Demo56-ViewPager的基本Demo

1. activity\_guide.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/activity\_guide"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**android.support.v4.view.ViewPager  
 android:id="@+id/viewpager\_guide"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
 </**android.support.v4.view.ViewPager**>  
  
  
 <**Button  
 android:id="@+id/btn\_start\_main"  
 android:background="@drawable/btn\_start\_main\_selector"  
 android:textColor="@drawable/btn\_start\_main\_text\_selector"  
 android:text="开始体验"  
 android:paddingLeft="20sp"  
 android:paddingRight="20sp"  
 android:visibility="gone"  
 android:layout\_marginBottom="80dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 <**RelativeLayout  
 android:layout\_marginBottom="40dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"**>  
  
 <**LinearLayout  
 android:id="@+id/ll\_point\_group"  
 android:orientation="horizontal"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 <**ImageView  
 android:id="@+id/iv\_red\_point"  
 android:layout\_width="10dp"  
 android:layout\_height="10dp"  
 android:background="@drawable/point\_red"** />  
 </**RelativeLayout**>  
  
</**RelativeLayout**>

2、GuideActivity.java

**import** android.app.Activity;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.v4.view.PagerAdapter;  
**import** android.support.v4.view.ViewPager;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.view.ViewTreeObserver;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.LinearLayout;  
**import** android.widget.RelativeLayout;  
**import** com.example.newbeijing.R;  
**import** com.example.newbeijing.SplashActivity;  
**import** com.example.newbeijing.utils.CacheUtil;  
**import** com.example.newbeijing.utils.DensityUtil;  
**import** java.util.ArrayList;  
  
*/\*\*  
 \* 面向对象的本质：构造对象，调用象方  
 \*/***public class** GuideActivity **extends** Activity {  
  
 *//两灰色点的间距声明* **private int leftMax**;  
 *//存放10dp转为px后的值，用于动态设置灰色点point的大小也是声明* **private int width**;  
  
 *//view声明* **private** ViewPager **viewpager\_guide**;  
 **private** Button **btn\_start\_main**;  
 **private** LinearLayout **ll\_point\_group**;  
 **private** ArrayList<ImageView> **imageViews**;  
 **private** ImageView **iv\_red\_point**;  
  
  
 *//继承重写Activity的生命周期方法：onCreate()，且super先调用父类的onCreate做一些初始化操作吧（可查一下）* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
  
 *//初始化view* initView();  
  
 *//初始化数据* initData();  
  
  
 *//把Data填充到view显示出来* setData();  
  
 }  
  
 *//初始化view* **private void** initView() {  
  
 *//将layout初始化成主view* setContentView(R.layout.***activity\_guide***);  
  
 *//初始化view对象* **viewpager\_guide** = (ViewPager) findViewById(R.id.***viewpager\_guide***);  
 **btn\_start\_main** = (Button) findViewById(R.id.***btn\_start\_main***);  
 **ll\_point\_group** = (LinearLayout) findViewById(R.id.***ll\_point\_group***);  
 **iv\_red\_point** = (ImageView) findViewById(R.id.***iv\_red\_point***);  
 }  
  
 *//初始化Data* **private void** initData() {  
  
 *//初始化Data：ViewPager的数据  
 // 初始化ImageView对象集合，等等添加值* **imageViews**=**new** ArrayList<>();  
  
 *//初始化Data：将drawable图片转化成int数组* **int**[] ids=**new int**[]{  
 R.drawable.***guide\_1***,  
 R.drawable.***guide\_2***,  
 R.drawable.***guide\_3*** };  
  
 *//初始化Data：准备好ViewPager数据* **for** (**int** i=0;i<ids.**length**;i++){  
 *//构造局部ImageView对象* ImageView imageView=**new** ImageView(**this**);  
 *//imageView对象调用setter初始化内部值* imageView.setBackgroundResource(ids[i]);  
  
 *//imageView集合的对象调用象方初始化内部值* **imageViews**.add(imageView);  
  
  
 *//构造局部ImageView对象,准备好灰色点的数据* ImageView point=**new** ImageView(**this**);  
 *//imageView对象调用setter初始化内部值  
 //值是自定义的drawable xml文件* point.setBackgroundResource(R.drawable.***point\_normal***);  
  
 *//调用象方得到10dp的px值* **width**= DensityUtil.*dip2px*(**this**,10);  
  
 *//构造LayoutParams对象，用于设置layout内部的布局参数* LinearLayout.LayoutParams params=**new** LinearLayout.LayoutParams(**width**,**width**);  
 **if** (i!=0){  
 *//调用象方初始化属性值* params.**leftMargin**=**width**;  
 }  
 point.setLayoutParams(params);  
 *//将Data添加到view显示出来，view也是Data的一种类型* **ll\_point\_group**.addView(point);  
 }  
 }  
  
  
 *//填充数据到view，显示出来，大多数是依靠适配器和监听接口* **private void** setData() {  
  
 *//填充Data到view显示出来：传入view的数据整理类adapter  
 // 适配器adapter应该理解为数据整理类  
 //真正显示view的方法是内置在ViewPager内部的，adapter只是整理好数据到对象方法里面  
 //让view内部去调用adapter的象方，就显示出来了  
 //正确理解adapter:数据整理类，数据写到象方，给view内部调用象方，显示成view* **viewpager\_guide**.setAdapter(**new** MyPagerAdapter());  
  
 *//红点移动=两灰色点间距\*临时屏幕滑动百分比,所以要求 两灰色点间距 屏幕百分比  
 //用视图树的根节点监听和和view绘制过程中的回调象方来获取 两灰色点间距* **iv\_red\_point**.getViewTreeObserver().addOnGlobalLayoutListener(**new** MyOnGlobalLayoutListener());  
  
 *//将Data填充到view显示出来：传入viewPager页面改变监听结果的处理类对象，回调象方处理* **viewpager\_guide**.addOnPageChangeListener(**new** MyOnPageChangeListener());  
  
 *//将Data填充到view显示出来：传入Button的点击监听结果的处理类对象，回调处理监听结果* **btn\_start\_main**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 *//用缓存类保存进入过主页面* CacheUtil.*putBoolean*(GuideActivity.**this**, SplashActivity.***START\_MAIN***,**true**);  
  
 *// 跳转到MainActivity* Intent intent=**new** Intent(GuideActivity.**this**,MainActivity.**class**);  
 startActivity(intent);  
  
 *//* finish();  
 }  
 });  
 }  
  
  
  
  
 *//adapter是ViewPager的数据整理类，把数据整理写到象方  
 // 然后在ViewPager内部的View显示方法里调用此象方* **class** MyPagerAdapter **extends** PagerAdapter{  
  
 */\*\*  
 \* 返回数据的总个数,即就是ViewPager的页面个数  
 \** ***@return*** *\*/* @Override  
 **public int** getCount() {  
 *// return 3;也行* **return imageViews**.size();  
 }  
  
 */\*\*  
 \*  
 \** ***@param container*** *container就是ViewPager；  
 \* 因为ViewPager是view，所以它也是一个容器  
 \** ***@param position*** *要创建页面的位置；  
 \** ***@return*** *返回和创建当前页面右关系的值；  
 \*/* @Override  
 **public** Object instantiateItem(ViewGroup container, **int** position) {  
 ImageView imageView=**imageViews**.get(position);  
 container.addView(imageView);  
 **return** imageView; *//作为下面isViewFromObject的object参数* }  
  
 */\*\*  
 \* 销毁页面  
 \** ***@param container*** *ViewPager  
 \** ***@param position*** *要页面的位置  
 \** ***@param object*** *要销毁的页面  
 \*/* @Override  
 **public void** destroyItem(ViewGroup container, **int** position, Object object) {  
 *//super.destroyItem(container, position, object);默认注销，否则会崩溃* container.removeView((View) object); *//把object指定成View类型* }  
  
 */\*\*  
 \* 此方法是要判断当前创建的视图和返回的页面是不是一样  
 \** ***@param view*** *当前创建的视图  
 \** ***@param object*** *上面instantiateItem返回的结果值  
 \** ***@return*** *\*/* @Override  
 **public boolean** isViewFromObject(View view, Object object) {  
 **return** view==object;  
 }  
 }  
  
  
 */\*\*  
 \* 同于回调象方拿到两灰色点间距  
 \*/* **class** MyOnGlobalLayoutListener **implements** ViewTreeObserver.OnGlobalLayoutListener {  
  
 @Override  
 **public void** onGlobalLayout() {  
 *//因为我们只需要执行一次就可以获得间距，就不需要多次执行，所以监听完先remove掉  
 //这就是提高性能的细节处理\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*非常重要\*\*\*\*\*\*\*\*\** **iv\_red\_point**.getViewTreeObserver().removeGlobalOnLayoutListener(**this**);  
 **leftMax**=**ll\_point\_group**.getChildAt(1).getLeft()-**ll\_point\_group**.getChildAt(0).getLeft();  
 }  
 }  
  
  
 */\*\*  
 \* ViewPager页面滑动监听结果的处理类，传到ViewPager内部进行回调该象方  
 \*/* **class** MyOnPageChangeListener **implements** ViewPager.OnPageChangeListener {  
 */\*\*  
 \* 当屏幕滑动的时候会回调此方法  
 \** ***@param position*** *代表当前滑动的是哪一张页面  
 \** ***@param positionOffset*** *页面滑动百分比==屏幕滑动百分比  
 \** ***@param positionOffsetPixels*** *滑动的像素  
 \*/* @Override  
 **public void** onPageScrolled(**int** position, **float** positionOffset, **int** positionOffsetPixels) {  
  
 *//红点移动的距离==两灰色点间距\*屏幕百分比 leftMax是两灰色点的间距 leftmagin就是红点最终坐标  
 //红点的坐标==原始坐标+滑动坐标* **int** leftmargin= (**int**) (position\***leftMax**+(**leftMax**\*positionOffset));  
 *//因为iv\_red\_point是写在layout中的，所以要getLayoutParams(),别模仿灰色点的写法，不过原理是一样的* RelativeLayout.LayoutParams params=(RelativeLayout.LayoutParams)**iv\_red\_point**.getLayoutParams();  
 params.**leftMargin**=leftmargin;  
 *//红点调用象方设置自己的布局属性* **iv\_red\_point**.setLayoutParams(params);  
  
 }  
  
 */\*\*  
 \* 当页面被选中的时候调用  
 \** ***@param position*** *\*/* @Override  
 **public void** onPageSelected(**int** position) {  
  
 **if** (position==**imageViews**.size()-1){  
 **btn\_start\_main**.setVisibility(View.***VISIBLE***);  
 }**else**{  
 **btn\_start\_main**.setVisibility(View.***GONE***);  
 }  
 }  
  
 */\*\*  
 \* 当页面状态改变的时候回调此方法： 静止 拖拽 惯性 总共3种状态  
 \** ***@param state*** *\*/* @Override  
 **public void** onPageScrollStateChanged(**int** state) {  
  
  
 }  
 }  
  
  
  
  
  
  
  
}