**События в Android-приложении**

|  |
| --- |
| Обработка событий в Android-приложении  Нажатие кнопки в Android-приложении  **Пример 1**. Файл layout/main.xml  <?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" >  <LinearLayout  android:id="@+id/linearLayout1"  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  android:layout\_margin="30dp"  android:orientation="vertical" >  <TextView  android:id="@+id/tvOut"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center\_horizontal"  android:layout\_marginBottom="50dp"  android:text="TextView" >  </TextView>  <Button  android:id="@+id/btnOk"  android:layout\_width="100dp"  android:layout\_height="wrap\_content"  android:layout\_gravity="center\_horizontal"  android:text="OK" >  </Button>  <Button  android:id="@+id/btnCancel"  android:layout\_width="100dp"  android:layout\_height="wrap\_content"  android:layout\_gravity="center\_horizontal"  android:text="Cancel" >  </Button>  </LinearLayout>  </LinearLayout>  **Пример 1**. Файл MainActivity.java.  package org.ggpi.AppOnClick01;  import android.app.Activity;  import android.os.Bundle;  import android.view.View;  import android.view.View.OnClickListener;  import android.widget.Button;  import android.widget.TextView;  public class MainActivity extends Activity {    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);    // создаем обработчик нажатия  OnClickListener oclBtnOk = new OnClickListener() {  @Override  public void onClick(View v) {  // Меняем текст в TextView (tvOut)  tvOut.setText("Нажата кнопка ОК");  }  };    // присвоим обработчик кнопке OK (btnOk)  btnOk.setOnClickListener(oclBtnOk);  }  }  **Задание**. Добавьте обработчик нажатия кнопки **Cancel**. |

Источники

1. [Пример 1](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/16-urok-9-obrabotchiki-sobytij-na-primere-button.html). Отдельный обработчик на каждую кнопку
2. [Пример 2](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/17-urok-10-optimiziruem-realizatsiju-obrabotchikov.html). Общий обработчик на все кнопки

# [Обработчики событий на примере Button.](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/16-urok-9-obrabotchiki-sobytij-na-primere-button.html)

22 августа 2011

В этом уроке мы:  
  
- научимся обрабатывать нажатие кнопки и узнаем, что такое обработчик

Создадим проект:

**Project name**: P0091\_OnClickButtons  
**Build Target**: Android 2.3.3  
**Application name**: OnClickButtons  
**Package name**: ru.startandroid.develop.onclickbuttons  
**Create Activity**: MainActivity

В layout-файл **main.xml** напишем следующее и сохраним:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout      xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="horizontal">      <LinearLayout          android:id="@+id/linearLayout1"          android:layout\_width="match\_parent"          android:layout\_height="match\_parent"          android:layout\_margin="30dp"          android:orientation="vertical">          <TextView              android:id="@+id/tvOut"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:layout\_marginBottom="50dp"              android:text="TextView">          </TextView>          <Button              android:id="@+id/btnOk"              android:layout\_width="100dp"              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:text="OK">          </Button>          <Button              android:id="@+id/btnCancel"              android:layout\_width="100dp"              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:text="Cancel">          </Button>      </LinearLayout>  </LinearLayout> |

У нас есть TextView с текстом и две кнопки: OK и Cancel. Мы сделаем так, чтобы по нажатию кнопки менялось содержимое TextView. По нажатию кнопки OK – будем выводить текст: «Нажата кнопка ОК», по нажатию Cancel – «Нажата кнопка Cancel».

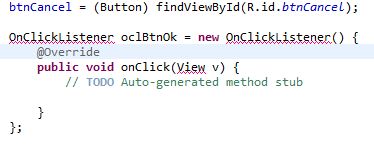
Открываем **MainActivity.java**. Описание объектов вынесем за пределы метода **onCreate**. Это сделано для того, чтобы мы могли из любого метода обращаться к ним. В onCreate мы эти объекты заполним с помощью уже пройденного нами метода**findViewById**. В итоге должен получиться такой код:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | public class MainActivity extends Activity {        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);        }  } |

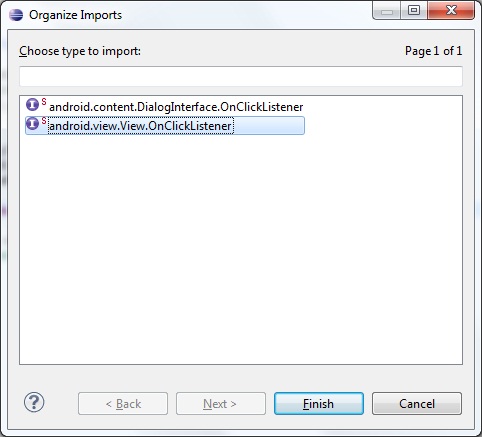
Обновляем секцию import (CTRL+SHIFT+O). Объекты tvOut, btnOk и btnCancel соответствуют View-элементам экрана и мы можем с ними работать. Нам надо научить кнопку реагировать на нажатие. Для этого у кнопки есть метод [setOnClickListener (View.OnClickListener l)](http://developer.android.com/reference/android/view/View.html" \l "setOnClickListener(android.view.View.OnClickListener)" \t "_blank). На вход подается объект с [интерфейсом](http://download.oracle.com/javase/tutorial/java/IandI/createinterface.html) [View.OnClickListener](http://developer.android.com/reference/android/view/View.OnClickListener.html" \t "_blank). Именно этому объекту кнопка поручит обрабатывать нажатия. Давайте создадим такой объект. Код продолжаем писать в onCreate:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | OnClickListener oclBtnOk = new OnClickListener() {      @Override      public void onClick(View v) {          // TODO Auto-generated method stub        }  }; |

Eclipse подчеркивает OnClickListener красной линией



т.к. пока не знает его. Необходимо обновить секцию import. Жмем CTRL+SHIFT+O, Eclipse показывает нам, что он знает два интерфейса с именем onClickListener и предлагает выбрать. Нам нужен View.OnClickListener, т.к. метод кнопки setOnClickListener принимает на вход именно его.



Итак, мы создали объект oclBtnOk, который реализует интерфейс View.OnClickListener. Объект содержит метод onClick – это как раз то, что нам нужно. Именно этот метод будет вызван при нажатии кнопки. Мы решили, что по нажатию будем выводить текст: «Нажата кнопка ОК» в TextView (tvOut). Реализуем это.

В методе onClick пишем:

|  |  |
| --- | --- |
| 1 | tvOut.setText("Нажата кнопка ОК"); |

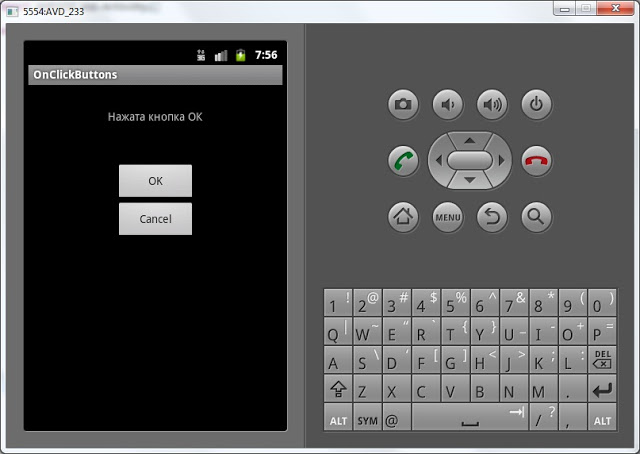
Обработчик нажатия готов. Осталось «скормить» его кнопке с помощью метода setOnClickListener.

|  |  |
| --- | --- |
| 1 | btnOk.setOnClickListener(oclBtnOk); |

В итоге должен получится такой код:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | public class MainActivity extends Activity {        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);            // создаем обработчик нажатия          OnClickListener oclBtnOk = new OnClickListener() {              @Override              public void onClick(View v) {                  // Меняем текст в TextView (tvOut)                  tvOut.setText("Нажата кнопка ОК");              }          };            // присвоим обработчик кнопке OK (btnOk)          btnOk.setOnClickListener(oclBtnOk);      }  } |

Все сохраняем и запускаем. Жмем на кнопку ОК и видим. Что текст изменился



Нажатие на Cancel пока ни к чему не приводит, т.к. для нее мы обработчик не создали и не присвоили. Давайте сделаем это аналогично, как для кнопки OK. Сначала мы создаем обработчик:

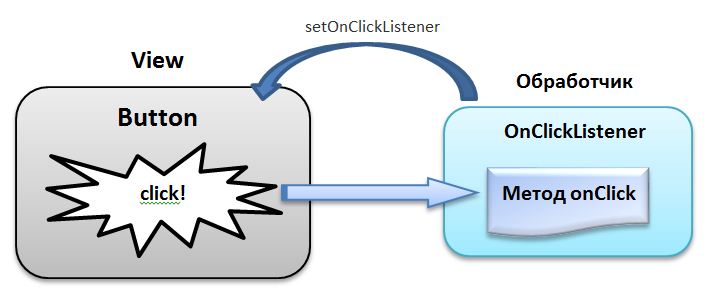
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | OnClickListener oclBtnCancel = new OnClickListener() {      @Override      public void onClick(View v) {          // Меняем текст в TextView (tvOut)          tvOut.setText("Нажата кнопка Cancel");      }  }; |

Потом присваиваем его кнопке:

|  |  |
| --- | --- |
| 1 | btnCancel.setOnClickListener(oclBtnCancel); |

Сохраняем, запускаем, проверяем. Обе кнопки теперь умеют обрабатывать нажатия.

Давайте еще раз проговорим механизм обработки событий на примере нажатия кнопки. Сама кнопка обрабатывать нажатия не умеет, ей нужен обработчик (его также называют слушателем - listener), который присваивается с помощью метода setOnClickListener. Когда на кнопку нажимают, обработчик реагирует и выполняет код из метода onClick. Это можно изобразить так:



Соответственно для реализации необходимо выполнить следующие шаги:

- создаем обработчик  
- заполняем метод onClick  
- присваиваем обработчик кнопке  
  
и система обработки событий готова.

# [Оптимизируем реализацию обработчиков.](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/17-urok-10-optimiziruem-realizatsiju-obrabotchikov.html)

25 августа 2011

В этом уроке мы:  
  
- научимся использовать один обработчик для нескольких View-элементов  
- научим Activity выступать в качестве обработчика

Создадим проект:  
**Project name**: P0101\_Listener  
**Build Target**: Android 2.3.3  
**Application name**: Listener  
**Package name**: ru.startandroid.develop.listener  
**Create Activity**: MainActivity

Будем работать с теми же View, что и в предыдущем уроке. Код для **main.xml**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout      xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_height="match\_parent"      android:layout\_width="match\_parent"      android:orientation="horizontal">      <LinearLayout          android:id="@+id/linearLayout1"          android:layout\_height="match\_parent"          android:orientation="vertical"          android:layout\_width="match\_parent"          android:layout\_margin="30dp">          <TextView              android:layout\_width="wrap\_content"              android:text="TextView"              android:layout\_height="wrap\_content"              android:id="@+id/tvOut"              android:layout\_gravity="center\_horizontal"              android:layout\_marginBottom="50dp">          </TextView>          <Button              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:id="@+id/btnOk"              android:text="OK"              android:layout\_width="100dp">          </Button>          <Button              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:id="@+id/btnCancel"              android:text="Cancel"              android:layout\_width="100dp">          </Button>      </LinearLayout>  </LinearLayout> |

### Один обработчик для двух кнопок

Итак, у нас есть TextView с текстом и две кнопки. Как и на прошлом уроке, мы сделаем так, чтобы по нажатию кнопки менялось содержимое TextView. По нажатию кнопки OK – будем выводить текст: «Нажата кнопка ОК», по нажатию Cancel – «Нажата кнопка Cancel». Но сейчас мы сделаем это с помощью **одного обработчика**, который будет обрабатывать нажатия **для обеих кнопок**.

Напомню механизм обработки событий на примере нажатия кнопки. Сама кнопка обрабатывать нажатия не умеет, ей нужен обработчик (listener), который присваивается с помощью метода setOnClickListener. Когда на кнопку нажимают, обработчик реагирует и выполняет код из метода onClick.

Соответственно для реализации необходимо выполнить следующие шаги:  
- создаем обработчик  
- заполняем метод onClick  
- присваиваем обработчик кнопке

В нашем случае мы будем присваивать один обработчик обеим кнопкам, а внутри обработчика надо будет определять, какая именно кнопка была нажата.

Подготовим объекты и создадим обработчик:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | public class MainActivity extends Activity {       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);         // создание обработчика       OnClickListener oclBtn = new OnClickListener() {         @Override         public void onClick(View v) {           // TODO Auto-generated method stub           }       };       }   } |

Давайте заполнять метод onClick. На вход ему подается объект класса **View**, это как раз то, что нам нужно. Это View на которой произошло нажатие и которая вызвала обработчик. Т.е. в нашем случае это будет либо кнопка OK либо Cancel. Нам осталось узнать **ID** этой **View** и сравнить его с нашими **R.id.btnOk** и **R.id.btnCancel**, чтобы определить какая именно это кнопка. Чтобы получить ID какой-либо View, используется метод [getId](http://developer.android.com/reference/android/view/View.html" \l "getId()" \t "_blank). Для перебора результатов используем java-оператор [switch](http://download.oracle.com/javase/tutorial/java/nutsandbolts/switch.html" \t "_blank).

Реализация метода onClick:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | 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;     }   } |

Если сейчас запустить приложение и проверить, то ничего не произойдет. Обработчик то мы создали, но не присвоили его кнопкам. Обеим кнопкам присваиваем один и тот же обработчик:

|  |  |
| --- | --- |
| 1  2 | btnOk.setOnClickListener(oclBtn);  btnCancel.setOnClickListener(oclBtn); |

 Вот теперь можем запускать и проверять, все должно работать.

Как вы понимаете, **один обработчик** может быть присвоен не двум, а **любому количеству** кнопок. И не только кнопкам. У остальных View-элементов тоже есть различные события, которые нуждаются в обработчиках. В дальнейшем мы еще будем с ними работать. А сейчас важно понять схему, как происходит обработка событий.

Отличие способа реализации на этом уроке от прошлого урока в том, что сейчас мы создали **один объект-обработчик для обеих кнопок**, а на прошлом уроке - **два объекта, по одному каждой кнопке**. Есть правило – чем меньше объектов вы создаете, тем лучше, т.к. под каждый объект выделяется память, а это достаточно ограниченный ресурс, особенно для телефонов. Поэтому создавать один обработчик для нескольких View это правильнее с точки зрения оптимизации. К тому же кода становится меньше и читать его удобнее.

Есть еще один способ создания обработчика, который вовсе не потребует создания объектов. Будет использоваться уже созданный объект – Activity

### Activity, как обработчик

Кнопка присваивает себе обработчика с помощью метода [setOnClickListener (View.OnClickListener l)](http://developer.android.com/reference/android/view/View.html" \l "setOnClickListener(android.view.View.OnClickListener)" \t "_blank). Т.е. подойдет любой объект с [интерфейсом](http://download.oracle.com/javase/tutorial/java/IandI/createinterface.html) [View.OnClickListener](http://developer.android.com/reference/android/view/View.OnClickListener.html" \t "_blank). Почему бы классу Activity не быть таким объектом? Мы просто укажем, что Activity-класс реализует интерфейс View.OnClickListener и заполним метод onCreate.

Создадим для этого новый проект:  
**Project name**: P0102\_ActivityListener  
**Build Target**: Android 2.3.3  
**Application name**: ActivityListener  
**Package name**: ru.startandroid.develop.activitylistener  
**Create Activity**: MainActivity

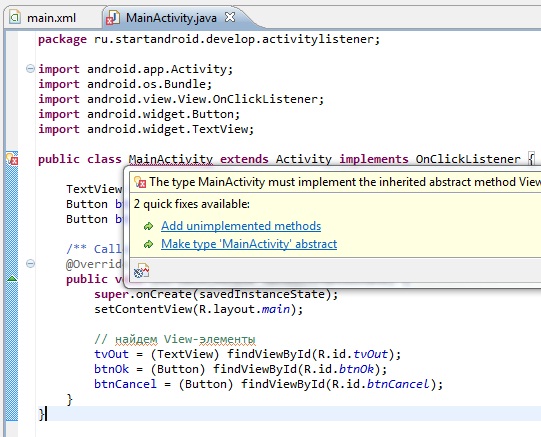
Экран снова возьмем тот же самый:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout      xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_height="match\_parent"      android:layout\_width="match\_parent"      android:orientation="horizontal">      <LinearLayout          android:id="@+id/linearLayout1"          android:layout\_height="match\_parent"          android:orientation="vertical"          android:layout\_width="match\_parent"          android:layout\_margin="30dp">          <TextView              android:layout\_width="wrap\_content"              android:text="TextView"              android:layout\_height="wrap\_content"              android:id="@+id/tvOut"              android:layout\_gravity="center\_horizontal"              android:layout\_marginBottom="50dp">          </TextView>          <Button              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:id="@+id/btnOk"              android:text="OK"              android:layout\_width="100dp">          </Button>          <Button              android:layout\_height="wrap\_content"              android:layout\_gravity="center\_horizontal"              android:id="@+id/btnCancel"              android:text="Cancel"              android:layout\_width="100dp">          </Button>      </LinearLayout>  </LinearLayout> |

Подготовим объекты и добавим реализацию интерфейса (implements onClickListener)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | 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);     }   } |

OnClickListener подчеркнут красным, т.к. его нет в импорте. Поэтому CTRL+SHIFT+O и выбираем View.OnClickListener.  
  
Теперь Eclipse ругается на класс MainActivity. Это происходит потому, что для класса **прописан интерфейс**, но **нет реализации методов** этого интерфейса. Исправим это с помощью Eclipse. Наведите курсор на **MainAcivity** и выберите **Add unimplemented methods**



Eclipse добавит знакомый нам метод **onClick**. Только теперь этот метод будет реализован в Activity, а не в отдельном объекте-обработчике. Соответственно Activity и будет выступать обработчиком.

Заполним метод точно так же как и раньше. Ничего не изменилось. Ему на вход так же подается View (на которой произошло событие), по Id мы определим, какая именно эта View и выполним соответствующие действия:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | 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;     }   } |

Осталось в методе onCreate присвоить обработчик кнопкам. Это будет объект this, т.е. текущий объект MainActivity.

|  |  |
| --- | --- |
| 1  2 | btnOk.setOnClickListener(this);  btnCancel.setOnClickListener(this); |

При такой реализации мы не создали ни одного лишнего объекта (Activity создается в любом случае) и затраты памяти минимальны, это рекомендуемый метод. Но, возможно, такой способ покажется сложным и непонятным, особенно если мало опыта в объектно-ориентированном программировании. В таком случае используйте ту реализацию, которая вам понятна и удобна. А со временем и опытом понимание обязательно придет.

Полный код:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | 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;       }     }     } |

### Самая простая реализация обработчика

Есть еще один способ реализации. В layout-файле (main.xml) при описании кнопки пишем:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | <Button      android:id="@+id/btnStart"      android:layout\_width="wrap\_content"      android:layout\_height="wrap\_content"      android:onClick="onClickStart"      android:text="start">  </Button> |

Т.е. используем атрибут [onClick](http://developer.android.com/reference/android/R.attr.html" \l "onClick" \t "_blank). В нем указываем имя метода из Activity. Этот метод и сработает при нажатии на кнопку.

Далее, добавляем этот метод в Activity (MainActivity.java). Требования к методу: public, void и на вход принимает View:

|  |  |
| --- | --- |
| 1  2  3 | public void onClickStart(View v) {    // действия при нажатии на кнопку  } |

В методе прописываете необходимые вам действия, и они будут выполнены при нажатии кнопки.