实验二 Button 事件处理

【实验目的】

1、 了解 Android 编程基础

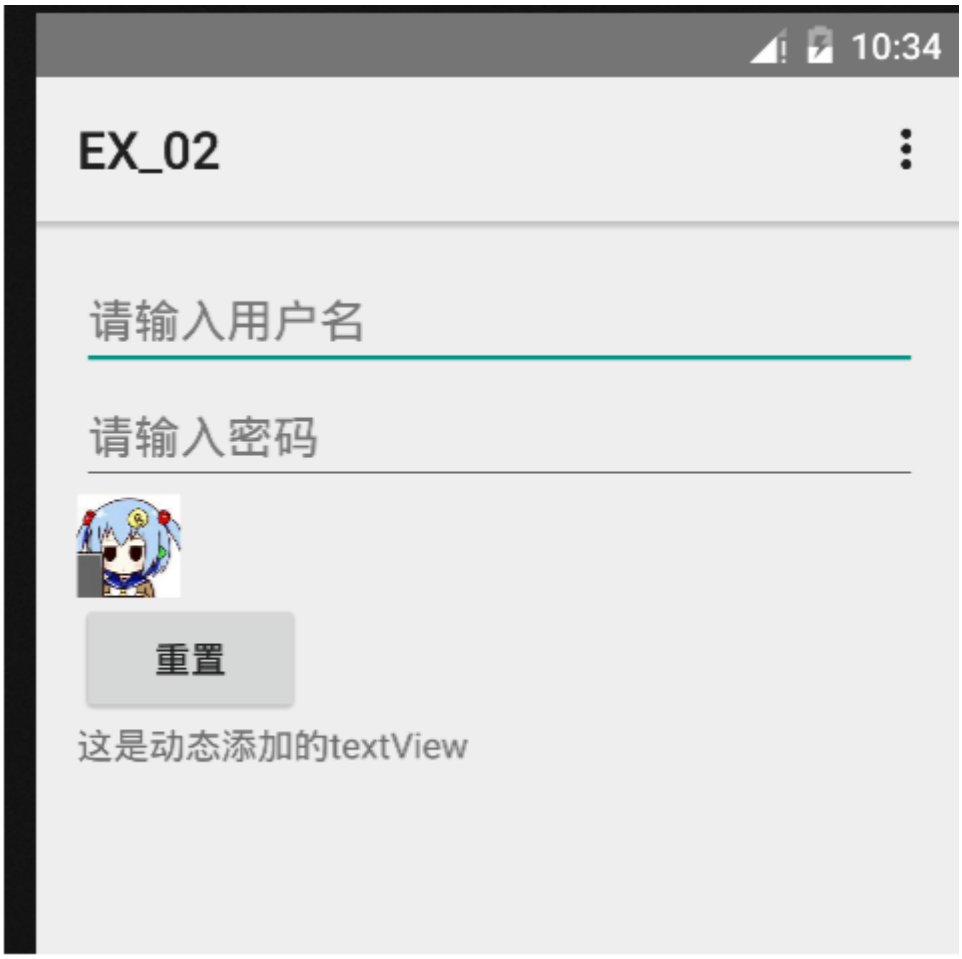
2、 熟悉 Button 控件、 ImageButton 控件并且可以对 Button 事件进行相应处理

【实验内容】

制作一个登陆框，要求用户输入账号和密码。点击一个 ImageButton 按钮，在后台判断用户输入的账号和密码，如果账号是“ LeiBusi”并且密码为”Halo3Q”,则将 ImageButton 按钮的图片换成 State2.png，同时将账号和密码栏隐藏.否则将 ImageButton 按钮的图片换成 state1.png,同时将光标放置于密码栏中，即密码 Edittext 获得焦点，并清空已输入密码，提示“密码错误”。在 ImageButton 按钮旁边要求有一个重置按钮，点击重置按钮要求所有控件回到初始状态。按钮下面用动态添加的方式添加一个 TextView。长按 ImageButton 也能动态添加一个 TextView

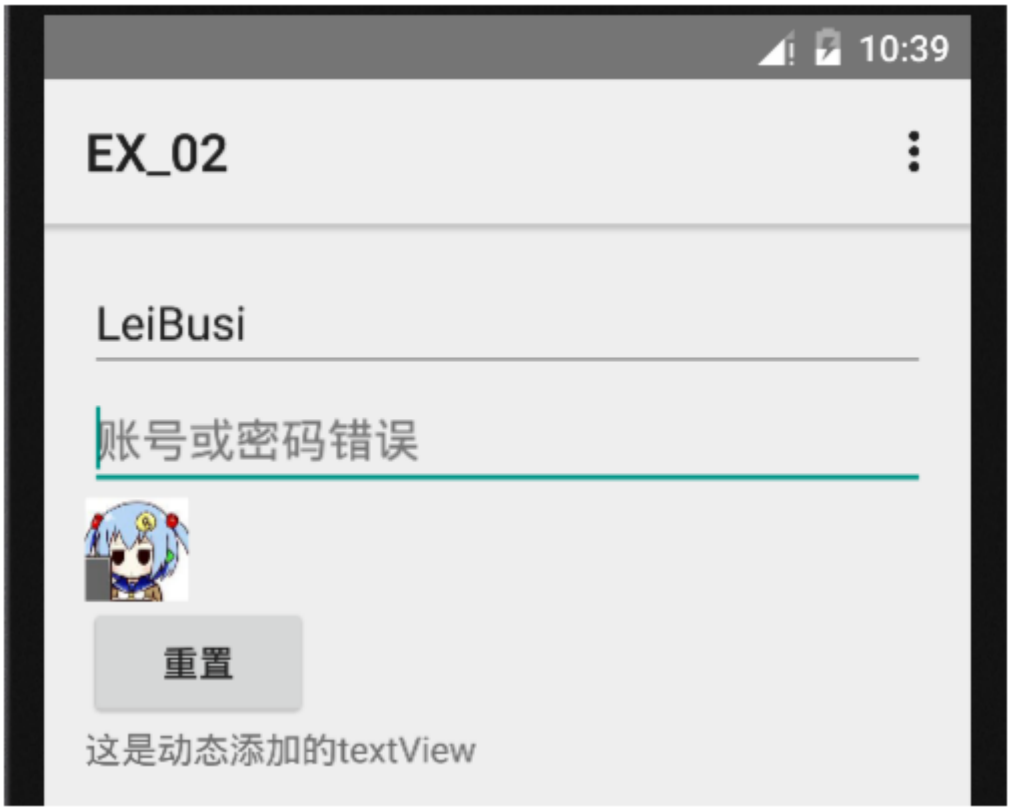
示例如下：

初始状态：



账号为 LeiBusi 密码为 Halo3Q 时，

账号或密码错误时：



点击重置时：


10:41
EX_02
请输入用户名
请输入密码
重置
这是动态添加的textView

长按图形按钮时：


10:42
EX_02
请输入用户名
请输入密码
重置
这是动态添加的textView
这是动态添加的textView
这是动态添加的textView

【参考】

1、 按钮是用的最多的控件，在 Android 平台中，按钮是通过 Button 来实现的，实现过程也非常简单。既然是按钮，被点击之后就要触发事件，所以需要对按钮设置 setOnClickListener 事件进行监听。下面的示例示范了在布局里添加了一个按钮并设计这个按钮的事件处理函数，当单击按钮的同时，更改 TextView 中的文字。

要实现这种效果，需要在 main.xml 里添加 TextView 和 Button 控件

```
<TextView
    android:id="@+id/tv1"
    android:text="Welcome to EX_02" android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
<Button
    android:id="@+id/btn1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="测试1"/>
```

(注意添加 id， 以及 id 的赋值为”@+id/…”)并在 Activity 代码中定义处理按钮事件。首先定义控件

```
private Button mBtn;
private TextView mTv;
```

通过 id 获取控件：（否则会出现 ClassNotFound 错误）

```
mBtn = (Button)findViewById( R.id.btn1);
mTv = (TextView)findViewById(R.id.tv1);
```

按键事件处理：

```
View.OnClickListener listener = new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        mTv.setText("按下了" + ((Button) view).getText());
    }
};
```

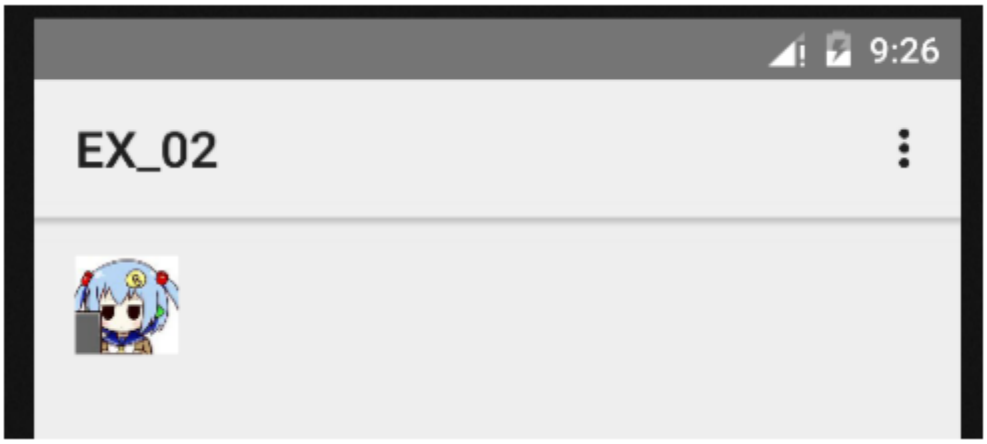
（ OnCLickListener 一共有 3 种实现方式，有兴趣的同学可以多试试）

2、 接下来示范怎样对 ImageButton 设置事件监听。首先，将下面两个按钮图片放到 res/mipmap 文件夹里面。 目录级别为：workspace\EX_02\app\src\main\res（ 不同密度的暂时影响不大）

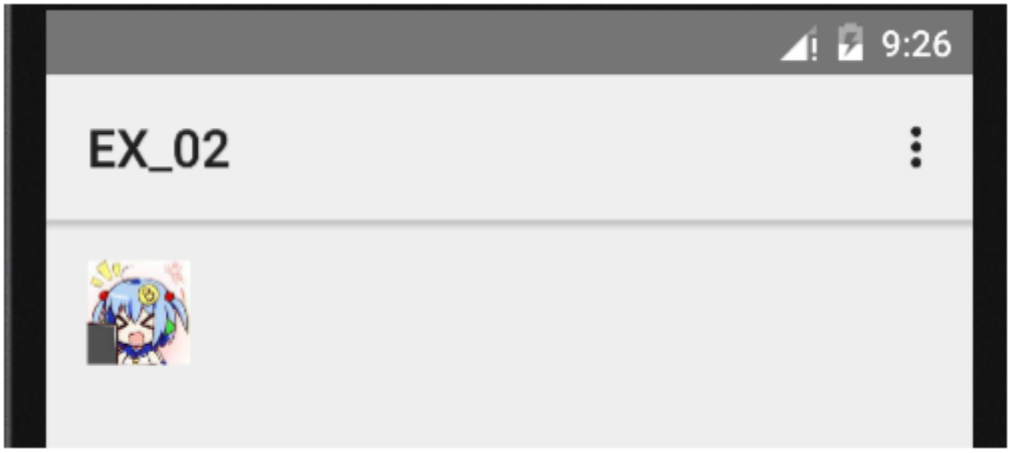
State1: 

State2: 

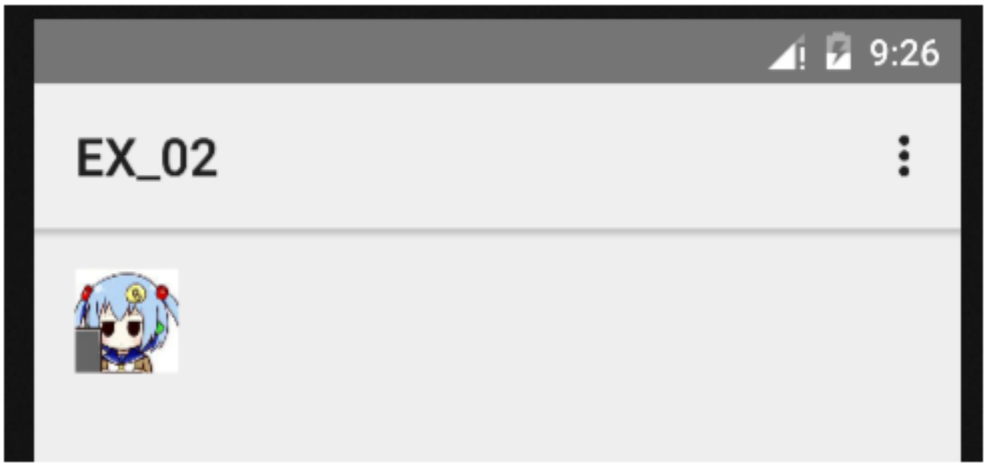
初始化：



点击：



再点击：



实现：
将两个按钮图片分别命名为 state1 和 state2，在 main.xml 里添加 ImageButton 控件。

```
<ImageButton
    android:id="@+id/imgBtn1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#00000000"
    android:src="@mipmap/state1"/>
```

和 1 一样在后台代码中添加 ImageButton 的响应函数，这里实现的效果是 ImageButton 被先初始化为 state1 的状态，当点击 ImageButton 后从 state1 状态变为 state2 状态，再点击 ImageButton 从 state2 状态变为 state1 状态，如此循环下去。

```
View.OnClickListener listener = new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (flag) {
            flag = false;
        } else {
            flag = true;
        }
        if (flag) {
            mImgBtn.setImageDrawable(getResources().getDrawable(R.mipmap.state2));
        } else {
            mImgBtn.setImageDrawable(getResources().getDrawable(R.mipmap.state1));
        }
    }
};
```

3、 动态添加按钮。首先，在 main.xml 里添加根节点 LinearLayout 的 android:id 属性，使后台代码可以通过 id 获取前台布局；然后，实例化一个 Button 对象并设置它的事件监听；最后通过 LinearLayout::addView 方法将 Button 对象添加到布局中。点击前：

点击后：

首先，定义：

```
private Button mBtn;
private Context mContext;
@Override
```

赋值与动态生成：

```
mContext = this;
mBtn = new Button(mContext);
mBtn.setText("动态添加按钮");
```

按键事件：

```
View.OnClickListener listener = new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Toast toast = Toast.makeText(mContext, "动态添加按钮", Toast.LENGTH_LONG);
        toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
        toast.show();
    }
};
```

(在此用到的 toast 类是 Android 里面的一个消息机制，有兴趣的同学可以多查询下)LinearlyLayout 动态添加

```
mBtn.setOnClickListener(listener);
LinearLayout linearLayout = (LinearLayout)findViewById(R.id.main);
linearLayout.addView(mBtn);
```

4、 练习提示

1) 获取 EditText 组件的文字： usr.getText(); //usr 为 EditText 类

设置提示文字：usr.setHint();

2) 组件隐藏： usr.setVisibility(View.GONE);

3) 组件显示： usr.setVisibility(View.VISIBLE);

4) 获取焦点： usr.requestFocus();

5） 长按事件：OnLongClickListener

重写方法为 public boolean onLongClick(View view) {}

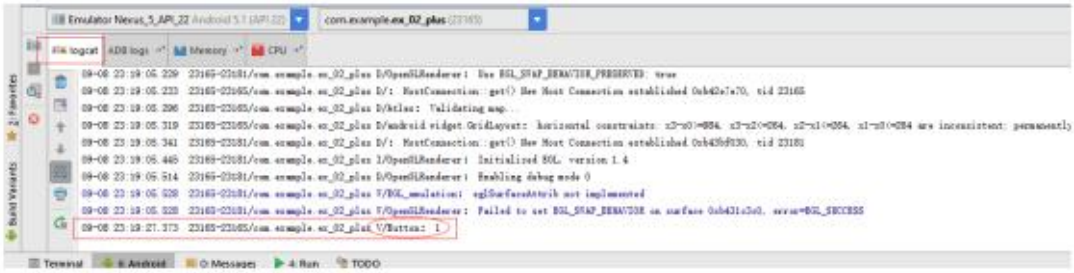
5. Log 类以及 logcat 的应用： 用于输出消息以便追寻值的变化以及 Debug，参数左右都为 String 类型参数(“tag”, “value”)左边为你设置的标签，右边为你想输出的值 1)Log.v 的调试颜色为黑色的，任何消息都会输出，这里的 v 代表 verbose 啰嗦的意思，平时使用就是 Log.v("","");2)Log.d 的输出颜色是蓝色的，仅输出 debug 调试的意思，但他会输出上层的信息，过滤起来可以通过 DDMS 的 Logcat 标签来选择.3)Log.i 的输出为绿色，一般提示性的消息 information，它不会输出 Log.v 和 Log.d 的信息，但会显示 i、 w 和 e 的信息

4) Log.w 的意思为橙色，可以看作为 warning 警告，一般需要我们注意优化 Android 代码，同时选择它后还会输出 Log.e 的信息。

5)Log.e 为红色，可以想到 error 错误，这里仅显示红色的错误信息，这些错误就需要我们认真的分析，查看栈的信息了。例如在按键事件内，有：

```
public void onClick(View view) {
    Log.v("Button", String.valueOf(((Button) view).getText()));
```

则在下方的 Logcat 有:



结合右边这两个可以进一步的筛选:

【扩展项】

设计如下一个手机拨号界面，要求使用动态添加按钮以及绑定响应函数。