метод setOnClickListener()

Button button = (Button) findViewById(R.id.button);

button.setOnClickListener()

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

}

});

Третий способ - интерфейс OnClickListener

public class MainActivity extends Activity implements OnClickListener { TextView tvOut;

Button btnOk; Button btnCancel;

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

@Override

public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.main);

// найдем View-элементы

tvOut = (TextView) findViewById(R.id.tvOut);

btnOk = (Button) findViewById(R.id.btnOk);

btnCancel = (Button) findViewById(R.id.btnCancel);

// присваиваем обработчик кнопкам

btnOk.setOnClickListener(this);

btnCancel.setOnClickListener(this);

}

@Override

public void onClick(View v) {

// по id определеяем кнопку, вызвавшую этот обработчик

switch (v.getId()) {

case R.id.btnOk:

// кнопка ОК

tvOut.setText("Нажата кнопка ОК");

break;

case R.id.btnCancel:

// кнопка Cancel

tvOut.setText("Нажата кнопка Cancel");

break;

}

}

}

ДЛИННОЕ НАЖАТИЕ

Кроме обычного щелчка, в Android есть особый вид нажатия на кнопку - продолжительное нажатие. Это событие происходит, когда пользователь нажимает и удерживает кнопку в течение одной секунды. Этот тип нажатия обрабатывается независимо от обычного щелчка.

Для обработки продолжительного нажатия нужно реализовать класс **View.OnLongClickListener** и передать его в метод **setOnLongClickListener()**. Класс **OnLongClickListener** имеет один обязательный метод **OnLongClick()**. В принципе это похоже на метод **OnClick()**, только имеет возвращаемое значение.

imageButton.setOnLongClickListener(new OnLongClickListener() {

@Override

public boolean onLongClick(View v) {

Toast toast = Toast.makeText(getApplicationContext(),

"Long Click",

Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

toast.show();

return false;

}

});

Коснись меня

Если вы внимательно понаблюдаете за поведением кнопки, то увидите, что код срабатывает в тот момент, когда вы отпускаете палец с кнопки. Для обычных приложений это вполне нормально, а для игр на скорость такой подход может оказаться слишком медленным. В подобных случаях лучше обрабатывать ситуацию не с нажатием кнопки, а с его касанием. В Android есть соответствующий слушатель **OnTouchListener()**:

Button button = (Button) findViewById(R.id.button);

button.setOnTouchListener(new OnTouchListener() {

@Override

public boolean onTouch(View v, MotionEvent event) {

if (event.getAction() == MotionEvent.ACTION\_DOWN) {

Toast.makeText(getApplicationContext(),

"Молодой человек, не прикасайтесь ко мне!",

Toast.LENGTH\_SHORT).show();

}

return false;

}

});

У метода **onTouch()** есть параметр **MotionEvent**, позволяющий более тонко определять касания экрана. Если произойдет событие, когда пользователь коснулся экрана, то ему будет соответствовать константа **ACTION\_DOWN**. Соответственно, если пользователь уберёт палец, то нужно использовать константу **ACTION\_UP**. Таким образом, можете расценивать щелчок кнопки как комбинацию двух событий - касания и отпускания.