StackView

Компонент **StackView** представляет собой набор прямоугольных областей в виде стопки, которые циклически сменяют друг друга по кругу. Данный элемент появился в Android 3.0. Управлять перемещением элементов можно пальцами на сенсорном экране или программно через соответствующие методы.

Создадим новый проект и добавим **StackView** и две кнопки для экспериментов.

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="horizontal" >

<StackView

android:id="@+id/mStackView"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:loopViews="true" />

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal" >

<Button

android:id="@+id/previousButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Назад" />

<Button

android:id="@+id/nextButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Вперед" />

</LinearLayout>

</LinearLayout>

Для **StackView** нужно создать адаптер, который свяжет набор данных, которые нужно отобразить на поверхностях отдельных элементов рассматриваемого представления. Реализуем свой адаптер **ColorAdapter.java** на основе BaseAdapter, в котором будут использоваться цветные прямоугольники.

package ru.alexanderklimov.stackviewdemo;

import android.content.Context;

import android.view.View;

import android.view.ViewGroup;

import android.widget.BaseAdapter;

import android.widget.LinearLayout;

public class ColorAdapter extends BaseAdapter {

private Context mContext;

private int[] mColors;

public ColorAdapter(Context context, int[] colors) {

mContext = context;

mColors = colors;

}

@Override

public int getCount() {

return mColors == null ? 0 : mColors.length;

}

@Override

public Object getItem(int position) {

return mColors == null ? null : mColors[position];

}

@Override

public long getItemId(int position) {

return position;

}

@Override

public View getView(int position, View cacheView, ViewGroup parent) {

LinearLayout.LayoutParams colorLayoutParams = new LinearLayout.LayoutParams(

200, 300);

LinearLayout colorLayout = new LinearLayout(mContext);

colorLayout.setBackgroundColor(mColors[position]);

colorLayout.setLayoutParams(colorLayoutParams);

return colorLayout;

}

}

Теперь переходим в основную активность и пишем код:

package ru.alexanderklimov.stackviewdemo;

import android.os.Bundle;

import android.app.Activity;

import android.graphics.Color;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.StackView;

public class MainActivity extends Activity {

private int[] mColors = { Color.BLUE, Color.CYAN, Color.MAGENTA, Color.GREEN,Color.RED };

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

final StackView stackView = (StackView) findViewById(R.id.mStackView);

ColorAdapter colorAdapter = new ColorAdapter(this, mColors);

stackView.setAdapter(colorAdapter);

final Button previousButon = findViewById(R.id.previousButton);

previousButon.setOnClickListener(new OnClickListener() {

public void onClick(View view) {

stackView.showPrevious();

}

});

final Button nextButton = findViewById(R.id.nextButton);

nextButton.setOnClickListener(new OnClickListener() {

public void onClick(View view) {

stackView.showNext();

}

});

}

}

Для программного пролистывания стопки используются методы **showPrevious()** и showNext(). Попробуйте модифицировать пример, чтобы стопка состояла из изображений котов, и пришлите код мне.

