**Запуск второй Activity**

В предыдущей теме мы создали приложение с использованием визуальных элементов: добавили к activity текстовое поле для ввода сообщения и кнопку. Теперь продолжим предыдущий проект и добавим запуск новой activity по нажатию на кнопку.

**Обработка нажатия кнопки**

Для подключения обработчика кнопки откроем файл **activity\_main.xml** и добавим в элемент **Button** атрибут **android:onClick**:

<?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" >

    <EditText android:id="@+id/edit\_message"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_weight="1"

        android:hint="Введите сообщение" />

    <Button

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:onClick="sendMessage"

        android:text="Отправить" />

</LinearLayout>

Значение "**sendMessage**", присвоенное атрибуту **android:onClick,** представляет собой имя метода, определенного в классе activity, который вызывает система при нажатии пользователем на кнопку. Далее мы определим этот метод.

**Создание объекта Intent**

Затем перейдем к классу **MainActivity**, который находится в проекте в папке **app/src/main/java/[название\_пакета]**, и изменим этот класс следующим образом (первая строчка остается неизменной):

import android.content.Intent; // подключаем класс Intent

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View; // подключаем класс View для обработки нажатия кнопки

import android.widget.EditText; // подключаем класс EditText

public class MainActivity extends AppCompatActivity {

    public final static String EXTRA\_MESSAGE = "EXTRA\_MESSAGE";

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

    }

    // Метод обработки нажатия на кнопку

    public void sendMessage(View view) {

        // действия, совершаемые после нажатия на кнопку

        // Создаем объект Intent для вызова новой Activity

        Intent intent = new Intent(this, DisplayMessageActivity.class);

        // Получаем текстовое поле в текущей Activity

        EditText editText = (EditText) findViewById(R.id.edit\_message);

        // Получае текст данного текстового поля

        String message = editText.getText().toString();

        // Добавляем с помощью свойства putExtra объект - первый параметр - ключ,

        // второй параметр - значение этого объекта

        intent.putExtra(EXTRA\_MESSAGE, message);

        // запуск activity

        startActivity(intent);

    }}

Далее для запуска второй activity необходим объект Intent. Объект **Intent** представляет некоторую задачу приложения, которую надо выполнить (например, запуск activity):

Intent intent = new Intent(this, DisplayMessageActivity.class);

Конструктор этого объекта принимает два параметра:

* Первый параметр представляет контекст - объект **Context** (ключевое слово this употребляется здесь, так как класс **MainActivity** является подклассом класса Context)
* Вторым параметром идет класс компонента, которому мы передаем объект Intent. В качестве него будет выступать объект **DisplayMessageActivity**, который мы создадим чуть позже

Внутри метода **sendMessage()** используем метод **findViewById**, чтобы получить элемент **EditText** и передать значение его текста в объект intent:

EditText editText = (EditText) findViewById(R.id.edit\_message);

String message = editText.getText().toString();

Затем полученный из текстового поля текст передается в запускаемую activity:

intent.putExtra(EXTRA\_MESSAGE, message);

Параметр "**EXTRA\_MESSAGE**" указывает на ключ передаваемых данных. То есть мы можем в новую activity передать множество данных, и чтобы их можно было разграничить, для них устанавливается ключ. В данном случае ключ **EXTRA\_MESSAGE** - эта обычная строка "**EXTRA\_MESSAGE**":

public final static String EXTRA\_MESSAGE = "EXTRA\_MESSAGE";

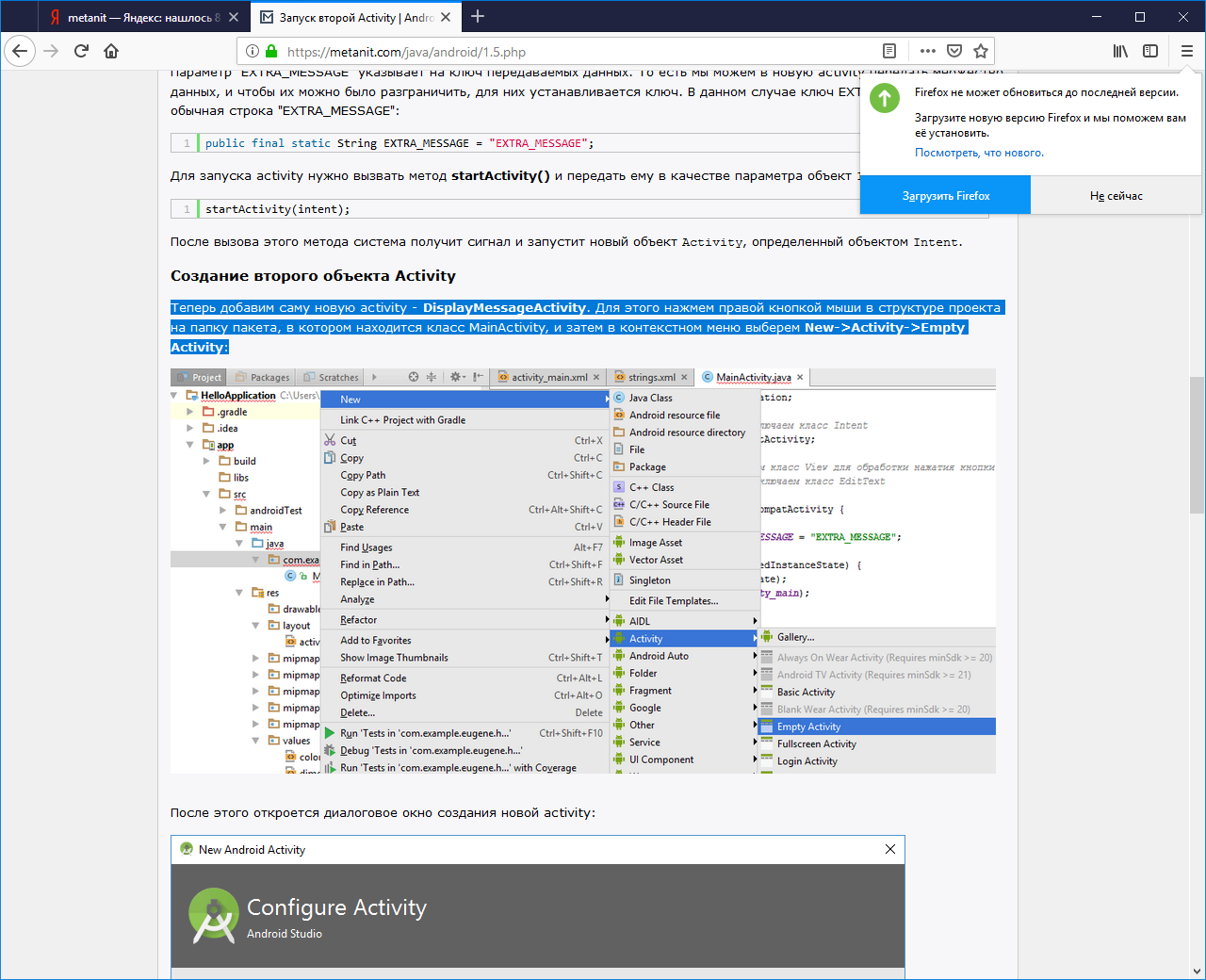
Для запуска activity нужно вызвать метод **startActivity()** и передать ему в качестве параметра объект Intent:

startActivity(intent);

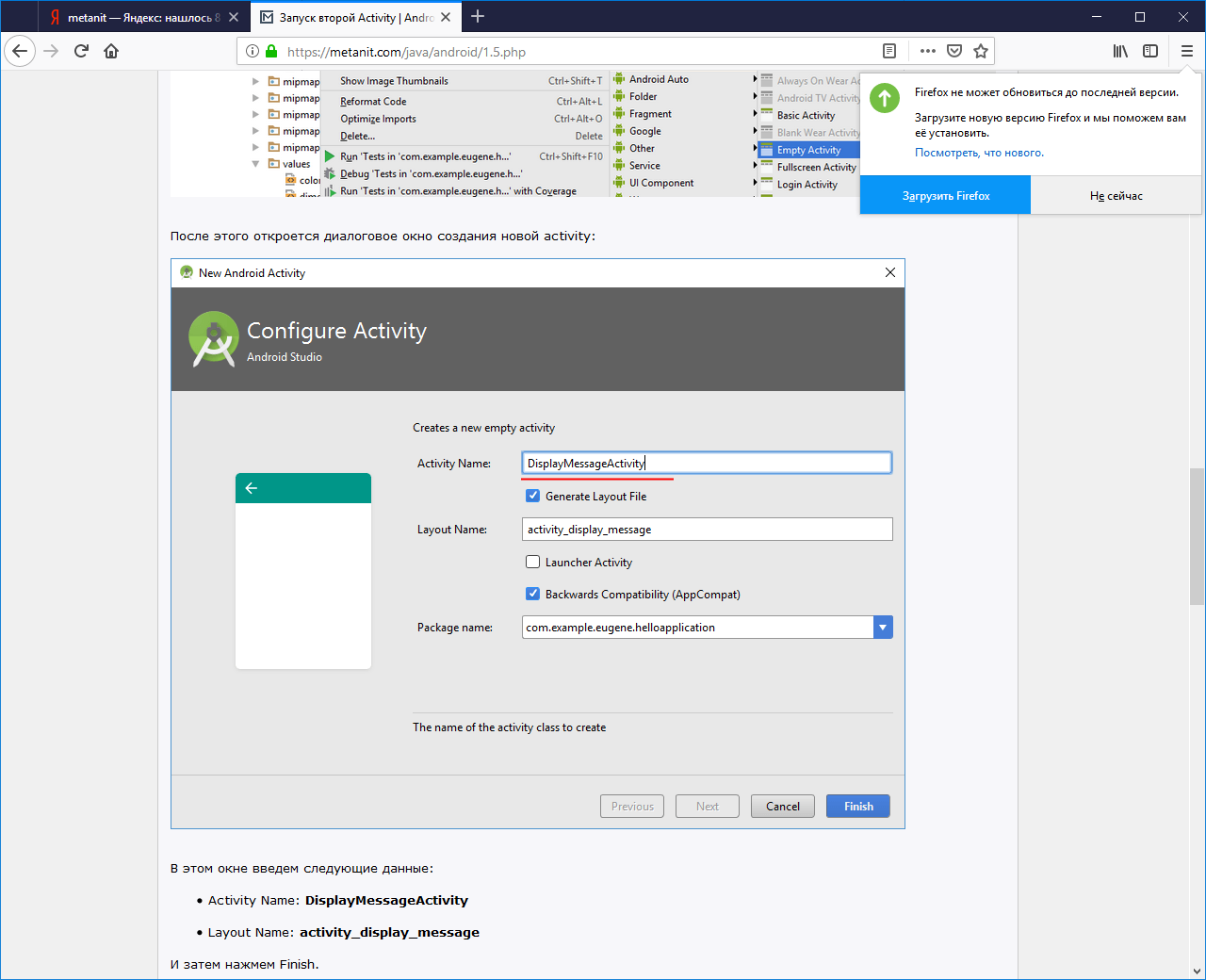
После вызова этого метода система получит сигнал и запустит новый объект Activity, определенный объектом Intent.

**Создание второго объекта Activity**

Теперь добавим саму новую activity - **DisplayMessageActivity**. Для этого нажмем правой кнопкой мыши в структуре проекта на папку пакета, в котором находится класс MainActivity, и затем в контекстном меню выберем **New->Activity->Empty Activity**:



После этого откроется диалоговое окно создания новой activity:

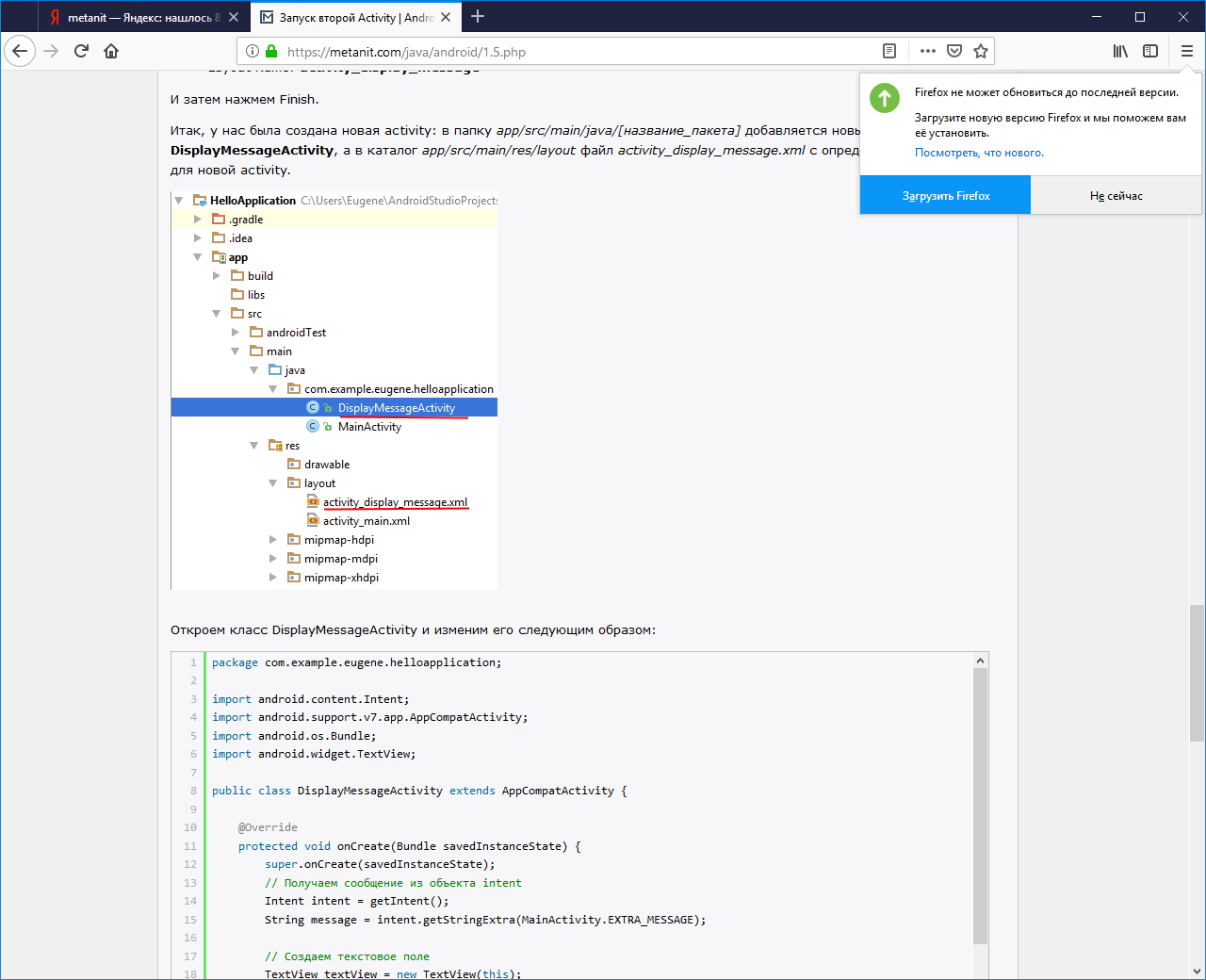


В этом окне введем следующие данные:

* Activity Name: **DisplayMessageActivity**
* Layout Name: **activity\_display\_message**

И затем нажмем Finish.

Итак, у нас была создана новая activity: в папку **app/src/main/java/[название\_пакета]** добавляется новый класс **DisplayMessageActivity**, а в каталог **app/src/main/res/layout** файл activity\_display\_message.xml с определением интерфейса для новой activity.



Откроем класс DisplayMessageActivity и изменим его следующим образом:

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.widget.TextView;

public class DisplayMessageActivity extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        // Получаем сообщение из объекта intent

        Intent intent = getIntent();

        String message = intent.getStringExtra(MainActivity.EXTRA\_MESSAGE);

        // Создаем текстовое поле

        TextView textView = new TextView(this);

        textView.setTextSize(40);

        textView.setText(message);

        // Устанавливаем текстовое поле в системе компоновки activity

        setContentView(textView);

    }

}

Также, как и в MainActivity (и в других activity), создание текущей activity происходит в методе onCreate(). Все подклассы Activity должны реализовать метод onCreate, так как система вызывает его при создании новой activity. Именно в этом методе задается компоновка нового объекта activity с помощью метода setContentView и именно здесь происходит начальная настройка компонентов.

**Вывод сообщения на экран**

Итак, как мы помним, вначале мы передавали из нашей первой и главной activity в DisplayMessageActivity текстовое сообщение, которое было введено в текстовое поле. Теперь получим его в DisplayMessageActivity и выведем на экран

Каждый объект Activity вызывается объектом Intent. Мы можем получить вызывающий объект Intent с помощью метода getIntent и, таким образом, получить переданные с ним данные.

Intent intent = getIntent();

String message = intent.getStringExtra(MainActivity.EXTRA\_MESSAGE);

Для вывода сообщения на экран создадим виджет TextView и с помощью его свойства setText установим его текст. А затем добавим новый виджет в DisplayMessageActivity при помощи метода setContentView.

TextView textView = new TextView(this);

textView.setTextSize(40);

textView.setText(message);

setContentView(textView);

После запуска приложения на эмуляторе мы увидим опять же текстовое поле с кнопкой, но после ввода текста и нажатия на кнопку будет запущена новая activity и отобразится другой экран с введенным ранее в текстовое поле сообщением.

