**Дрожь земли (анимация)**

Рассмотрим пример анимации, которая создаёт эффект дрожания. Подобный эффект можно применять в играх и даже в серьёзных приложениях для привлечения внимания.

Суть анимации состоит в циклическом повторении операции перемещения объекта за короткий промежуток времени, что создаёт иллюзию дрожания. Поэтому создадим папку /res/anim, в которой надо создать два файла: shake.xml и cycle\_7.xml.

**shake.xml**

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

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

android:duration="1000"

android:fromXDelta="0"

android:interpolator="@anim/cycle\_7"

android:toXDelta="10" />

**cycle\_7.xml**

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

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

android:cycles="7" />

В этом примере мы сдвигаем объект на 10 пикселей и повторяем подобное смещение 7 раз. Этого вполне достаточно для примера. Вы можете придумать свою анимацию.

Расположим на форме кнопку и заставим её дрожать при нажатии.

package ru.alexanderklimov.shake;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.animation.Animation;

import android.view.animation.AnimationUtils;

import android.widget.Button;

public class ShakeDemoActivity extends Activity {

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

final Button shakeButton = (Button) findViewById(R.id.button1);

final Animation shakeanimation = AnimationUtils.loadAnimation(this, R.anim.shake);

shakeButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

// TODO Auto-generated method stub

shakeButton.startAnimation(shakeanimation);

}

});

}

}

Ничего нам не мешает применить эффект дрожания и к целому экрану (разметка LinearLayout). Добавим пару строчек кода

final View linear = (View) findViewById(R.id.linear);

linear.startAnimation(shakeanimation);

// shakeButton.startAnimation(shakeanimation);

Как видите, ничего сложного здесь нет. Кстати, в документации к Android также используется подобный приём. Предположим, у нас есть два текстовых поля для ввода логина и пин-кода. Если пользователь вводит пин-код, длиной меньше или больше, чем четыре символа — окно ввода сотрясается. При правильном вводе пин-кода переходим на следующее окно.

final EditText editLogin = (EditText) this.findViewById(R.id.login);

final EditText editPin = (EditText) this.findViewById(R.id.pin);

shakeButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

// TODO Auto-generated method stub

// linear.startAnimation(shakeanimation);

// shakeButton.startAnimation(a);

// Копируем пин-код который ввел пользователь в переменную pass

String pass = editPin.getText().toString();

// Проверяю равна ли длина 4

if (pass.length() == 4) {

// Выводим всплывающее окно

Toast toast = Toast.makeText(getApplicationContext(),

"Добро пожаловать, " + editLogin.getText().toString(),

Toast.LENGTH\_SHORT);

toast.show();

// Создаем класс перехода

// Intent intent = new Intent();

// intent.setClass(ShakeDemoActivity.this,

// WelcomeAndroid.class);

// startActivity(intent);

// // Закрываем текущий Activity

// finish();

} else {

// Если пин-код неправильной длины, то применияем анимацию

editPin.startAnimation(shakeanimation);

// Выводем всплывающее окно "Wrong pin, must be 4 digits"

Toast toast = Toast.makeText(getApplicationContext(),

"Неверный пин-код, должно быть четыре цифры",

Toast.LENGTH\_SHORT);

toast.show();

// Очищаем форму ввода пин кода

editPin.setText("");

}

}

});

