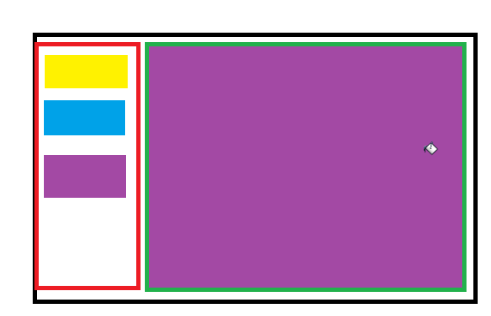
# Android3.0的新特性

# Fragment



* 在一个activity上，点击黄色按钮，右边显示黄色内容，点紫色，右边显示紫色

## 实现过程

* 使用android.app里面的api
* 先定义出activity的布局：
  + 左边线性布局显示三个按钮的内容
  + 右边帧布局，显示fragment的内容
  + 以为要显示fragment的内容，必须使用帧布局

|  |
| --- |
| <LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*  xmlns:tools=*"http://schemas.android.com/tools"*  android:layout\_width=*"match\_parent"*  android:layout\_height=*"match\_parent"*  tools:context=*".MainActivity"*  android:orientation=*"horizontal"* >  <LinearLayout  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  android:orientation=*"vertical"*>    <Button  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  android:text=*"fragment1"*  android:onClick=*"fragment1"*/>  <Button  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  android:text=*"fragment2"*  android:onClick=*"fragment2"*/>  <Button  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  android:text=*"fragment3"*  android:onClick=*"fragment3"*/>  </LinearLayout>    <FrameLayout  android:id=*"@+id/fl"*  android:layout\_width=*"match\_parent"*  android:layout\_height=*"match\_parent"*>    </FrameLayout>  </LinearLayout> |

* 定义三个布局文件，分别显示三个fragment的内容
* 然后定义三个fragment类，继承android.app.Fragment
  + 重写：View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
    - 在这里面，把要在fragment01上显示的布局文件，填充成view对象返回
    - 可以使用方法参数上的布局填充器，也可以自行获取布局填充器对象，获取布局填充器对象需要一个上下文。Fragment#getActivity()，

因为fragment也是显示在activity上的，所以可以获取到activity

# Fragment的向下兼容



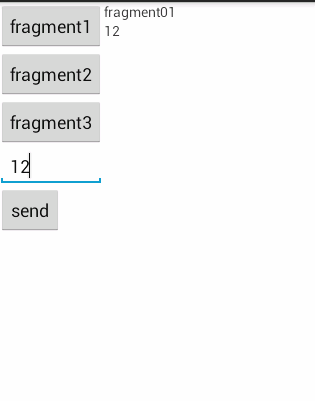
* 这个包，是为了低版本能使用高版本api而存在的过度包。这里面封装有低版本不能使用的高版本api。虽然不是所有的高版本api都在里面，但是fragment正是其中一个
* 在低版本中，使用这个包里面的fragment。就能实现上面fragment的效果

## 实现过程

* 把上面所有跟fragment有关的api都改成过度包下面的api
  + android.support.v4.app.Fragment;
  + android.support.v4.app.FragmentManager;
  + android.support.v4.app.FragmentTransaction;
* 因为getFragmentManager()得到的是android.app.FragmentManager;
* 但是我们这里的是android.support.v4.app.FragmentManager;
* 所以为了能得到android.support.v4.app.FragmentManager;
* 把MAinActivity改成继承android.support.v4.app.FragmentActivity;
* 再使用android.support.v4.app.FragmentActivity# getSupportFragmentManager();
* 得到android.support.v4.app.FragmentManager;

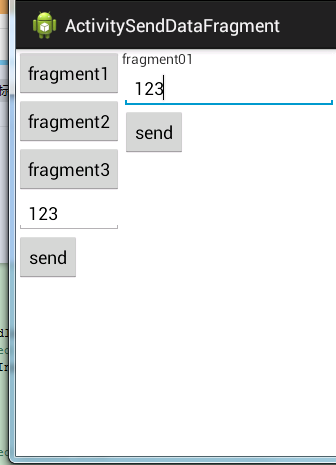
# Activity和Fragment之间传递数据

## Activity传递数据给fragment



* send按钮按下，就把数据显示大fragment1的布局文件上（右边）
* 在Fragment01类中，得到fragment01布局文件中的TextView
* 再在Fragment01类中，定义一个修改TextView文本数据的方法
* 因为在MainActivity中，可以得到Fragment01的对象，所以直接调用它修改数据的方法即可

## fragment传递数据给Activity



* 点击fragment01的按钮，把数据发送到右边
* MAinActivity中设置修改文本的方法，被Fragment01调用
* Fragment01使用Fragment#getActivity()方法，得到他所在的activity的对象
* 有两点需要注意：
  + - 右边的按钮的点击事件，需要设置侦听才行了。而不能使用onclick属性
      * 因为，onclick属性，是调用布局文件所在上下文对象中的方法
      * 而，fragment并不是上下文对象或其子类。所以不行
    - 右边和左边都有两个输入框。是两个布局文件。两个布局文件中，是可以有相同id的
    - 但是这里有个小问题：

MAinActivity

|  |
| --- |
| **public** **void** setText(String text) {  EditText et\_main = (EditText) findViewById(R.id.*et\_main*);  et\_main.setText(text);  } |

Fragment01

|  |
| --- |
| //设置点击侦听  Button bt\_f01 = (Button) v.findViewById(R.id.*bt\_f01*);  bt\_f01.setOnClickListener(**new** OnClickListener() {    @Override  **public** **void** onClick(View v) {  String text = et\_f01.getText().toString();  //得到Fragment01所在的activity，把数据传递过去  ((MainActivity)getActivity()).setText(text);  }    }); |

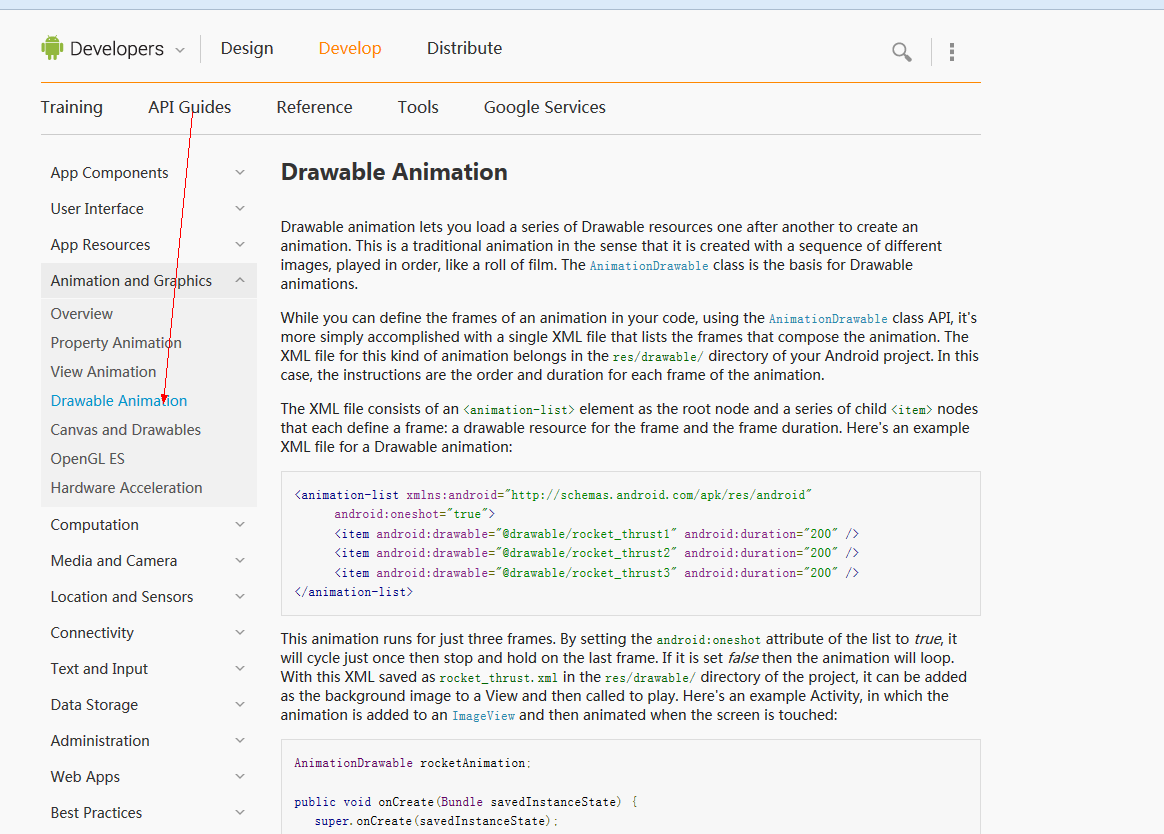
* 因为：setText()方法是在Fragment01中调用的，所以，setText方法中的findViewById()是会现在Fragment01的布局文件中查找组件的，如果没有找到组件，再去MIanActivity的布局文件中找
* 所以：如果两个布局文件中，如果有两个组件相同id，就会导致findViewById()在Fragme01的布局文件中找到了组件，然后把文本设置到了这个组件上，最后MAinActivity布局文件上的组件的文本并没有被改变
* 解决方法：两个
  + Id不同名
  + 把MAinActivity#setText()方法中的查找组件那一句，放到oncreate()中运行即可

# Fragment的生命周期方法

* Fragment切换是先销毁旧的在创建新的
* Fragment的生命周期方法和activity的生命周期方法是绑定的，即activity的生命周期方法调用，Fragment的生命周期方法也调用
  + onCreate(Bundle savedInstanceState)
  + onStart()
  + onResume()
* onPause()
* onStop()
* onDestroy()

# 帧动画FrameAnimation|Drawable Animation

* 多张图片快速切换，形成动画效果



* 把要显示的图片放在drawable-hdpi中
* 在res中，创建一个文件夹drawable，存放帧动画的xml文件
* 把帧动画定义在xml文件中

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"utf-8"*?>  <animation-list xmlns:android=*"http://schemas.android.com/apk/res/android"*  android:oneshot=*"true"*>  <item android:drawable=*"@drawable/g1"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g2"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g3"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g4"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g5"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g6"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g7"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g8"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g9"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g10"* android:duration=*"200"* />  <item android:drawable=*"@drawable/g11"* android:duration=*"200"* />  </animation-list> |

* 在布局文件中，使用ImageView显示图片
* 除了可以用src指定显示的图片外，还可以使用background来显示图片
* Background能指定资源id，也可以指定资源文件
* 两个属性的显示效果是有差别的
* 在MainActivity中，进行图片的切换显示

# 补间动画

* 原形态变成新形态时为了过渡变形过程，生成的动画就叫补间动画
* 在android中有四种：
  + 位移
  + 旋转
  + 缩放
  + 透明

## 位移

## 旋转

## 缩放

## 透明

## 所有动画一起播放（一起飞）

# 属性动画（3.0新特性，不可向下兼容）PropertyAnimation|ObjectAnimator

* 和补间动画的区别：
* 补间动画只是界面上的一个效果，实际上组件是没有改变原来所在位置的
* 属性动画是真实的改变组件的位置

## 位移

## 缩放

## 透明

## 旋转

## 可以使用xml文件定义属性动画

* 放在res\ animator下面即可

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"utf-8"*?>  <set xmlns:android=*"http://schemas.android.com/apk/res/android"* >  <objectAnimator  android:propertyName=*"translationX"*  android:duration=*"2000"*  android:repeatCount=*"1"*  android:repeatMode=*"reverse"*  android:valueFrom=*"10"*  android:valueTo=*"100"*></objectAnimator>  </set> |