTime.setText("2088-08-08");
    holder.tvContent.setText("这是内容");
    Glide.with(mContext).load("").into(holder.imageView);
    return convertView;
}
```

## 8、ListViewActivity类中获取展示内容

```
mLv1 = (ListView) findViewById(R.id.lv_1);
mLv1.setAdapter(new MyListAdapter(ListViewActivity.this));
```

## 9、设置网格线

```
    android:layout_height="wrap_content"
    android:divider=""
    android:dividerHeight="1px"
    android:duplicateSelectionMode="Spanned" />
```

## 10、点击事件和长点击事件

```
mLv1.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Toast.makeText(ListViewActivity.this,"pos:"+position,Toast.LENGTH_SHORT).show();
    }
});

mLv1.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {
        return false;
    }
});
```

## 六、GridView



### 1、主要属性

**numColumns** 行数

**horizontalSpacing** 行之间的间距

**verticalSpacing** 列之间的间距

```
<GridView
    android:id="@+id/gv"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:numColumns="3"
    android:horizontalSpacing="10dp"
    android:verticalSpacing="10dp"/>
```

## 2. 设置展示内容

```
static class ViewHolder{
    public ImageView imageView;
    public TextView textView;
}

@Override
public View getView(int position, View convertView, ViewGroup parent) {
    ViewHolder holder = null;
    if(convertView == null){
        convertView = mLayoutInflater.inflate(R.layout.layout_grid_item,null);
        holder = new ViewHolder();
        holder.imageView = (ImageView) convertView.findViewById(R.id.iv_grid);
        holder.textView = (TextView) convertView.findViewById(R.id.tv_title);
        convertView.setTag(holder);
    }else{
        holder = (ViewHolder) convertView.getTag();
    }
    //赋值
    holder.textView.setText("花");
    Glide.with(mContext).load("http://").into(holder.imageView);
    return convertView;
}
```

## 3. 点击事件和长点击事件

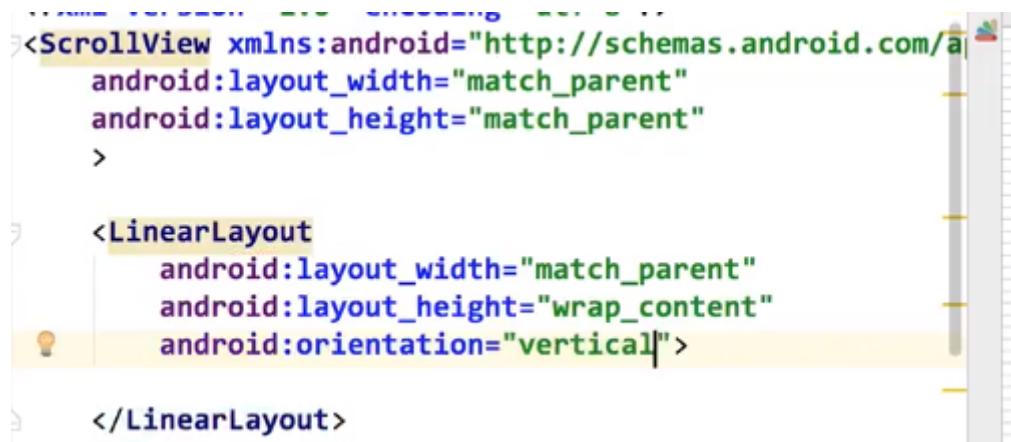
```
mGv = (GridView) findViewById(R.id.gv);
mGv.setAdapter(new MyGridViewAdapter.GridViewActivity.this));
mGv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Toast.makeText(GridViewActivity.this,"点击 pos:"+position,Toast.LENGTH_SHORT).show();
    }
});
mGv.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {
        Toast.makeText(GridViewActivity.this,"点击 pos:"+position,Toast.LENGTH_SHORT).show();
        return false;
    }
});
```

## 七、滚动视图ScrollView



## 1. 属性

子视图只能有一个，可以在需要滚动的视图外加一个ScrollView



## 2. 水平滚动

```
<HorizontalScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <Button
            android:layout_width="200dp"
            android:layout_height="300dp"
            android:text="Text"
            android:textAllCaps="false"/>
    </LinearLayout>
</HorizontalScrollView>
```

## 八、RecyclerView

### 1. 主要特点

The slide has a dark background with a white content area. At the top, the title "2-11-1 RecyclerView (一)" is displayed in red. To the right of the title is a small blue button labeled "秀哥在奔跑". Below the title, there is a large amount of white space. At the bottom of the slide, there is a footer bar with several icons: a left arrow, a pencil, a list icon, and a right arrow.

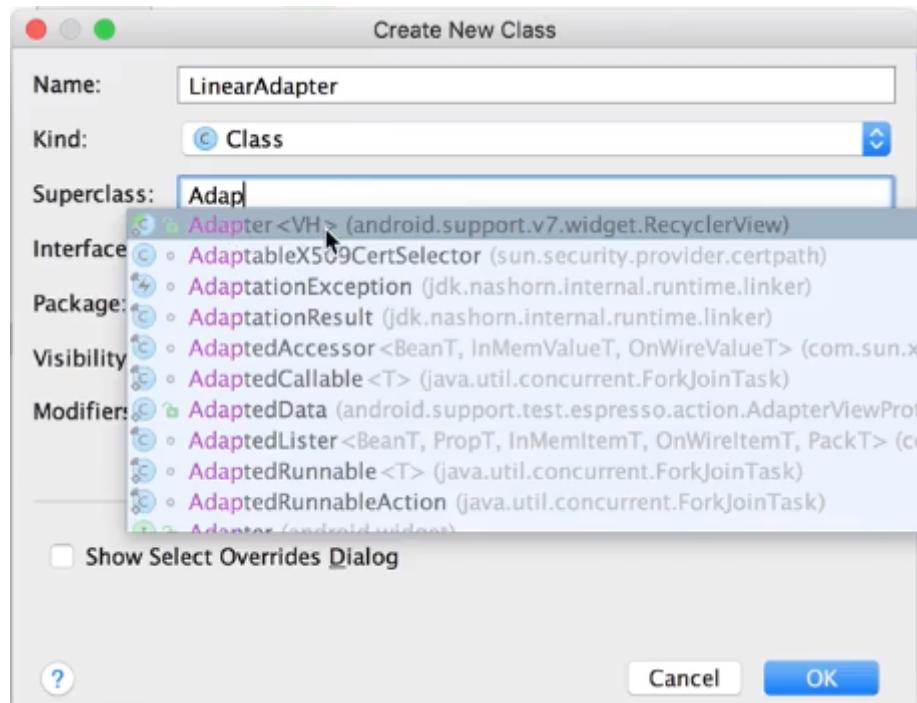
RecyclerView能够灵活实现大数据集的展示，视图的复用管理比ListView更好，能够显示列表、网格、瀑布流等形式，且不同的ViewHolder能够实现item多元化的功能。

但是使用起来会稍微麻烦一点，并且没有类似ListView的onItemClickListener监听事件，需要开发者自己实现。

### 1.1 需要的库

```
compile 'com.android.support:design:25.3.1'
```

## 2、实现Adapter



```
LinearAdapter onCreateViewHolder()
/* Created by skypan on 2017/8/31.
*/
public class LinearAdapter extends RecyclerView.Adapter<LinearAdapter.LinearViewHolder> {
    @Override
    public LinearAdapter.LinearViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        return null;
    }

    @Override
    public void onBindViewHolder(LinearAdapter.LinearViewHolder holder, int position) {
    }

    @Override
    public int getItemCount() {
        return 0;
    }

    class LinearViewHolder extends RecyclerView.ViewHolder {
        public LinearViewHolder(View itemView) {
            super(itemView);
        }
    }
}
```

```
private Context mContext;

public LinearAdapter(Context context){
    this.mContext = context;
}

@Override
public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    return new LinearViewHolder(LayoutInflater.from(mContext).inflate(R.layout.layout_linear_item, parent, false));
}
```

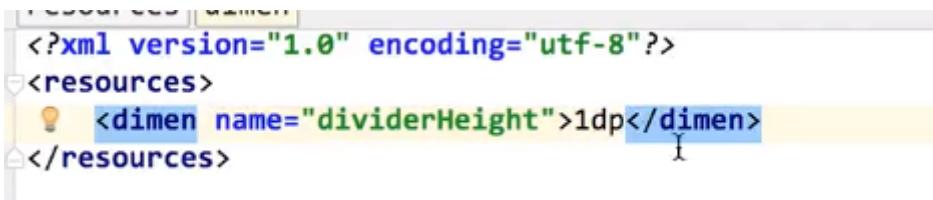
### 3、获取展示内容

```
mRvMain = (RecyclerView) findViewById(R.id.rv_main);
mRvMain.setLayoutManager(new LinearLayoutManager(LinearRecyclerViewActivity.this));
mRvMain.setAdapter(new LinearAdapter(LinearRecyclerViewActivity.this));
```

### 4、添加分割线

```
mRvMain.addItemDecoration(new MyDecoration());
```

#### 4.1 添加dimen



#### 4.2 实现类

```
class MyDecoration extends RecyclerView.ItemDecoration{
    @Override
    public void getItemOffsets(Rect outRect, View view, RecyclerView parent, RecyclerView.State state) {
        super.getItemOffsets(outRect, view, parent, state);
        outRect.set(0,0,0,getResources().getDimensionPixelOffset(R.dimen.dividerHeight));
    }
}
```

### 5、监听点击事件

```
}
@Override
public void onBindViewHolder(LinearAdapter.LinearViewHolder holder, final int position) {
    holder.textView.setText("Hello World!");
    holder.itemView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText(mContext,"click..."+position,Toast.LENGTH_SHORT).show();
        }
    });
}
```

#### 5.1 可以通过回调方法实现与其他视图一样的监听方式

```
public interface OnItemClickListener{
    void onClick(int pos);
}
```

```

@Override
public void onBindViewHolder(LinearLayoutAdapter.LinearLayoutHolder holder, final int position) {
    holder.textView.setText("Hello World!");
    holder.itemView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            mListener.onClick(position);
        }
    });
}

private Context mContext;
private OnItemClickListener mListener;

public LinearLayoutAdapter(Context context,OnItemClickListener listener){
    this.mContext = context;
    this.mListener = listener;
}

mRvMain.setAdapter(new LinearLayoutAdapter(LinearRecyclerViewActivity.this, new LinearLayout.OnItemClic
@Override
public void onClick(int pos) {
    Toast.makeText(LinearRecyclerViewActivity.this,"click "+pos,Toast.LENGTH_SHORT).show();
}
));

```

## 6、水平展示

```

    linearLayoutManager = new LinearLayoutManager(this);
    linearLayoutManager.setOrientation(LinearLayoutManager.HORIZONTAL);
    mRvMain.setLayoutManager(linearLayoutManager);

```

## 7、网格展示

gridLayoutManager的第二个参数为分成几列

```

mRvGrid = (RecyclerView) findViewById(R.id.rv_grid);
mRvGrid.setLayoutManager(new GridLayoutManager(GridRecyclerViewActivity.this,3));
mRvGrid.setAdapter(new GridAdapter(GridRecyclerViewActivity.this, new GridAdapter.OnItemClickListener{
    @Override
    public void onClick(int pos) {
    }
}));

```

## 8、瀑布流

StaggeredGridLayoutManager的第一个参数为行数或列数

```

mRvPu = (RecyclerView) findViewById(R.id.rv_pu);
mRvPu.setLayoutManager(new StaggeredGridLayoutManager(2,StaggeredGridLayoutManager.VERTICAL));
mRvPu.set

```

加载图片

```

holder.imageView.setImageResource(R.drawable.image1);
holder.itemView.setOnClickListener(new View.OnClickListener()

```

## 9、ViewHolder

## 不同的ViewHolder

XRecyclerView : addHeadView、addFootView、下拉刷新、上拉加载

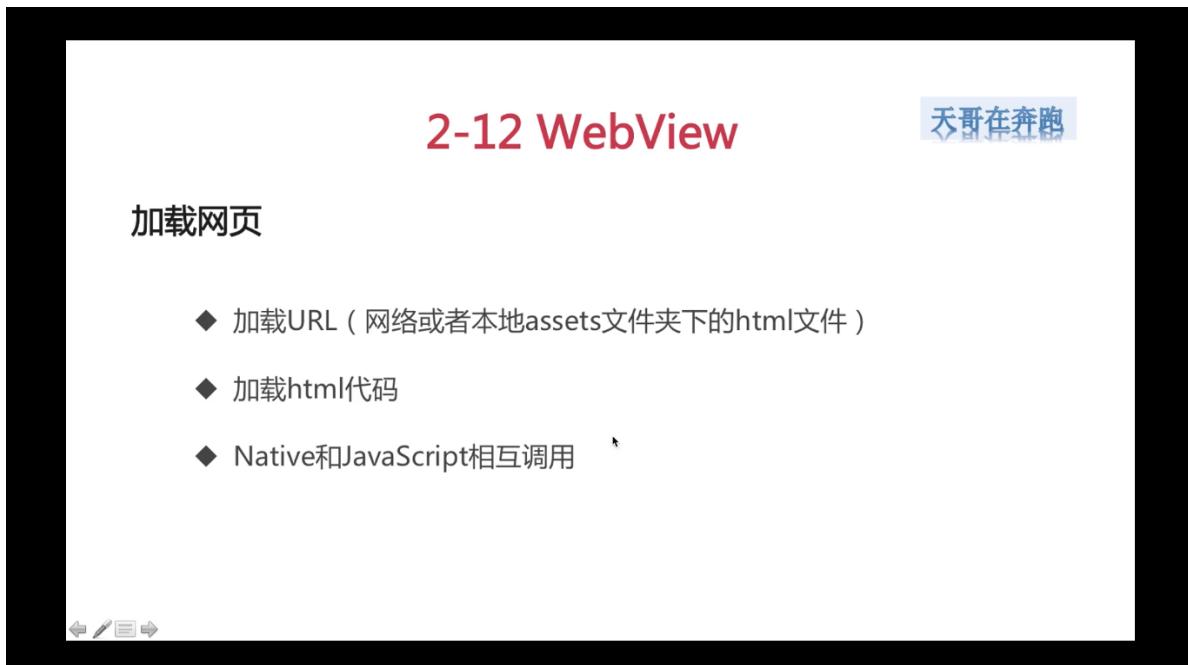
### 重写onCreateViewHolder方法

返回类型更改为所有视图类型的父类

```
@Override  
public LinearAdapter.LinearViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {  
    if(viewType == 0){  
        return new LinearViewHolder(LayoutInflater.from(mContext).inflate(R.layout.layout_linear_item,parent,false));  
    }else{  
        return new LinearViewHolder2(LayoutInflater.from(mContext).inflate(R.layout.layout_linear_item,parent,false));  
    }  
}  
  
@Override
```

## 九、WebView网络视图

### 1、加载网页



#### 1.1 加载url

### 加载网络URL

- ◆ webview.loadUrl( "http://www.tiantiantech.cn" );

### 加载assets下的html文件

- ◆ webview.loadUrl( "file:///android\_asset/test.html" );

## 1.2 加载html代码

### 加载html代码

- ◆ webview.loadData();

## 1.3 Native和JavaScript互相调用

## 2、网页的前进后退

### 网页的前进后退

- ◆ webview.canGoBack()
- ◆ webview.goBack()
- ◆ webview.canGoForward()
- ◆ webview.goForward()
- ◆ webview.canGoBackOrForward(int steps)
- ◆ webview.goBackOrForward(int steps)

按下返回键，默认是退出当前Activity，如果希望是WebView内页面后退

```
@Override  
public boolean onKeyDown(int keyCode, KeyEvent event) {  
    if ((keyCode == KeyEvent.KEYCODE_BACK) && webView.canGoBack()) {  
        webView.goBack();  
        return true;  
    }  
    return super.onKeyDown(keyCode, event);  
}
```

## 3、使用WebView

```
<WebView  
    android:id="@+id/wv"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"/>
```

### 3.1 加载本地html

```
    setContentView(R.layout.activity_main);
    mWvMain = (WebView) findViewById(R.id.wv);
    mWvMain.loadUrl("file:///android_asset/test.html");
}
```

### 3.2 加载网络url

```
//加载网络URL
mWvMain.getSettings().setJavaScriptEnabled(true);
mWvMain.loadUrl("https://m.baidu.com");
```