**Оптимизируем реализацию обработчиков.**

Создадим проект:

Project name: P0101\_Listener

Build Target: Android 2.3.3

Application name: Listener

Package name: ru.startandroid.develop.listener

Create Activity: MainActivity

Код для main.xml:

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

<LinearLayout xmlns:android=[*"http://schemas.android.com/apk/res/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*](mailto:@+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*](mailto:@+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*](mailto:@+id)*/btnOk"* android:text=*"OK"* android:layout\_width=*"100dp"*>

</Button>

<Button android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center\_horizontal"* android:id=[*"@+id*](mailto:@+id)*/btnCancel"* android:text=*"Cancel"* android:layout\_width=*"100dp"*>

</Button>

</LinearLayout>

</LinearLayout>

**Один обработчик для двух кнопок ( метод – 1 )**

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:

public class MainActivity extends Activity {

TextView tvOut;

Button btnOk;

Button btnCancel;

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

[@Override](mailto:@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](mailto:@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 наш ем случае это будет либо кнопка OK либо Cancel. Нам осталось узнать ID этой View и сравнить его с наш ими R.id.btnOk и R.id.btnCancel, чтобы определить какая именно это кнопка. Чтобы получить ID какой-либо View, используется метод getId. Для перебора результатов используем java-оператор switch.

Реализация метода onClick:

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:

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 (метод – 2)**

Кнопка присваивает себе обработчика с помощью метода setOnClickListener (View.OnClickListener). Т.е. подойдет любой объект с интерфейсом View.OnClickListener. 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:

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

<LinearLayout xmlns:android=[*"http://schemas.android.com/apk/res/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*](mailto:@+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*](mailto:@+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*](mailto:@+id)*/btnOk"*

android:text=*"OK"*

android:layout\_width=*"100dp"*>

</Button>

<Button android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center\_horizontal"*

android:id=[*"@+id*](mailto:@+id)*/btnCancel"*

android:text=*"Cancel"*

android:layout\_width=*"100dp"*>

</Button>

</LinearLayout>

</LinearLayout>

Подготовим объекты и добавим реализацию интерфейса (implements onClickListener)

public class MainActivity extends Activity implements OnClickListener {

TextView tvOut;

Button btnOk;

Button btnCancel;

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

[@Override](mailto:@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 ALT+ENTER выбираем View.OnClickListener.

Теперь AS ругается на класс MainActivity. Это происходит потому, что для класса прописан интерфейс, но нет реализации методов этого интерфейса. Исправим это с помощью среды программирования. Наведите курсор на MainAcivity и выберите Add unimplemented methods



Android Studio 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:

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.

btnOk.setOnClickListener(this);

btnCancel.setOnClickListener(this);

При такой реализации мы не создали ни одного лиш него 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:

public class MainActivity extends Activity implements OnClickListener { TextView tvOut;

Button btnOk; Button btnCancel;

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

[@Override](mailto:@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](mailto:@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 (метод – 3)**

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

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

<Button android:id=[*"@+id*](mailto:@+id)*/btnStart"*

android:layout\_width=*"wrap\_content"* android:layout\_height=*"wrap\_content"* android:onClick=*"onClickStart"* android:text=*"start"*>

</Button>

Т.е. используем атрибут onClick. 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:

public void onClickStart(View v) {

// действия при нажатии на кнопку

}

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