1. USTB 教务管理系统

1.1 xml 文件

1.1.1 AndroidMainFest.xml

涉及三种权限

```
<!--网络连接权限-->
<uses-permission android:name="android.permission.INTERNET" />
<!--网络状态权限-->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<!--手机振动权限-->
<uses-permission android:name="android.permission.VIBRATE" />
```

自定义 Activity 风格

```
<activity
    android:name=".activity.Exit_settings"
    android:theme="@style/MyDialogStyleBottom" />
```

1.1.2 styles.xml

```
<style name="MyDialogStyleBottom" parent="Base.Theme.AppCompat.Dialog">
    <item name="android:windowAnimationStyle">@style/AnimBottom</item>
    <item name="android:windowFrame">@null</item><!--边框-->
    <item name="android:windowIsFloating">true</item><!--是否浮现在 activity 之上-->
    <item name="android:windowIsTranslucent">true</item><!--半透明-->
    <item name="windowNoTitle">true</item>
    <item name="android:windowBackground">@android:color/transparent</item><!--背景透明
-->
    <item name="android:backgroundDimEnabled">true</item><!--模糊-->
</style>
```

利用 style 可以给 activity 设置风格，实现新 activity 悬浮在当前 acitivity 之上等效果

android:windowIsFloating

设置之后当前 activity 不会有整个界面飘进来的感觉，只是当前

activity 的 body 部分弹出

android:windowIsTranslucent

当前 activity 背景透明，可以解决刷新 activity 时的短暂黑屏问题，但是设置之后可能会出现 activity 切换动画失效问题，这时候要主要 Animation 的继承应该设置为

```
parent="@android:style/Animation.Translucent"
```

Tips:

关于 windowNoTitle 和 android:windowNotitle 区别

AppCompatActivity 中出现了 windowNoTitle 属性，并且前边不需要加“Android:”

其实 windowNoTitle 是 appcompat-v7 中的属性在 appcompat-v7\res\values\values.xml 中定义的。

打开 appcompat-v7\res\values\values.xml 搜索 AppCompatTheme 在<declare-styleable name="AppCompatTheme">中可以找到定义的“windowNoTitle”“windowActionBar”等属性。

“windowNoTitle”属性在代码中可以使用 R.attr.windowNoTitle 访问，“android:windowNoTitle”则需要使用 android.R.attr.windowNoTitle 访问。

使用 AppCompatActivity 时（Activity 必须使用 Theme.AppCompat 主题及其子主题），经过测试发现：

“android:windowActionBar”属性在 AppCompatActivity 中不起作用；

windowNoTitle＝false 并且 android:windowNoTitle＝false 时，会出

现两个标题，位于下方的是 AppCompatActivity 的标题栏。

当 windowNoTitle = false , windowActionBar = false 时,会报错:

AppCompat does not support the current theme features

windowNoTitle = true 并且 android:windowNoTitle = true 时 ,无标题。

1.1.3 selector 选择器

属性介绍：

android:state_selected 选中

android:state_focused 获得焦点

android:state_pressed 点击

android:state_enabled 设置是否响应事件,指所有事件

主要属性使用例子：

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <selector xmlns:android="http://schemas.android.com/apk/res/android">
3.  <!-- 默认时的背景图片-->
4.    <item android:drawable="@drawable/pic1" />
5.  <!-- 没有焦点时的背景图片 -->
6.    <item android:state_window_focused="false"
7.          android:drawable="@drawable/pic1" />
8.  <!-- 非触摸模式下获得焦点并单击时的背景图片 -->
9.    <item android:state_focused="true" android:state_pressed="true"   android:
    drawable= "@drawable/pic2" />
```

```
10. <!-- 触摸模式下单击时的背景图片-->
11. <item android:state_focused="false" android:state_pressed="true"   android:d
    rawable="@drawable/pic3" />
12. <!--选中时的图片背景-->
13.   <item android:state_selected="true"   android:drawable="@drawable/pic4" />


14. <!--获得焦点时的图片背景-->
15.   <item android:state_focused="true"   android:drawable="@drawable/pic5" />


16. </selector>
```

1.1.4 shape

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

    <!-- 圆角 -->
    <corners
        android:radius="9dp"
        android:topLeftRadius="2dp"
        android:topRightRadius="2dp"
        android:bottomLeftRadius="2dp"
        android:bottomRightRadius="2dp"/><!-- 设置圆角半径 -->

    <!-- 渐变 -->
    <gradient
        android:startColor="@android:color/white"
        android:centerColor="@android:color/black"
        android:endColor="@android:color/black"
        android:useLevel="true"
        android:angle="45"
        android:type="radial"
        android:centerX="0"
        android:centerY="0"
        android:gradientRadius="90"/>
```

```
    <!-- 间隔 -->
    <padding
        android:left="2dp"
        android:top="2dp"
        android:right="2dp"
        android:bottom="2dp"/><!-- 各方向的间隔 -->

    <!-- 大小 -->
    <size
        android:width="50dp"
        android:height="50dp"/><!-- 宽度和高度 -->

    <!-- 填充 -->
    <solid
        android:color="@android:color/white"/><!-- 填充的颜色 -
->

    <!-- 描边 -->
    <stroke
        android:width="2dp"
        android:color="@android:color/black"
        android:dashWidth="1dp"
        android:dashGap="2dp"/>
```

```
</shape>
```

填充：设置填充的颜色

间隔：设置四个方向上的间隔

大小：设置大小

圆角：同时设置五个属性，则 Radius 属性无效

android:Radius="20dp"　　设置四个角的半径

android:topLeftRadius="20dp"　　设置左上角的半径
android:topRightRadius="20dp"　　设置右上角的半径
android:bottomLeftRadius="20dp"　　设置右下角的半径
android:bottomRightRadius="20dp"　　设置左下角的半径

描边：dashWidth 和 dashGap 属性，只要其中一个设置为 0dp，则边框为实线边框

android:width="20dp"　　设置边边的宽度
android:color="@android:color/black"　　设置边边的颜色
android:dashWidth="2dp"　　设置虚线的宽度
android:dashGap="20dp"　　设置虚线的间隔宽度

渐变：当设置填充颜色后，无渐变效果。angle 的值必须是 45 的倍数（包括 0），仅在 type="linear"有效，不然会报错。android:useLevel 这个属性不知道有什么用。

1.2 java 文件

1.2.1 HttpURLConnection

```
//请求地址 URL
realUrl = new URL("http://seam.ustb.edu.cn:8080/jwgl/Login");
//通过 URL 打开一个请求连接
con = (HttpURLConnection) realUrl.openConnection();
//设置请求方式，主要有 get,post 等
con.setRequestMethod("POST");
//设置请求头，一般需要设置一个 Cooike 用来验证会话 sessionID,这里是登录验证，所以不用
//con.setRequestProperty("Cookie", sessionID);
con.setRequestProperty("Accept", "text/html");
//是否自动重定向
con.setInstanceFollowRedirects(false);
//设置允许输入输出
con.setDoInput(true);
con.setDoOutput(true);
String str = "username=" + userName + "&password="
        + password + "&usertype=student";
con.connect();
//服务器写入请求信息
os = con.getOutputStream();
os.write(str.getBytes());
os.flush();
//获得输出流（服务器响应数据）
is = con.getInputStream();
```

1.2.2 获取网络状态

```
ConnectivityManager cm = (ConnectivityManager)
context.getApplicationContext().getSystemService(Context.CONNECTIVITY_SERVICE);
```

得到一个 cm 对象，cm 包含网络状态，网络连接信息，用来判断用户是否联网或者处于何种网络状态（wifi，数据流量）

1.2.3 回调函数的使用

```
//子线程设置属性
private ResponseForClass responseForClass;
//在 UI 线程中调用 set 方法，传给子线程一个接口的实现，用来在子线程回调，实现线程之间的数据通信
public void setResponseForClass(ResponseForClass responseForClass) {
    this.responseForClass = responseForClass;
}
//在子线程中定义的接口，供 UI 线程实现，以实现在子线程中对 UI 线程实现回调
public interface ResponseForClass {
    void receiveMessage(ArrayList<String> list);
}
```

1.3 开源工程的使用

GifView 实现播放 gif 动画

步骤：

build.gradle(Project)

```
allprojects {
    repositories {
        jcenter()
        maven { url "https://jitpack.io" }
    }
}
```

bulid.gradle(Module)

```
compile 'com.github.Cutta:GifView:1.1'
```

使用 custom 引用 gif 资源。

```
custom:gif="@drawable/roll"
```

需要注册命名空间

```
xmlns:custom="http://schemas.android.com/apk/res-auto"
```