宝贵建议请发送至：wangzhenyang@itcast.cn

黑马程序员 itheima.com -编程，始于黑马

Android 课程同步笔记

Alpha 0.01 版

By 阳哥

Android-通知&反编译&NinePatch&Fragment

1.通知（★★★）

通知用于在状态栏显示消息，消息到来时以图标方式表示，如果需要查看消息，可以拖动状态栏到屏幕下方即可查看消息，在 Android 中通过通知管理器 NotificationManager 来发出或关闭一个通知。

使用步骤：

1. 获取通知管理器对象

```
private NotificationManager manager;//通知管理器
//获取通知管理器服务
manager =
(NotificationManager)getSystemService(NOTIFICATION_SERVICE);
```

2. 如何发出一个通知

```
public void notify(View view) {
        // 创建一个通知
        Notification notification = new
Notification(R.drawable.ic_launcher,// 消息的图标
                "您有一条来自黑马程序员的通知", // 消息的标题
                System.currentTimeMillis());// 消息发送的时刻 立即发送
        // 定义一个隐式意图：指定点击通知时要打开的 Activity
        Intent intent = new Intent();
        intent.setAction("com.itheima.noftify");
        /*
         * PendingIntent
         * 是延时意图，在未来某个时间开启一个界面，并且可以指定使用的次数，
其实就是对 Intent 进行的一个包装，并且指定了使用次数
         */
```

```
PendingIntent pendingIntent = PendingIntent.getActivity(this, 100,
intent, PendingIntent.FLAG_ONE_SHOT);
        //设置消息的内容和意图
        notification.setLatestEventInfo(this, "这是详细通知的标题", "
这是通知的详细内容", pendingIntent);
        //设置通知点击后自动关闭
        notification.flags = Notification.FLAG_AUTO_CANCEL;
        //设置消息发送时开启灯光、声音、震动等特效，如果震动开启了需要设置
权限
        notification.defaults = Notification.DEFAULT_ALL;
        //发送通知，给该通知的 id 为 1
        manager.notify(1, notification);
    }
```

3. 如何关闭通知

```
    // 关闭通知
    public void close(View view) {
        //关闭 id 为 1 的通知
        manager.cancel(1);
    }
```

4. 通知的效果如下

点击通知，会跳转到指定页面。

注意：

1. 若设置了弹出通知会有声音/震动/亮灯的效果，注意添加对应权限，否则会抛错比如，设置震动需加权限 android.permission.VIBRATE

2．获取延期意图 PendingIntent 时，封装的意图对象必须采用隐式的方式

2.反编译（★）

Android 打包好的 APK 如果直接解压缩，那么里面的布局文件和字节码文件如法直接使用，但是我们可以通过以下三个反编译工具对一个 APK 进行反编译。

1.apktool 反编译布局文件，反编译之后会在 apk 相同的目录下创建一个

命令：apktool d xxx.apk

2. dex2jar 把.dex 文件转换成.jar 的文件，会在相同目录下生成一个 xxx.jar

命令：dex2jar xxx.dex

3.jd-gui 查看 jar 文件

3.NinePatch（★）

3.1 什么是 NinePatch 图片

NinePatch 是一种很有用的 PNG 图片文件夹格式，它可以在特定区域随文字大小进行缩放

上图中背景图片的中间区域会随着文字的大小进行缩放背景图片就是一张 NinePatch 图片。

3.2 如何制作 NinePatch 图片

NinePatch 图片可以使用 android 自带的 draw9patch 工具来制作，该工具在 SDK 安装路径的 tools 目录下

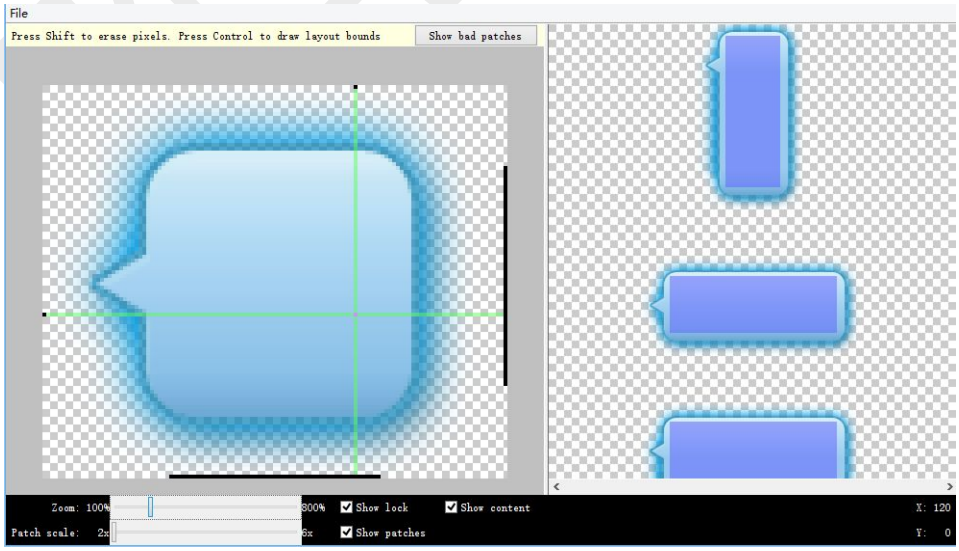
打开“sdk\tools\draw9patch.bat”，执行此工具然后点击“File”->“open 9-path”

打开一张用于制作 NinePatch 图片的原图，通过在画布的四边画线来指定缩放区域和文字所在区域。

画布的左边和上边是控制图片拉伸的;画布的右边和底边是控制内容显示的区域的

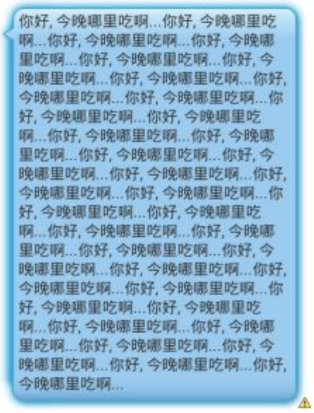
使用鼠标左键点击拖动来画线，使用鼠标右键点击拖动才擦除线条

点击 File，点击保存，就会生成后缀名为”.9.png”格式的图片



3.3 使用 NinePatch 图片做背景的效果

通过设置 android:background="@drawable/bg"属性来设置背景图片，使用一般图片作为背景和使用 NinePatch 图片做背景的效果对比。



4.Fragment（★★★★）

4.1 什么是 Fragment

Fragment 就是小型的 Activity,它是在 Android3.0 时出现的。

Fragment 是表现 Activity 中 UI 的一个行为或者一部分。可以把 fragment 想象成 activity 的一个模块化区域，有它自己的生命周期，接收属于它自己的输入事件，并且可以在 activity 运行期间添加和删除（有点像一个可以在不同的 activity 中重用的“子 Activity”）。

Fragment 必须被嵌入到一个 activity 中。它们的生命周期直接受其宿主 activity 的生命周期影响。当一个 activity 正在运行时，就可以独立地操作每一个 Fragment，比如添加或删除它们。

Fragment 可以定义自己的布局、生命周期回调方法，因此可以将 fragment 重用到多个 activity 中，因此可以根据不同的屏幕尺寸或者使用场合改变 fragment 组合。

4.2 如何创建一个 Fragment

1. 为 Fragment 定义一个布局
2. 定义类继承 Fragment
3. 重写类中的 onCreateView 方法 ,返回一个 View 对象作为当前 Fragment 的布局。

   fragment 第一次绘制它的用户界面的时候，系统会调用 onCreateView()方法。为了绘制 fragment 的 UI，此方法必须返回一个作为 fragment 布局的根的 view。

   如果 fragment 不提供 UI，可以返回 null。

```
/**
 * 定义类继承 Fragment
 */
public class TitleFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {
        //使用打气筒生成一个 View 对象
        View view = inflater.inflate(R.layout.fragment_title, null);
        return view;
    }
}
```

4.3 如何将 Fragment 添加到 Activity

Activity 必须在清单文件中进行声明，但是 Fragment 不需要，Fragment 只需要在 Activity 的布局文件中声明就可以了。

```
<fragment
        android:id="@+id/fmt_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:name="com.itheima.fragment.TitleFragment"
        />
```

注意：代码中的四个属性是必须的要给的，“android:name”属性：指定了在 layout

中实例化的 Fragment 类是哪个。

当系统创建这个 activity layout 时，它实例化每一个在 layout 中指定的 Fragment，并调用它们的 onCreateView()方法，来获取每一个 Fragment 的 layout，系统将从 Fragment 返回的 View 直接插入到<fragment>元素所在的地方

每一个 fragment 都需要一个唯一的标识，如果 activity 重启，系统可以用来恢复 Fragment，并且可以用 id 来捕获 Fragment 来处理事务，例如移除它。

有 3 种方法来为一个 fragment 提供一个 ID：

为 android:id 属性提供一个唯一 ID;

为 android:tag 属性提供一个唯一字符串;

如果以上 2 个你都没有提供，系统将使用容器 view 的 ID;

4.4 如何切换 Fragment

要在 activity 中管理 Fragment，需要使用 FragmentManager 可以通过调用 activity 的 getFragmentManager()取得它的实例。

案例：点击不同的按钮切换到不同的 Fragment 进行显示

步骤：

1. 设置布局文件：添加三个按钮用于切换 Fragment，并在按钮下方添加一个 FrameLayout 用来替换成响应的 Fragment。
2. 创建三个 Fragment ,SportsFragment、NewsFragment、GameFragment。

```
public class SportsFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {
        // 使用打气筒生成一个 View 对象
```

```
        View view = inflater.inflate(R.layout.fragment_sports, null);
        return view;
    }
}
```

其余两个 Fragment 跟 SportsFragment 代码一致，只是布局文件不同。

3. 添加切换 Fragment 的逻辑

```
public void news(View view){
        //获取 Fragment 管理器对象
        FragmentManager manager = getFragmentManager();
        //开启事务
        FragmentTransaction transaction = manager.beginTransaction();
        //将 FrameLayout 控件替换成 Fragment 对象
        transaction.replace(R.id.frame, new NewsFragment());
        //提交事务
        transaction.commit();
    }
```

sports（）方法、games（）方法同上，因此不再给出代码清单。

4. 运行程序，效果如下

4.5 Fragment 的生命周期

Fragment 的生命周期和 activity 生命周期很像。

onAttach：绑定到 activity

onCreate：创建 fragment

onCreateView： 创建 fragment 的布局

onActivityCreated： activity 创建完成后

onStart： 可见，不可交互

onResume： 可见，可交互

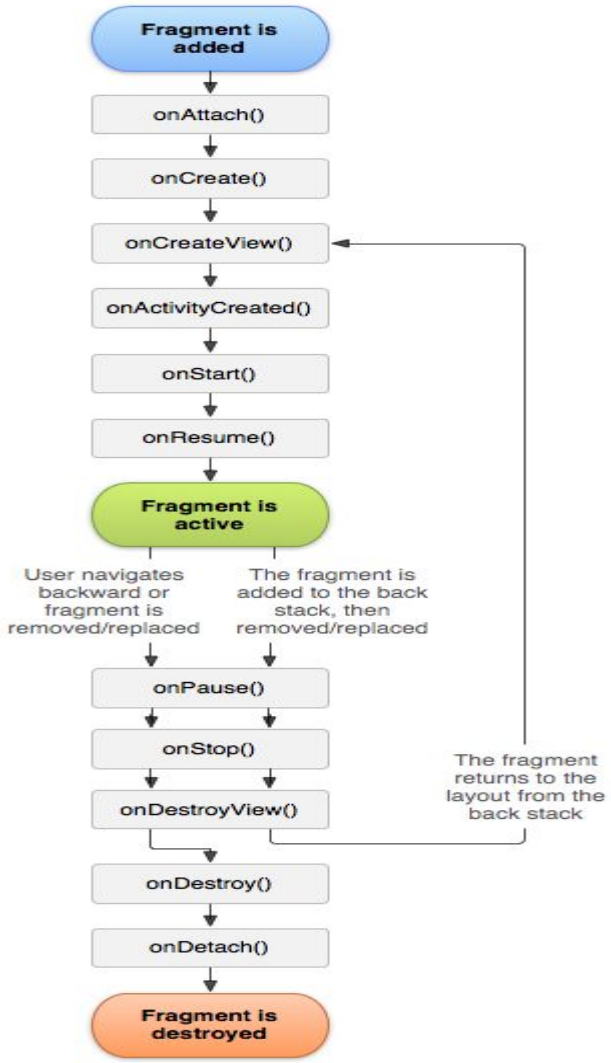
onPause： 部分可见，不可交互

onStop：不可见

onDestroyView： 销毁 fragment 的 view 对象

onDestroy： fragment 销毁了

onDetach： 从 activity 解绑了



4.6 Fragment 的向下兼容

Fragment 是在 Android 3.0 才推出的，若想在 3.0 的低版本下使用 Fragment，则需要执行下面 2 步：

1. 把所有 Fragment 和 FragmentManager 改成 support-v4 包下的类
2. 把 Activity 的继承改为 FragmentActivity(support-v4 包下的)

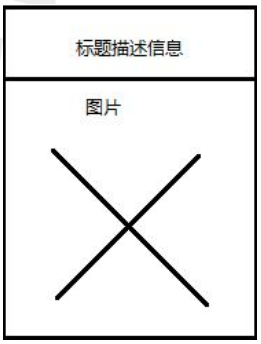
4.7 Fragment 之间的通信案例

案例：创建一个用于显示选项卡的 Fragment 和一个用于显示内容的 Fragment，当选项卡切换时，使内容的 Fragment 信息跟着一起切换。

步骤：

1. 先创建选项卡 Fragment 和内容 Fragment，并在 activity_main.xml 布局文件中进行设置，其布局效果如下：

2. 为选项卡 Fragment 创建布局文件，其布局效果如下：



3. 添加内容区域 Fragment 的代码逻辑

```
public class ContentFragment extends Fragment {
    private TextView tv_title;//标题
    private ImageView iv_pic;//内容
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {
        //使用打气筒填充布局
        View view = inflater.inflate(R.layout.fragment_content,null);
        //获取布局中的控件
        tv_title = (TextView) view.findViewById(R.id.tv_title);
        iv_pic = (ImageView) view.findViewById(R.id.iv_pic);
        return view;
    }
    //定义一个方法，用于改变标题和图片
    public void setTitleAndImage(String title,int picId){
        tv_title.setText(title);
        iv_pic.setImageResource(picId);
    }
}
```

4.添加选项卡区域 Fragment 的代码逻辑

```
public class TabFragment extends Fragment implements
OnItemClickListener {
    private String[] datas;
    private int[] picIds;
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {
        //直接使用 ListView 作为布局，因此不需要布局文件
        ListView listView = new ListView(getActivity());
        //定义一些常量数据作为册数数据
        datas = new String[]{"新闻","体育","财经","社会","娱乐","国际"};
        picIds = new
int[]{R.drawable.a0,R.drawable.a1,R.drawable.a2,R.drawable.a3,R.dr
awable.a4,R.drawable.a5};
        //设置点击事件
        listView.setOnItemClickListener(this);
        //声明一个 ArrayAdapter 对象
        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(getActivity(),
android.R.layout.simple_expandable_list_item_1, datas);
```

```
        //给 ListView 设置 Adapter
        listView.setAdapter(adapter);
        //返回视图
        return listView;
    }
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {
        //获取当前的 title 和图片 id
        String title = datas[position];
        int picId = picIds[position];
        //获取 Fragment 管理器
        FragmentManager manager = getFragmentManager();
        //通过 ID 选择出 ContentFragment 对象
        ContentFragment fragment = (ContentFragment)
manager.findFragmentById(R.id.fragment_content);
        //调用 ContentFragment 的方法，给 ContentFragment 传递参数，实现不
同 Fragment 直接的通信
        fragment.setTitleAndImage(title, picId);
    }
}
```

5.运行程序，效果如下：

至此，本文档完！

2014 年 12 月 26 日星期五 0:49:31
北京市海淀区东北旺中路东馨园