Android-studio

Editing line numbers:

Right click on the left bar space next to the code, will open a menu, choose the “show line numbers” option.

Editing code folding:

To remove the code folding, in order to see easily the whole code:

File-> settings-> Editor-> code folding->

Now uncheck all the checked options

Accessing the Key-map list:

Help-> Key-map

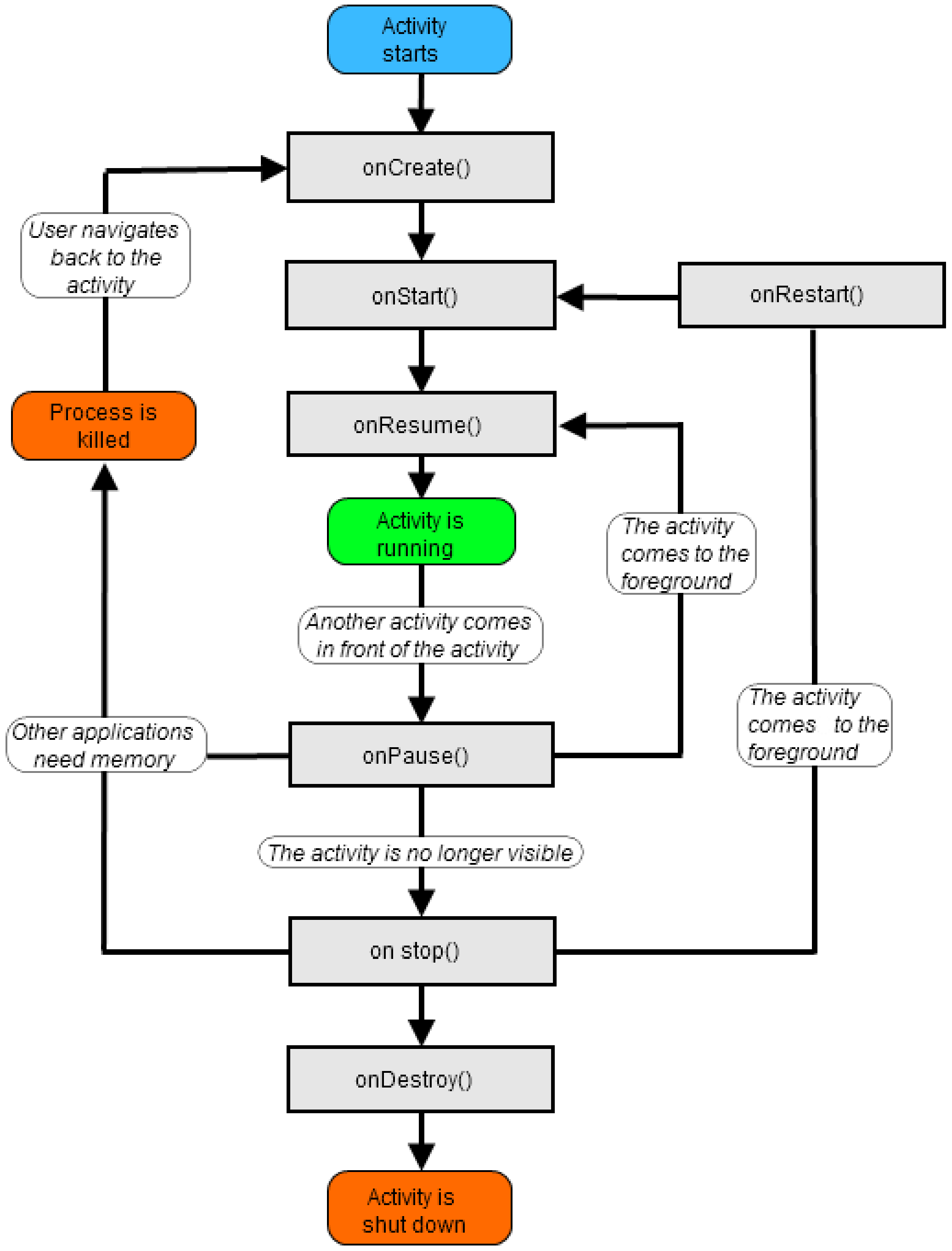
Adding activity to the project:

Right click on the java folder in the files window ->New-> activity-> blank activity.

If the activity that we create is the main activity we have to choose the “Launcher Activity” option.

(Launcher Activity - is the activity that will run every time the user runs the app)

Activity’s life cycle


Activity starts
onCreate()
User navigates back to the activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
The activity comes to the foreground
Another activity comes in front of the activity
The activity comes to the foreground
Other applications need memory
onPause()
The activity is no longer visible
on stop()
onDestroy()
Activity is shut down

Before the activity runs this 3 methods are activated:

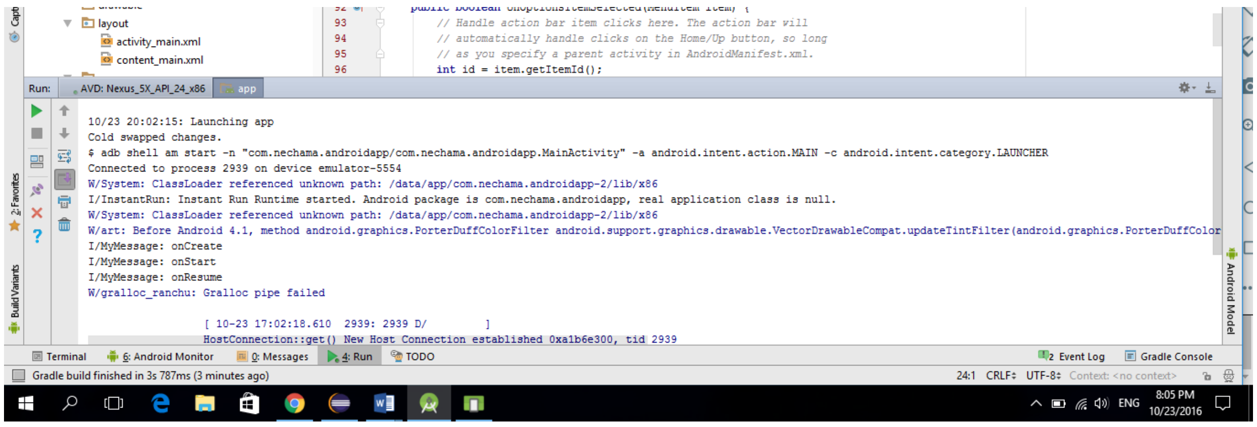
onCraete, onStart, onResume.

The way to kill an activity is with the finish() method. When we call from one activity to another activity, by default the current activity will call her "onPause" activity, so if we will add inside it finish(), the current activity will be killed when the next called activity is shown.

Printing log messages to the console window:

```java
package com.anna.androidapp;

import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.util.Log;

//every activity in android inherits from the Activity (AppCompatActivity inherits
Activity)
public class MainActivity extends AppCompatActivity {

    private static final String TAG = "MyMessage";


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //to the Log.i we sent 2 parameters: a tag, and the text that we want to print
        Log.i(TAG, "onCreate");
    }

    @Override
    protected void onStart() {
        super.onStart();
        Log.i(TAG, "onStart");
    }


    @Override
    protected void onResume() {
        super.onResume();
        Log.i(TAG, "onResume");
    }


    @Override
    protected void onPause() {
        super.onPause();
        Log.i(TAG, "onPause");
    }


    @Override
    protected void onStop() {
        super.onStop();
        Log.i(TAG, "onStop");
    }


    @Override
    protected void onRestart() {
        super.onRestart();
        Log.i(TAG, "onRestart");
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        Log.i(TAG, "onDestroy");
    }

    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        Log.i(TAG, "onSaveInstanceState");
    }

    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        Log.i(TAG, "onRestoreInstanceState");
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

when we run the application we will see the printed messages in the console window:



```
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
```

```
10/23 20:02:15: Launching app
Cold swapped changes.
$ adb shell am start -n "com.nechama.androidapp/com.nechama.androidapp.MainActivity" -a android.intent.action.MAIN -c android.intent.category.LAUNCHER
Connected to process 2939 on device emulator-5554
W/System: ClassLoader referenced unknown path: /data/app/com.nechama.androidapp-2/lib/x86
I/InstantRun: Instant Run Runtime started. Android package is com.nechama.androidapp, real application class is null.
W/System: ClassLoader referenced unknown path: /data/app/com.nechama.androidapp-2/lib/x86
W/art: Before Android 4.1, method android.graphics.PorterDuffColorFilter android.support.graphics.drawable.VectorDrawableCompat.updateTintFilter(android.graphics.PorterDuffColor
I/MyMessage: onCreate
I/MyMessage: onStart
I/MyMessage: onResume
W/gralloc_ranchu: Gralloc pipe failed

                 [ 10-23 17:02:18.610  2939: 2939 D/         ]
                 HostConnection::get() New Host Connection established 0xa1b6e300, tid 2939
```

Gradle build finished in 3s 787ms (3 minutes ago)

<u>“string.xml” tools</u>

When we create our app’s interface, and want to set the text that the user will see, we can just enter the content to the “android:text” attribute of the component.

But a better way to set the text is to declare it in a separate file that is called “string.xml” (is under “res” -> “values”).

In the “string.xml” file we will give each string name and content.

For example: `<string name="SignInTitle">Sign In</string>`

And in the component’s attribute we will point to this resource in this way:

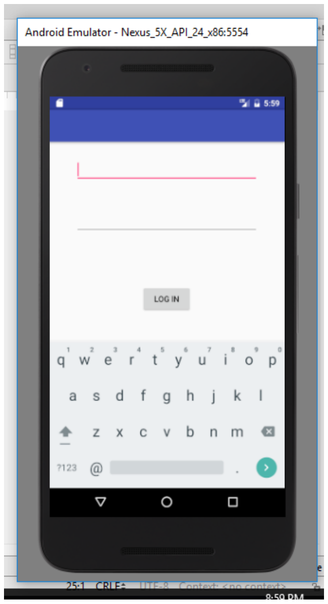
```
android:text="@string/SignInTitle"
```

<u>This is the code for a full app:</u>

```
***** Main
Activity.java
```

```
package your.package.name


import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;


public class MainActivity extends AppCompatActivity {


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }


    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();


        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }


        return super.onOptionsItemSelected(item);
    }
}




***** activity_main.xml


<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="glwhitney.info.nbvideos11_12.MainActivity">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="@string/SignInTitle"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:singleLine="true"/>


    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textEmailAddress"
        android:ems="10"
        android:id="@+id/editText"
        android:layout_below="@+id/textView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="36dp"
        android:width="320dp"/>


    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        android:ems="10"
        android:id="@+id/editText2"
        android:layout_marginTop="45dp"
        android:layout_below="@+id/editText"
        android:layout_alignLeft="@+id/editText"
        android:layout_alignStart="@+id/editText"
        android:width="320dp"/>


    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/SignInButtonText"
        android:id="@+id/button"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />
</RelativeLayout>


***** strings.xml


<resources>
    <string name="app_name">NB Videos 11_12</string>
    <string name="title_activity_main">MainActivity</string>



    <string name="action_settings">Settings</string>
    <string name="SignInTitle">Sign In</string>
    <string name="SignInButtonText">Log In</string>
</resources>
```

<u>This is the app on runtime:</u>



Creating a EditText with a place holder

```
<EditText


    android:hint="my text"
    android:id="@+id/textView6" />
```

```
The EditText will contain by default the “my text” string,
that is a place holder, and when the user starts to enter his
input, the place holder string disappears.
If the user deleats all the inputthat he
```

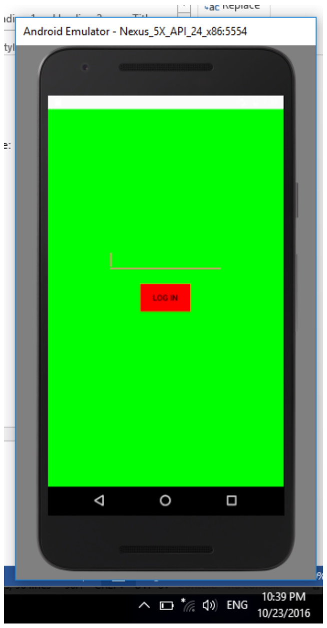
Creating a user interface by code

In this example the only code that we have wrote to this app is the following code:

(No xml activity code was edited, everything is with java code)

```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.RelativeLayout;
import android.widget.Button;
import android.graphics.Color;
import android.widget.EditText;
import android.content.res.Resources;
import android.util.TypedValue;

//every activity in android inherits from the Activity (AppCompatActivity inherits
Activity)
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //creating a Layout and setting it's background color
        RelativeLayout MyLayout = new RelativeLayout(this);
        MyLayout.setBackgroundColor(Color.GREEN);

        //creating a Button
        Button redButton = new Button(this);
        redButton.setText("Log In");
        redButton.setBackgroundColor(Color.RED);

        //creating an EditText that will be used to enter input by the user
        EditText username = new EditText(this);

       redButton.setId(1);
       //username.setId(2);

        //here we set the height and width of the button and EditText to match their
content
        RelativeLayout.LayoutParams buttonDetails = new RelativeLayout.LayoutParams(
                RelativeLayout.LayoutParams.WRAP_CONTENT,
                RelativeLayout.LayoutParams.WRAP_CONTENT
        );
        RelativeLayout.LayoutParams usernameDetails = new RelativeLayout.LayoutParams(
                RelativeLayout.LayoutParams.WRAP_CONTENT,
                RelativeLayout.LayoutParams.WRAP_CONTENT
        );

        //Give rules to position the
        usernameDetails.addRule(RelativeLayout.ABOVE, redButton.getId());
        usernameDetails.addRule(RelativeLayout.CENTER_HORIZONTAL);
        usernameDetails.setMargins(0,0,0,50);

        //Give rules to position the button un the middle of the layout
        buttonDetails.addRule(RelativeLayout.CENTER_HORIZONTAL);
        buttonDetails.addRule(RelativeLayout.CENTER_VERTICAL);


        //setting the EditText size by dp casted to px ("setWidth" takes only px)
        Resources r = getResources();
        int px = (int) TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP,200,
                r.getDisplayMetrics()
        );

        username.setWidth(px);

        //Add widget to layout(button and EditText are now children of layout)
        MyLayout.addView(redButton, buttonDetails);
        MyLayout.addView(username, usernameDetails);

        //Set the "activity_main" content to the layout that we created here by code
        setContentView(MyLayout);

    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

when we run the app the result will be:

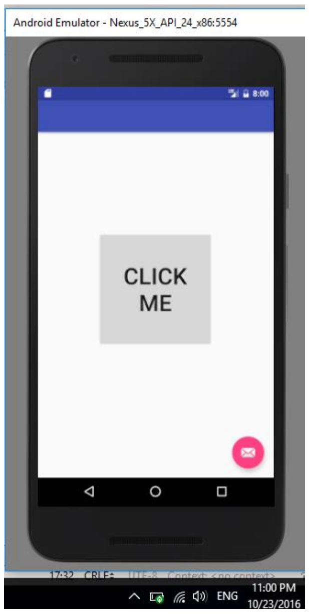


Add event onclick

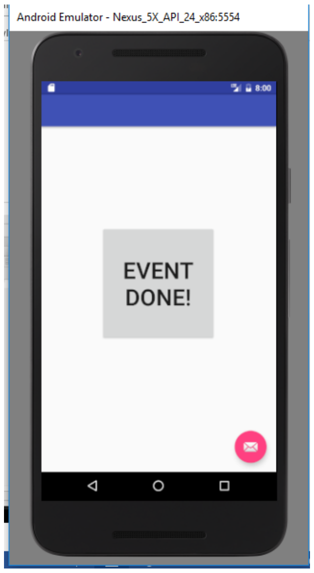
We create a project that onclick will change the button's default content to different content

```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Button;
import android.view.View;
import android.widget.TextView;

//every activity in android inherits from the Activity (AppCompatActivity inherits
Activity)
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

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

        b.setOnClickListener(

//an event that is activated in a regular click
b.setOnClickListener(

        //this is the event listner
        new Button.OnClickListener(){

            //this is the callback method that will be activated by the event
            public void onClick(View v){
                TextView buckysText = (TextView)findViewById(R.id.b);
                buckysText.setText("event Done!");
            }
        }
);
//an event that is activated in a long click (in a long click event we must to return
a bool value)
b.setOnLongClickListener(
        new Button.OnLongClickListener(){
            public boolean onLongClick(View v){
                TextView buckysText = (TextView)findViewById(R.id.b);
                buckysText.setText("long click!");

                //whenever we return true in a long click event, the callback function
will be activated
                return true;
            }
        }
);
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
}
```

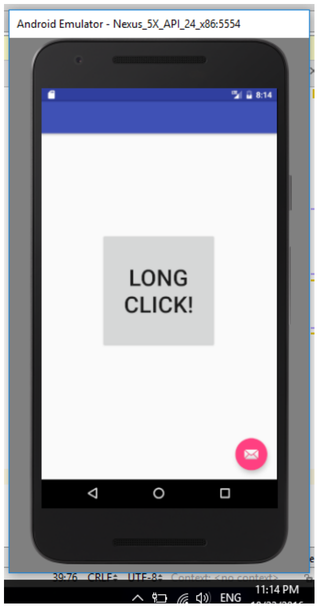
The app activity before clicking the button:



The app activity after clicking the button:



The app activity after a long click on the button:

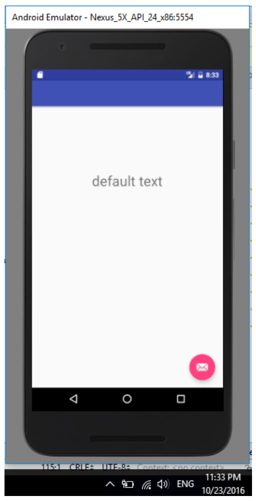


Gesture events

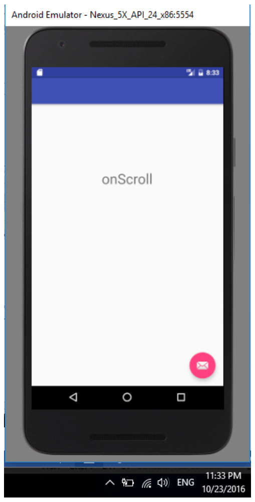
Gesture events are different touch screen events, in this app we changed the textview content in every event to a different content.

```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;
import android.view.MotionEvent;
import android.view.GestureDetector;
import android.support.v4.view.GestureDetectorCompat;

//every activity in android inherits from the Activity (AppCompatActivity inherits
Activity)
//for every Gesture event we have to implement a special matching interface
public class MainActivity extends AppCompatActivity implements
GestureDetector.OnGestureListener,
GestureDetector.OnDoubleTapListener {
        private TextView t;
        private GestureDetectorCompat gestureDetector;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);

                t = (TextView) findViewById(R.id.t);
                this.gestureDetector = new GestureDetectorCompat(this, this);
                gestureDetector.setOnDoubleTapListener(this);
                }
        ///////// OnGestureListener METHODS //////////
        @Override
        public boolean onSingleTapConfirmed(MotionEvent e) {
                t.setText("onSingleTapConfirmed");
                return true;
                }

        @Override
        public boolean onDoubleTap(MotionEvent e) {
                t.setText("onDoubleTap");
                return true;
                }

        @Override
        public boolean onDoubleTapEvent(MotionEvent e) {
                t.setText("onDoubleTapEvent");
                return true;
                }

        @Override
        public boolean onDown(MotionEvent e) {
                t.setText("onDown");
                return true;
                }

        @Override
        public void onShowPress(MotionEvent e) {
                t.setText("onShowPress");

                }

        @Override
        public boolean onSingleTapUp(MotionEvent e) {
                t.setText("onSingleTapUp");
                return true;
                }

        @Override
        public boolean onScroll(MotionEvent e1, MotionEvent e2, float distanceX, float
distanceY) {
                t.setText("onScroll");
                return true;
                }

        @Override
        public void onLongPress(MotionEvent e) {
                t.setText("onLongPress");

                }

        @Override
        public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX, float
velocityY) {
                t.setText("onFling");
                return true;
                }

        ///////// OnDoubleTapListener METHODS //////////


        @Override
        public boolean onTouchEvent(MotionEvent event) {
                this.gestureDetector.onTouchEvent(event);
                return super.onTouchEvent(event);
                }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action bar if it is
present.
                getMenuInflater().inflate(R.menu.menu_main, menu);
                return true;
                }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
                // Handle action bar item clicks here. The action bar will
                // automatically handle clicks on the Home/Up button, so long
                // as you specify a parent activity in AndroidManifest.xml.
                int id = item.getItemId();

                //noinspection SimplifiableIfStatement
                if (id == R.id.action_settings) {
                return true;
                }

                return super.onOptionsItemSelected(item);
                }
        }
```

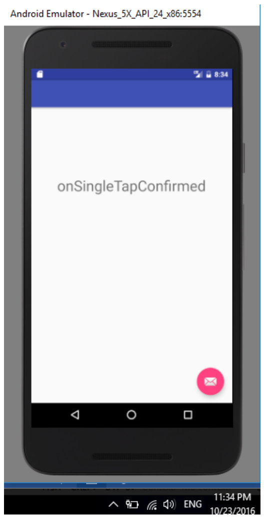
<u>the default activity</u>



<u>The activity while scrolling the screen:</u>



The activity while clicking on the screen:



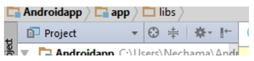
Fragments

The way to create a part of an activity in a separate class, in order to enable using this part in different activities in our project, without having to copy the part inside each activity.

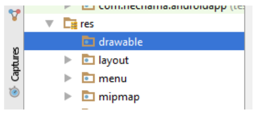
(We can even use multiple fragments in one activity).

In this app we used images, and the way to add images to an android project is:

First we need to change our files window view option to the "project" option



Then we need to find the "res" file



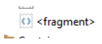
The last step is dragging the wanted image into the "drawable" directory. (the image name has to be created only with lower case chars, and an underscore), in the xml file we can accesses the image in the following syntax: @ drawable/ the image name (without extension)

Creating fragment:

Every fragment is created with 2 different parts:

1) the class part - will contain the functional logic (this class must extend Fragment class)
2) the activity interface - an xml file (right click in the "layout" area-> new -> layout resource file).

After everything is done, we want to add the fragment to the main_activity. We can do it by writing the matching attribute inside the xml file, or by the designer - dragg this option:



Then selecting the fragment that we want to add, from a special window that will appear.

* Important:

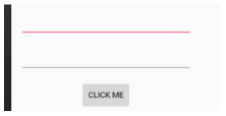
One fragment can't communicate directly with other fragment, and the only way to communicate between them is trough the activity that contains both fragments.

Fragments full example:

In this example we created 2 fragments, and one main activity that contains both fragments.

The first fragment - Top section fragment:

Contains two edit boxes and one button



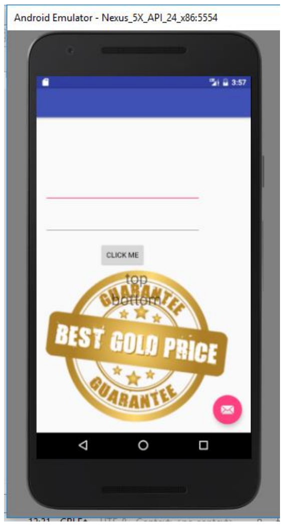
The second fragment - Bottom section fragment:

Contains an image that is the fragment's background, and two text boxes that their default content is: top, bottom.

The main activity:

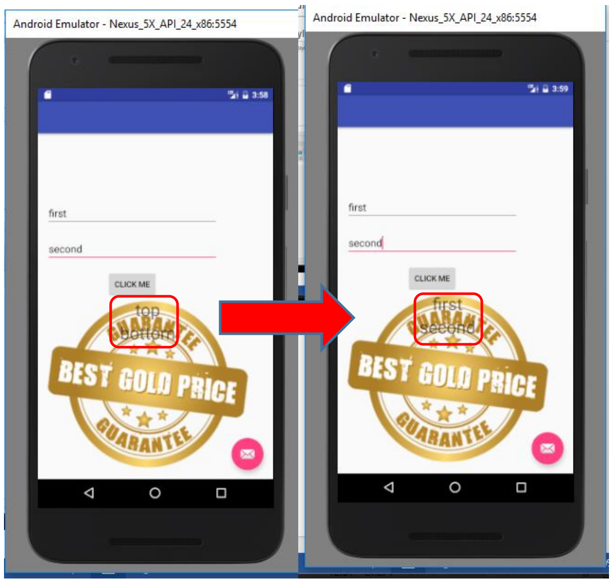
Contains both fragments.

By default will be:



Every time the user enters in the Top fragment's edit boxes two strings, and clicks on the button, the content of the Bottom fragment's text boxes will be changed to the new entered strings.

Before click: after click:


Android Emulator - Nexus_5X_API_24_x86:5554
first
second
CLICK ME
top
bottom
Android Emulator - Nexus_5X_API_24_x86:5554
first
second
CLICK ME
first
second

Main activity:

Class code:

```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;


//every activity in android inherits from the Activity (AppCompatActivity inherits
Activity)

public class MainActivity extends AppCompatActivity implements
TopSectionFragment.TopSectionListener {

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
        }

        //This code is called by the Top Fragment when the user clicked the button
        //this function implements the TopSectionListener interface
        @Override
        public void createClick(String top, String bottom) {
               BottomPictureFragment bottomFragment =
(BottomPictureFragment)getSupportFragmentManager().findFragmentById(R.id.fragment2);
               bottomFragment.setClickedText(top, bottom);
        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action bar if it is
present.
                getMenuInflater().inflate(R.menu.menu_main, menu);
                return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
                // Handle action bar item clicks here. The action bar will
                // automatically handle clicks on the Home/Up button, so long
                // as you specify a parent activity in AndroidManifest.xml.
                int id = item.getItemId();

                //noinspection SimplifiableIfStatement
                if (id == R.id.action_settings) {
                        return true;
                }

                return super.onOptionsItemSelected(item);
        }

}
```

activity xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">


    <fragment
        android:layout_width="match_parent"
        android:layout_height="10px"

        android:name="com.anna.androidapp.BottomPictureFragment"
        android:layout_marginRight="26dp"
        android:id="@+id/fragment2"
        tools:layout="@layout/buttom_picture_fregment"
        android:layout_below="@+id/fragment"
        android:layout_alignParentBottom="true"
         />

    <fragment
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:name="com.anna.androidapp.TopSectionFragment"
        android:id="@+id/fragment"
        tools:layout="@layout/top_section_fregment"
        android:layout_alignParentTop="true" />
</RelativeLayout>
```

Top section fragment:

Class code:

```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.app.Activity;
import android.widget.Button;
import android.widget.EditText;

public class TopSectionFragment extends Fragment {

    private static EditText etTopTextInput;
    private static EditText etBottomTextInput;


    //an property of the inner interface
    TopSectionListener activityCommander;


    //an inner interface that is used to force the activity that will use this class,
```

```java
to implement the createClick function
    public interface TopSectionListener{
        public void createClick(String top, String bottom);
    }


    //this function is activated each time this fregment is attached to an activity
    //in this function we get the interface that the activity created
    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);
        try{
            activityCommander = (TopSectionListener) activity;
        }catch (ClassCastException e){
            throw new ClassCastException(activity.toString());
        }
    }

    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
@Nullable Bundle savedInstanceState) {
        View = inflater.inflate(R.layout.top_section_fregment, container, false);

        etTopTextInput = (EditText)view. findViewById(R.id.text1);
        etBottomTextInput = (EditText)view.findViewById(R.id.text2);

        final Button = (Button)view.findViewById(R.id.btn1);

        button.setOnClickListener(
                new View.OnClickListener(){

                    @Override
                    public void onClick(View v) {
                        buttonClicked(v);
                    }
                }
        );

        return view;
    }

    //Calls this when button clicked
    public void buttonClicked(View view){
        activityCommander.createClick(etTopTextInput.getText().toString(),
etBottomTextInput.getText().toString());
    }
}
```

activity xml:

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <Button
        android:id="@+id/btn1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/text2"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="15dp"
        android:text="click me"/>
    <EditText
        android:id="@+id/text1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="157dp"
        android:width="300dp"
        android:selectAllOnFocus="false"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />

    <EditText
        android:id="@+id/text2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="17dp"
        android:width="300dp"
        android:layout_below="@+id/text1"
        android:layout_alignLeft="@+id/text1"
        android:layout_alignStart="@+id/text1" />
</RelativeLayout>
```

<u>Bottom section fragment:</u>

<u>Class code:</u>

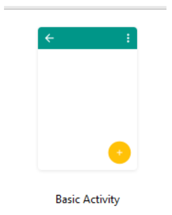
```
package com.anna.androidapp;

import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;


public class BottomPictureFragment extends Fragment {

    private static TextView txtTop;
    private static TextView txtBottom;

    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
@Nullable Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.buttom_picture_fregment, container,
false);
        txtTop = (TextView)view.findViewById(R.id.txtTop);
        txtBottom = (TextView)view.findViewById(R.id.txtBottom);
        return view;
    }

    public void setClickedText(String top, String bottom){
        txtTop.setText(top);
        txtBottom.setText(bottom);
    }

}
```

activity xml:

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/price">

    <TextView

        android:text="top"
        android:textSize="30dp"
        android:gravity="center_horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/txtTop" />

    <TextView
        android:text="bottom"
        android:textSize="30dp"
        android:gravity="center_horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"

        android:id="@+id/txtBottom" />
</LinearLayout>
```

Creating a flow menu:

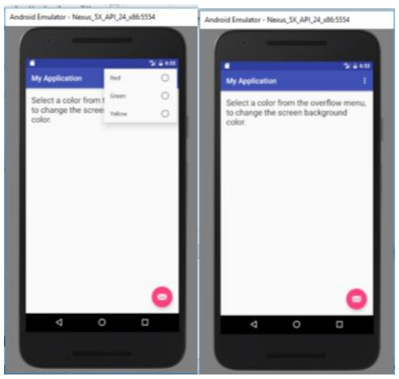
Open a new project and choose this layout:



Then the new project will have one java class to the mainActivity, and two xml files to the activity_main and the menu.

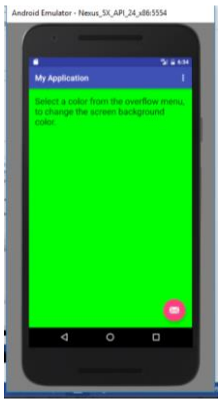
The following code creates an app that contains a flow menu with color options, and sets the main activities background to the checked color.

The default app activity:



Select a color from the overflow menu, to change the screen background color.

The app activity after selecting the green option:



Select a color from the overflow menu, to change the screen background color.

<u>mainActivity class:</u>

```
package com.anna.myapplication;

import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.graphics.Color;
import android.widget.RelativeLayout;


public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        RelativeLayout main_view = (RelativeLayout) findViewById(R.id.content_main);
        switch (item.getItemId()) {
            case R.id.menu_red: {
                if (item.isChecked())
                    item.setChecked(false);
                else
                    item.setChecked(true);
                main_view.setBackgroundColor(Color.RED);
                return true;
            }

            case R.id.menu_green: {
                if (item.isChecked())
                    item.setChecked(false);
                else
                    item.setChecked(true);
                main_view.setBackgroundColor(Color.GREEN);
                return true;
            }

            case R.id.menu_yellow: {
                if (item.isChecked())
                    item.setChecked(false);
                else
                    item.setChecked(true);
                main_view.setBackgroundColor(Color.YELLOW);
                return true;
            }

            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

activity_main xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/content_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.anna.myapplication.MainActivity"
    tools:showIn="@layout/activity_main">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="@string/tvInstructions"
        android:id="@+id/tvInstructions"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />
</RelativeLayout>
```

Menu_main xml:

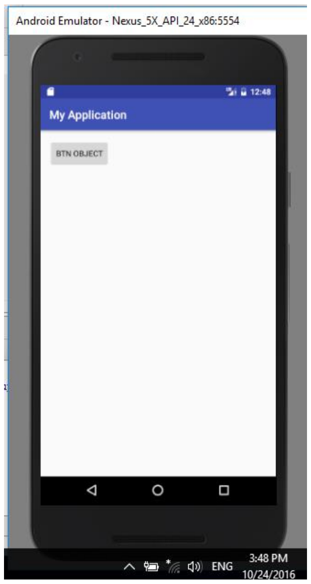
```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".MainActivity"
    >
<group android:checkableBehavior="single">
<item
    android:id="@+id/menu_red"
    android:orderInCategory="1"
    app:showAsAction="never"
    android:title="@string/strRed"/>

<item
    android:id="@+id/menu_green"
    android:orderInCategory="2"
    app:showAsAction="never"
    android:title="@string/strGreen"/>

<item
    android:id="@+id/menu_yellow"
    android:orderInCategory="3"
    app:showAsAction="never"
    android:title="@string/strYellow"/>
</group>

    </menu>
```

Transitions

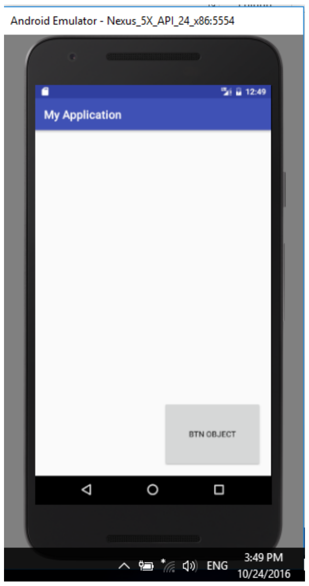
To create the next project, we need to create a minimum 19 version (KitKat version).

The project shows by default a button in the top left side, and by clicking on the screen the button will resize to a bigger size, and remove to the bottom right side.

Default layout:



After clicking:



<u>MainActivity file:</u>

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.transition.TransitionManager;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.RelativeLayout;

public class MainActivity extends AppCompatActivity {

        ViewGroup rlParent;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            rlParent = (ViewGroup)findViewById(R.id.activity_main);
            rlParent.setOnTouchListener(
                    new RelativeLayout.OnTouchListener() {
                        @Override
                        public boolean onTouch(View v, MotionEvent event) {
                            moveButton();
                            return true;
                        }
                    }
            );
        }

        public void moveButton(){
            View btnObject = findViewById(R.id.btnObject);
            TransitionManager.beginDelayedTransition(rlParent);

            // Change the position of the button
            RelativeLayout.LayoutParams positionRules = new
RelativeLayout.LayoutParams(
                    RelativeLayout.LayoutParams.WRAP_CONTENT,
RelativeLayout.LayoutParams.WRAP_CONTENT);
            positionRules.addRule(RelativeLayout.ALIGN_PARENT_BOTTOM,
RelativeLayout.TRUE);
            positionRules.addRule(RelativeLayout.ALIGN_PARENT_RIGHT,
RelativeLayout.TRUE);
            btnObject.setLayoutParams(positionRules);

            //Change the size of the button
            ViewGroup.LayoutParams sizeRules = btnObject.getLayoutParams();
            sizeRules.width = 450;
            sizeRules.height= 300;
            btnObject.setLayoutParams(sizeRules);
        }




}
```

Layout file:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">


    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btn_object"
        android:id="@+id/btnObject"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true" />
</RelativeLayout>
```

Intent

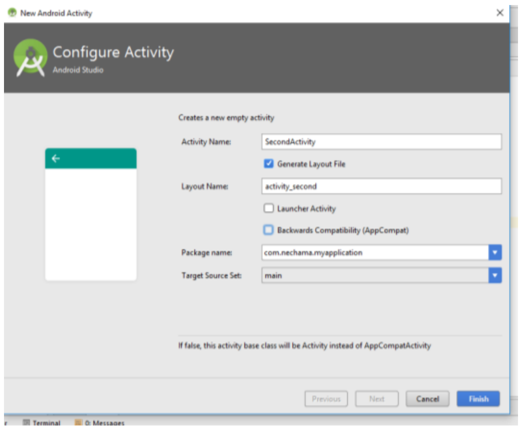
Intent- having few activities in one app, and switching between them.

Adding a new activity to an existing project:

Right click on the “app” directory ->new -> activity -> empty activity

A window with the new activity settings will appear, and we can choose names to the xml file, and the class file.

Important: if it’s not the main activity the “” option must be unchecked.



New Android Activity

Configure Activity
Android Studio

Creates a new empty activity

Activity Name: SecondActivity

☑ Generate Layout File

Layout Name: activity_second

☐ Launcher Activity

☐ Backwards Compatibility (AppCompat)

Package name: com.nechama.myapplication

Target Source Set: main

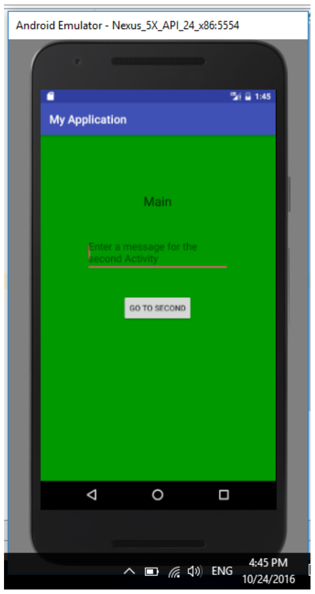
If false, this activity base class will be Activity instead of AppCompatActivity

In the next example, we created two activities:

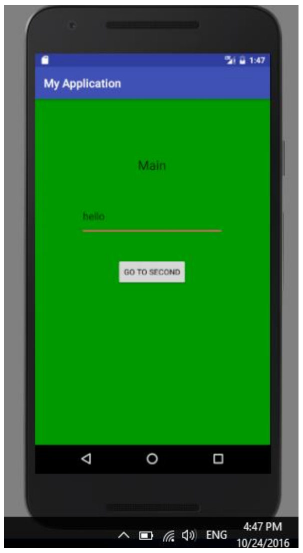
1. main activity- contains a Edit text to enter string, and a button to move to the second activity
2. second activity- contains a text box, and a button to move to the Main activity

The text box in the second activity, will contain the string that the user entered in the Main activity’s edit box.

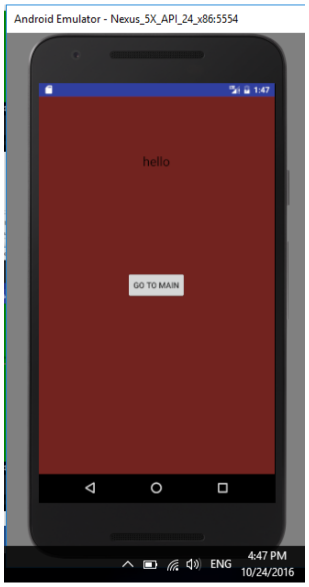
The default activity:


Android Emulator - Nexus_5X_API_24_x86:5554
1:45
My Application
Main
Enter a message for the Second Activity
GO TO SECOND
ENG
4:45 PM
10/24/2016

Main activity after entering input:



Second activity: (contains the Main activity's input)



Main activity:

Class file:

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;;
import android.view.View;
import android.content.Intent;
import android.widget.EditText;


public class MainActivity extends AppCompatActivity {


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void intentToSecondClick(View view){
            //this is the way to switch between activities
            Intent I = new Intent(this, SecondActivity.class);
            // get user input from main activity
            final EditText userInput = (EditText) findViewById(R.id.ptvUserInput);
            String userMessage = userInput.getText().toString();

            // pass user input in the intent to the second activity
            //the first parameter is the key to this message. and the second parameter
is the value
            I.putExtra("MainMessage", userMessage);

            startActivity(I);
        }


    }
```

Xml file:

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:background="#009900"
    tools:context="com.anna.myapplication.MainActivity">



    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="@string/tvApplesText"
        android:id="@+id/tvApples"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="84dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btnBacon"
        android:id="@+id/btnBacon"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:onClick="intentToSecondClick"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/ptvUserInput"
        android:layout_below="@+id/tvApples"
        android:layout_marginTop="42dp"
        android:width="250dp"
        android:layout_centerHorizontal="true"
        android:hint="Enter a message for the second Activity" />

</RelativeLayout>
```

Second activity:

Class file:

```java
package com.anna.myapplication;

import android.app.Activity;
import android.os.Bundle;
import android.content.Intent;
import android.view.View;
import android.widget.TextView;
public class SecondActivity extends Activity {
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_second);

            //if we didn't get any data from the first activity
            //we skip the next code that reads the data from the previous activity
            Bundle prevData = getIntent().getExtras();
            if (prevData== null){
                return;
            }

            //here we set the TextView content to the message frim the first activity
            String MainMessage = prevData.getString("MainMessage");
            TextView tvBacon = (TextView) findViewById(R.id.tvBacon);
            tvBacon.setText(MainMessage);
        }
        public void intentToFirstClick(View view){
            Intent I = new Intent(this, MainActivity.class);

            //startActivity method takes the intent that we want to be the activity
            startActivity(I);
        }



}
```

Xml file:

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_second"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.SecondActivity"
    android:background="#72231f">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="@string/tvBacon"
        android:id="@+id/tvBacon"
        android:layout_marginTop="82dp"
        android:singleLine="true"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btnApplesText"
        android:id="@+id/btnApples"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:onClick="intentToFirstClick" />
</RelativeLayout>
```

<u>Resources: (string.xml)</u>

```
<resources>
    <string name="app_name">My Application</string>
    <string name="action_settings">Settings</string>
    <string name="tvApplesText">Main</string>
    <string name="btnBacon">Go To Second</string>
    <string name="title_activity_bacon">Bacon</string>
    <string name="tvBacon">Second</string>
    <string name="btnApplesText">Go To Main</string>
</resources>
```

<u>Creating a broadcast app between two different projects:</u>

1) <u>first project:</u>

We created a simple project with a class and activity. The activity has a button that onclick activates the function that sends broadcast data.

<u>Class code:</u>

```
package com.example.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Intent;
import android.view.View;



public class MainActivity extends AppCompatActivity {


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void sendBroadcast(View view){
            //setting an Intent that doesn't get a class as a parameter, because he
sends brodcast to everyone that listens
            Intent i = new Intent();

            //here we set the key of this Broadcast sender with the package name
            i.setAction("com.example.anna.myapplication");


            //adding support to all android versions
            i.addFlags(Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
            sendBroadcast(i);
        }



    }
```

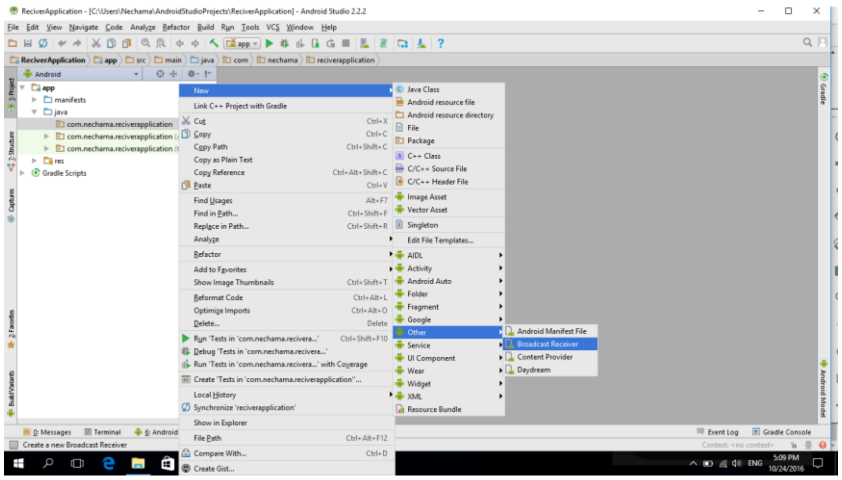
<u>xml file:</u>

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.anna.myapplication.MainActivity">


    <Button
        android:text="send Broadcast"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="125dp"
        android:id="@+id/button2"
        android:onClick="sendBroadcast"
        tools:ignore="HardcodedText" />



</RelativeLayout>
```

<u>2) Second project:</u>

We created a new project, and selected the “no activity” option.

In the Project window we added a class in the directory that holds all the java classes, by clicking right click on this directory, and then:


ReciverApplication - [C:\Users\Nechama\AndroidStudioProjects\ReciverApplication] - Android Studio 2.2.2
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
ReciverApplication app src main java com nechama reciverapplication
Android
app
manifests
java
com.nechama.reciverapplication
res
Gradle Scripts
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy Ctrl+C
Copy Path Ctrl+Shift+C
Copy as Plain Text
Copy Reference Ctrl+Alt+Shift+C
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Show Image Thumbnails Ctrl+Shift+T
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Run 'Tests in 'com.nechama.recivera...' Ctrl+Shift+F10
Debug 'Tests in 'com.nechama.recivera...'
Run 'Tests in 'com.nechama.recivera...' with Coverage
Create 'Tests in 'com.nechama.reciverapplication''...
Local History
Synchronize 'reciverapplication'
Show in Explorer
File Path Ctrl+Alt+F12
Compare With... Ctrl+D
Create Gist...
Java Class
Android resource file
Android resource directory
File
Package
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset
Singleton
Edit File Templates...
AIDL
Activity
Android Auto
Folder
Fragment
Google
Other
Service
UI Component
Wear
Widget
XML
Resource Bundle
Android Manifest File
Broadcast Receiver
Content Provider
Daydream
0: Messages
Terminal
6: Android
Event Log
Gradle Console
Create a new Broadcast Receiver
5:09 PM
10/24/2016

<u>In the class we edited the following code:</u>

```java
package com.anna.reciverapplication;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.widget.Toast;

public class MyReceiver extends BroadcastReceiver {
    public MyReceiver() {
    }

    //this function is executed when this app receives brodcast
    @Override
    public void onReceive(Context context, Intent intent) {

        //the "Toast.makeText" function shows a popup message to the screen
        //the 3 parameters that we send are:context, the string that will be the
message, and the length that this popup will be shown
        Toast.makeText(context,"bradcost has been recived",Toast.LENGTH_LONG).show();
    }
}
```

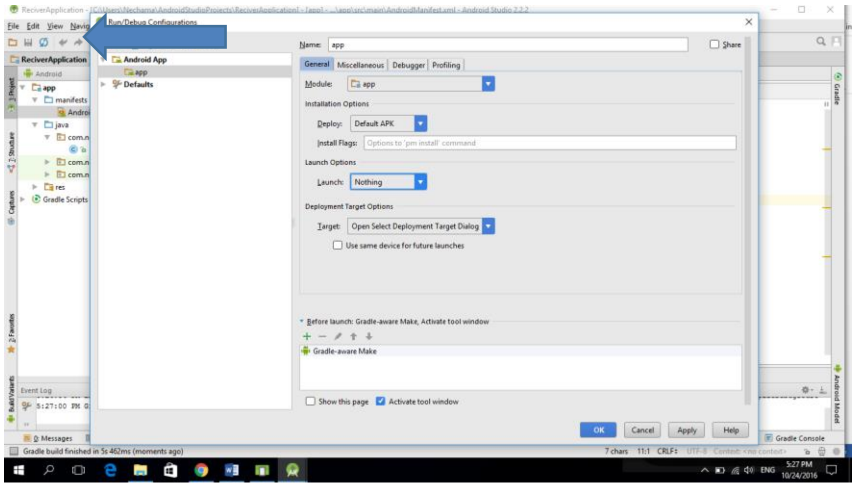
<u>in the manifest we added the receiver configurations:</u>

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.anna.reciverapplication">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:exported="true">
this                <intent-filter>
```

```
 we               <action android:name="com.example.anna.myapplication"></action> added
</intent-filter>
        </receiver>
    </application>

</manifest>
```

Running the Receiver app:

Because this project don't have any activity, before running it we have to set the "lunch" option to "nothing"



Run/Debug Configurations

Name: app

General | Miscellaneous | Debugger | Profiling

Module: app

Installation Options

Deploy: Default APK

Install Flags: Options to 'pm install' command

Launch Options

Launch: Nothing

Deployment Target Options

Target: Open Select Deployment Target Dialog

☐ Use same device for future launches

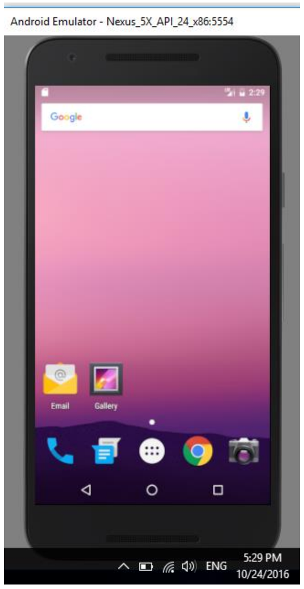
Before launch: Gradle-aware Make, Activate tool window

Gradle-aware Make

☐ Show this page ☑ Activate tool window

Gradle build finished in 5s 462ms (moments ago)

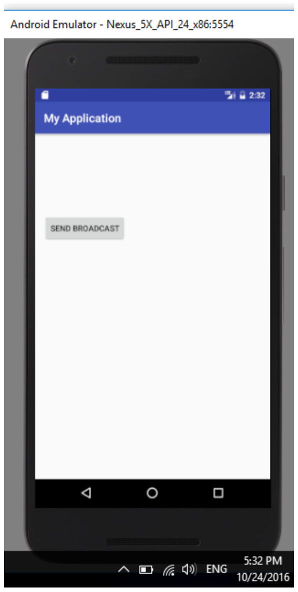
Then we can run the app and the result will be:



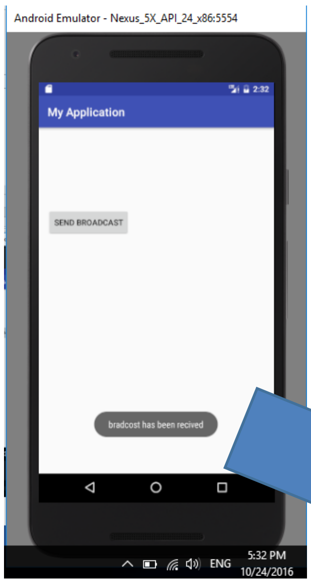
Running the sender app:

This project contains an activity, and we will run it normally, on the same emulator that the receiver is running.

The result will be:



After clicking the button:



bradcost has been recived

This in shown by

the receiver app

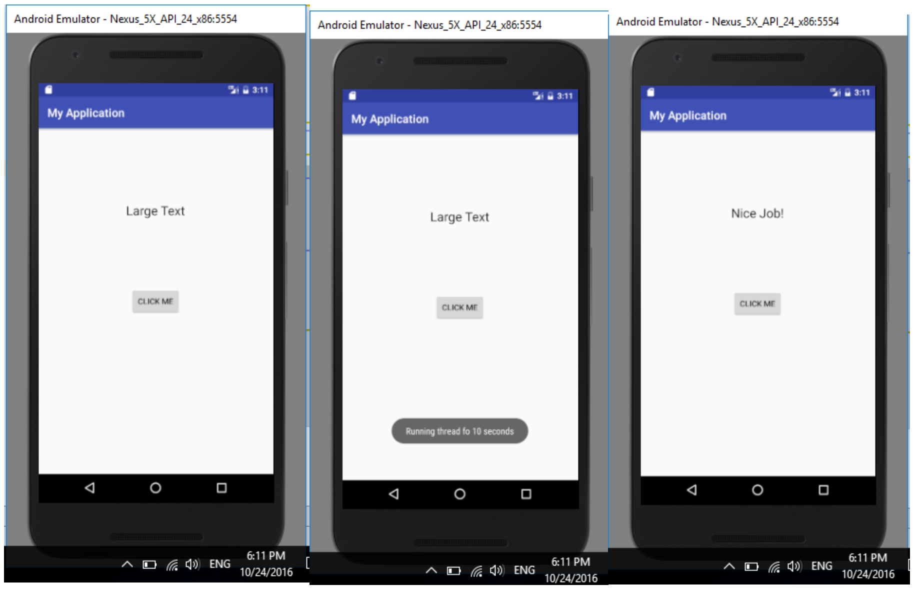
Threads in android

The threads are used to do actions in the background, without blocking the whole entire process.

Important: to change interface component from a thread we need to create a special handler.

In the following project we created an app that shows a popup message right after the click, and changes the text view content 10 seconds after the click.

Before clicking: right after clicking: 10 s after clicking:


Android Emulator - Nexus_5X_API_24_x86:5554
My Application
Large Text
CLICK ME
Android Emulator - Nexus_5X_API_24_x86:5554
My Application
Large Text
CLICK ME
Running thread fo 10 seconds
Android Emulator - Nexus_5X_API_24_x86:5554
My Application
Nice Job!
CLICK ME
6:11 PM
10/24/2016

<u>The java class:</u>

```java
package com.example.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.os.Handler;
import android.os.Message;
import android.widget.TextView;
import android.widget.Toast;



public class MainActivity extends AppCompatActivity {

        Handler waitMsgHandler = new Handler(){
            @Override
            public void handleMessage(Message msg) {
                TextView tv1 = (TextView) findViewById(R.id.tv1);
                tv1.setText("Nice Job!");
            }
        };
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void btn1Clicked(View view){

            Runnable r = new Runnable() {

 //this function runs when we start the thread that stores this runnable object
                @Override
                public void run() {
                    long futureTime = (System.currentTimeMillis()+ 10000);//10 sec
                    while (System.currentTimeMillis()< futureTime){
                        synchronized(this){
                            try {
                                wait(futureTime - System.currentTimeMillis());
                            }catch(Exception e){}
                        }
                    }

                    //updating the interface will be only trough a waitMsgHandler
                    waitMsgHandler.sendEmptyMessage(0);

                }
            };

//here we initialized the thread object with the runnable object
//but we can also do it in a different way -creating an anonymous runnable class
instead of passing a runnable object
            Thread waitThread = new Thread(r);

            waitThread.start();
            //This toast will execute immediately
            Toast.makeText(MainActivity.this,"Running thread fo 10
seconds",Toast.LENGTH_LONG).show();

}

    }
```

The xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.anna.myapplication.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Large Text"
        android:id="@+id/tv1"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="112dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click Me"
        android:id="@+id/btn1"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:onClick="btn1Clicked" />

</RelativeLayout>
```

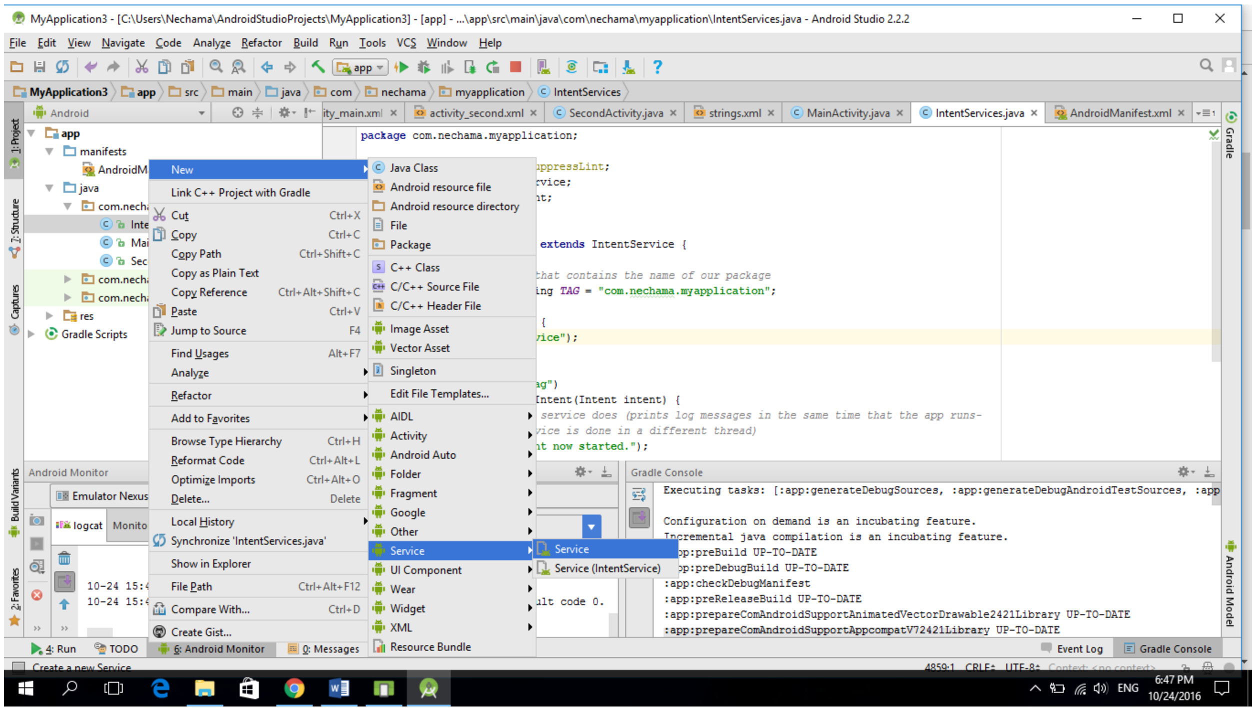
Services

Services - actions that are running in the background, and don't have any interface.

(etc. downloading images).

Crating a service is in the following way:

Right click on the app directory, then:



```
package com.nechama.myapplication;

uppressLint;
rvice;
nt;

 extends IntentService {

that contains the name of our package
ing TAG = "com.nechama.myapplication";

 {
vice");

ag")
Intent(Intent intent) {
 service does (prints log messages in the same time that the app runs-
vice is done in a different thread)
nt now started.");
```

```
10-24 15:
10-24 15:
ult code 0.
```

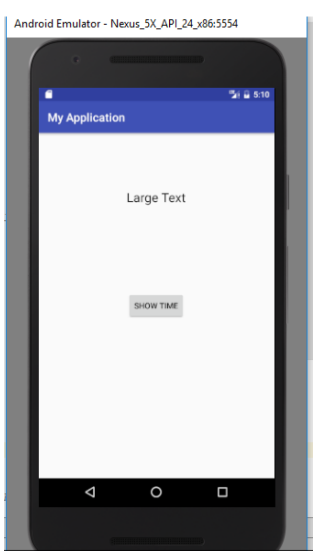
```
Executing tasks: [:app:generateDebugSources, :app:generateDebugAndroidTestSources, :app

Configuration on demand is an incubating feature.
Incremental java compilation is an incubating feature.
pp:preBuild UP-TO-DATE
pp:preDebugBuild UP-TO-DATE
:app:checkDebugManifest
:app:preReleaseBuild UP-TO-DATE
:app:prepareComAndroidSupportAnimatedVectorDrawable2421Library UP-TO-DATE
:app:prepareComAndroidSupportAppcompatV72421Library UP-TO-DATE
```

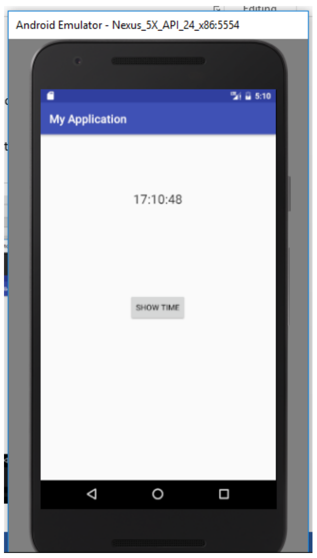
Binding service to activity

In the following example we created a service that returns the current time, and an activity that contains a button and a text view. Each click on the button sets the text view content to the current time according to the service information.

In order to insert the service data into an interface component (a service cant directly access the interface), we created a binder.

Before clicking: After clicking:





<u>Main activity code:</u>

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.os.IBinder;
import android.content.Context;
import android.content.Intent;
import android.content.ComponentName;
import android.content.ServiceConnection;
import com.anna.myapplication.TimeService.MyLocalBinder;

public class MainActivity extends AppCompatActivity {

    TimeService theService;
    boolean isBound = false;

    public void btnShowTime(View view){

        //connecting to the service and getting the current time
        String currentTime = theService.getCurrentTime();

        //setting the current time to the textview's content
        TextView tvShowTime = (TextView) findViewById(R.id.tvShowTime);
        tvShowTime.setText(currentTime);
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //here we create an instance to the TimeService
        Intent i = new Intent(this, TimeService.class);

        //here we create the bind
        //the 3 parameters are: the Intent, ServiceConnection, and
Context.BIND_AUTO_CREATE)
        bindService(i, theConnection, Context.BIND_AUTO_CREATE);
    }

    //here we create a ServiceConnection that gives as access to the TimeService
    private ServiceConnection theConnection = new ServiceConnection() {

        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            // get a ref to the binder class and call the bind method
            MyLocalBinder binder = (MyLocalBinder) service;
            theService = binder.getService();
            isBound = true;
        }

        @Override
        public void onServiceDisconnected(ComponentName name) {
            isBound = false;
        }
    };


}
```

main activity xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Large Text"
        android:id="@+id/tvShowTime"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="81dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Show Time"
        android:id="@+id/btnShowTime"
        android:onClick="btnShowTime"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```

TimeServise code:

```
package com.anna.myapplication;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.os.Binder;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

public class TimeService extends Service {

        //this is the object that holds the binder for the interface from this service
        //this object is an instance of the MyLocalBinder class (inner class in this
class)
        private final IBinder Binder = new MyLocalBinder();


        public TimeService() {
        }

        @Override
        public IBinder onBind(Intent intent) {
            return Binder;
        }

        //this function actually returns the service result (the current time)
        public String getCurrentTime(){
            SimpleDateFormat df = new SimpleDateFormat("HH:mm:ss", Locale.US);
            return (df.format(new Date()));
        }

        //this is an inner class that extends Binder and enables this service affect
the interface
        public class MyLocalBinder extends Binder {

            //this function returns a reference to this class, so we can use it's
information
            TimeService getService(){
                return TimeService.this;
            }
        }

    }
```

Insert items into List view

The MainActivity code:

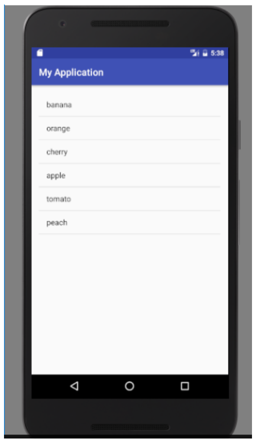
```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.Toast;
```

```java
public class MainActivity extends AppCompatActivity {


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            //crete string array
            String[] foods = {"banana", "orange", "cherry", "apple", "tomato",
"peach"};

            //the parameter "android.R.layout.simple_list_item_1" is setting the list
format design
            ListAdapter MyAdapter = new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1, foods);

            //getting the ListView component from the layout
            ListView MyListView = (ListView) findViewById(R.id.MyListView);

            //adding the ListAdapter that we created to the ListView
            MyListView.setAdapter(MyAdapter);

            //adding event listener to the ListView
            MyListView.setOnItemClickListener(
                    //On item click we will show to the screen a popup with the item's
content
                    new AdapterView.OnItemClickListener(){
                        @Override
                        public void onItemClick(AdapterView<?> parent, View, int
position, long id) {

                            String food =
String.valueOf(parent.getItemAtPosition(position));
                            Toast.makeText(MainActivity.this, food,
Toast.LENGTH_LONG).show();
                        }
                    }
            );
        }

    }
```

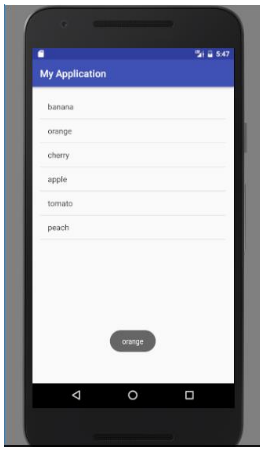
<u>The xml file:</u>

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">
    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/MyListView"></ListView>

</RelativeLayout>
```

The result will be



After clicking on the "orange" item:



Creating a customer list view

In the following project we created a template of a list item in a separate xml file, and then creating a list view from the main activity, by setting each list item to the template list item.

(the template contains an image named "back" that we added to our project)

The mainActivity code:

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.Toast;


public class MainActivity extends AppCompatActivity {


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            //crete string array
            String[] foods = {"banana", "orange", "cherry", "apple", "tomato",
"peach"};


            ListAdapter customListAdapter = new CustomAdapter(this,foods);// Pass the
food arrary to the constructor.
            ListView customListView = (ListView) findViewById(R.id.custom_ListView);
            customListView.setAdapter(customListAdapter);

            //adding event listener to the ListView
            customListView.setOnItemClickListener(
                    //On item click we will show to the screen a popup with the item's
content
                    new AdapterView.OnItemClickListener(){
                        @Override
                        public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {

                            String food =
String.valueOf(parent.getItemAtPosition(position));
                            Toast.makeText(MainActivity.this, food,
Toast.LENGTH_LONG).show();
                        }
                    }
            );
        }

    }
```

<u>The CustomAdapter code:</u>

```
package com.anna.myapplication;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

class CustomAdapter extends ArrayAdapter<String>{
    public CustomAdapter(Context context, String[] foods) {
        super(context, R.layout.custom_row ,foods);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

        LayoutInflater myCustomInflater = LayoutInflater.from(getContext());
        View customView = myCustomInflater.inflate(R.layout.custom_row, parent,
false);

        // get references.
        String singleFoodItem = getItem(position);
        TextView itemText = (TextView) customView.findViewById(R.id.item_text);
        ImageView itemImage = (ImageView)
customView.findViewById(R.id.my_profile_image);

        // dynamically update the text from the array
        itemText.setText(singleFoodItem);

        // using the same image every time
        itemImage.setImageResource(R.drawable.back);

        // Now we can finally return our custom View or custom item
        return customView;
    }
}
```

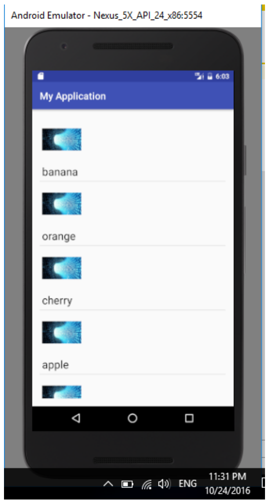
The mainActivity xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">
    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/custom_ListView"></ListView>
</RelativeLayout>
```

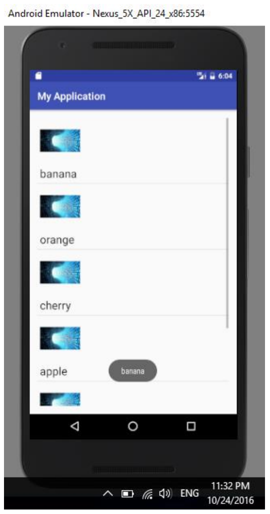
The list-item xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:layout_width="80dp"
        android:layout_height="80dp"
        android:id="@+id/my_profile_image"
        android:src="@drawable/back"
        android:layout_margin="5dp" />


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Large Text"
        android:id="@+id/item_text"
        android:layout_margin="5dp" />
</LinearLayout>
```

The result will be:



After clicking on the “banana” item:

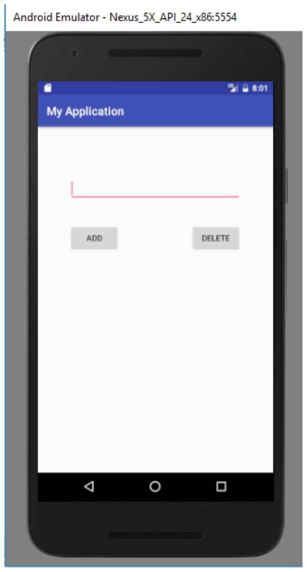


<u>Saving data in android DB</u>

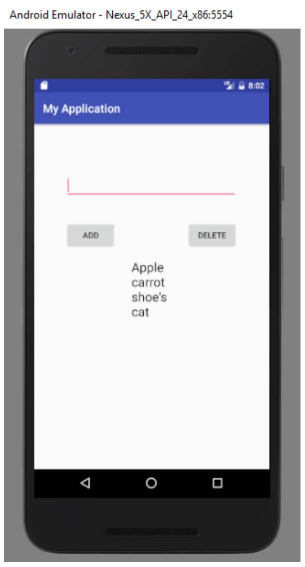
Creating an app that allows to save products in the SQLite.

The user can add or delete products. And can see all the products that are stored in the DB.

<u>The default activity on first run:</u>

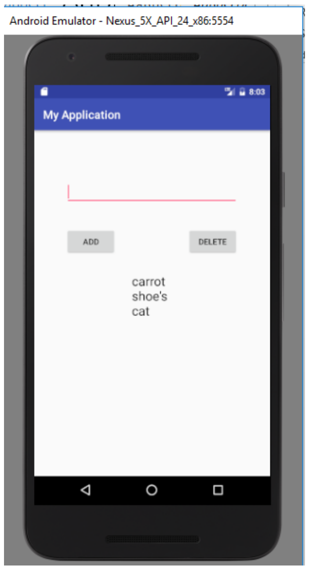


<u>The activity after adding some products to the Db:</u>



Apple
carrot
shoe's
cat

The activity after deleting the "Apple" product from Db:


Android Emulator - Nexus_5X_API_24_x86:5554

carrot
shoe's
cat

Product class

```
package com.anna.myapplication;

/**
 * Created by Anna on 10/25/2016.
 */

public class Products {
    private int _id;
    private String _productname;

    //Added this empty constructor in case we ever want to create the object and
assign it later.
    public Products(){

    }
    public Products(String productName) {
        this._productname = productName;
    }

    public int get_id() {
        return _id;
    }

    public void set_id(int _id) {
        this._id = _id;
    }

    public String get_productname() {
        return _productname;
    }

    public void set_productname(String _productname) {
        this._productname = _productname;
    }
}
```

Class MyDBHandler

```
package com.anna.myapplication;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.database.Cursor;
import android.content.Context;
import android.content.ContentValues;


//this class handles the all database tasks and extends the SQLiteOpenHelper (a build
class)
//in this class we must to implement the methods "onCreate" and "onUpgrade" from the
SQLiteOpenHelper class
public class MyDBHandler extends SQLiteOpenHelper{

    private static final int DATABASE_VERSION = 1;
    private static final String DATABASE_NAME = "productDB.db";
    public static final String TABLE_PRODUCTS = "products";
    public static final String COLUMN_ID = "_id";
    public static final String COLUMN_PRODUCTNAME = "productname";


    //We need to pass database information along to superclass because the super class
doesn't have any default constructor
    public MyDBHandler(Context context, String name, SQLiteDatabase.CursorFactory
factory, int version) {
        super(context, DATABASE_NAME, factory, DATABASE_VERSION);
    }

    //in the first time that we run this app we want to create our table in order to
store data
    @Override
    public void onCreate(SQLiteDatabase db) {

        //every column is created by it's name, and by it's data type
        String query = "CREATE TABLE " + TABLE_PRODUCTS + "(" +
                COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                COLUMN_PRODUCTNAME + " TEXT " +
                ");";

        db.execSQL(query);
    }

    //upgrade method is activated when we want to update our table
    //first we delete our old table, and then we create a new table by calling the
"onCreate" method
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_PRODUCTS);
        onCreate(db);
    }

    //Add a new row to the database
    public void addProduct(Products product){
        ContentValues values = new ContentValues();
        values.put(COLUMN_PRODUCTNAME, product.get_productname());  //takes column's
name and value
        SQLiteDatabase db = getWritableDatabase();   //here we get ref to our db
        db.insert(TABLE_PRODUCTS, null, values);
        db.close();  //after adding a product we close the connection to the DB
    }

    //Delete a product from the database
    public void deleteProduct(String productName){
        SQLiteDatabase db = getWritableDatabase();
        db.execSQL("DELETE FROM " + TABLE_PRODUCTS + " WHERE " + COLUMN_PRODUCTNAME +
"=\"" + productName + "\";");
    }

    // this is used in record_TextView in the Main activity, and returns a string that
contains all product's information
    public String databaseToString(){
        String dbString = "";
        SQLiteDatabase db = getWritableDatabase();
        String query = "SELECT * FROM " + TABLE_PRODUCTS + " WHERE 1"; // the
condition "WHERE 1"

        //Cursor can point to a row location in your results
        Cursor recordSet = db.rawQuery(query, null);

        //Move to the first row in your results
        recordSet.moveToFirst();

        //the loop will run while there are rows to read
        while (!recordSet.isAfterLast()) {

            // null could happen if we used our empty constructor
            if (recordSet.getString(recordSet.getColumnIndex("productname")) != null)
{
                dbString +=
recordSet.getString(recordSet.getColumnIndex("productname"));
                dbString += "\n";
            }
            recordSet.moveToNext();
        }
        db.close();
        return dbString;
    }

}
```

MainActivity class

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    EditText userInput;
    TextView recordsTextView;
    MyDBHandler dbHandler;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        userInput = (EditText) findViewById(R.id.user_Input);
        recordsTextView = (TextView) findViewById(R.id.records_TextView);


        /* Can pass nulls because of the constants in the helper.
         * the 1 means version 1 so don't run update.
         */
        dbHandler = new MyDBHandler(this, null, null, 1);
        printDatabase();
    }

    //Print the database into the textView
    public void printDatabase(){
        String dbString = dbHandler.databaseToString();
        recordsTextView.setText(dbString);
        userInput.setText("");
    }

    //add your elements onclick methods.
    //Add a product to the database
    public void addButtonClicked(View view){

        // dbHandler.add needs an product parameter.
        Products product = new Products(userInput.getText().toString());
        dbHandler.addProduct(product);
        printDatabase();  //updates the output list
    }

    //Delete items
    public void deleteButtonClicked(View view){
        // dbHandler delete needs string to find in the db
        String inputText = userInput.getText().toString();
        dbHandler.deleteProduct(inputText);
        printDatabase();  //updates the output list
    }

}
```

MainActivity xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/user_Input"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="69dp"
        android:width="300dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Add"
        android:id="@+id/add_Button"
        android:layout_below="@+id/user_Input"
        android:layout_alignStart="@+id/user_Input"
        android:layout_marginTop="40dp"
        android:onClick="addButtonClicked" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Delete"
        android:id="@+id/delete_Button"
        android:layout_alignTop="@+id/add_Button"
        android:layout_alignEnd="@+id/user_Input"
        android:onClick="deleteButtonClicked" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Large Text"
        android:id="@+id/records_TextView"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```

Adding video to an activity

In order to add a video to our app according to the following code, we need to set the project's version, to minimum 15's version.

MainActivity Class:

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.VideoView;
import android.widget.MediaController;
```

```
public class MainActivity extends AppCompatActivity {


        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            final VideoView myVideoView = (VideoView) findViewById(R.id.MyVideoView);
            myVideoView.setVideoPath("testVideo.mp4");

            //Player controls(play, pause, stop, etc...)
            MediaController = new MediaController(this);
            mediaController.setAnchorView(myVideoView);
            myVideoView.setMediaController(mediaController);

            myVideoView.start();
        }

    }
```

<u>xml file:</u>

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">

    <VideoView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/MyVideoView" />
</RelativeLayout>
```

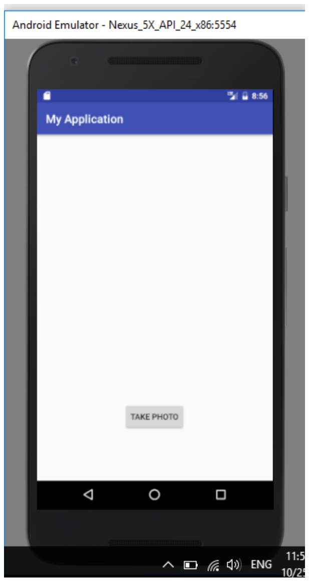
<u>Controlling the device's camera</u>

We created an activity with a button that onclick opens the mobile's camera, and then displays the taken image inside an image view.

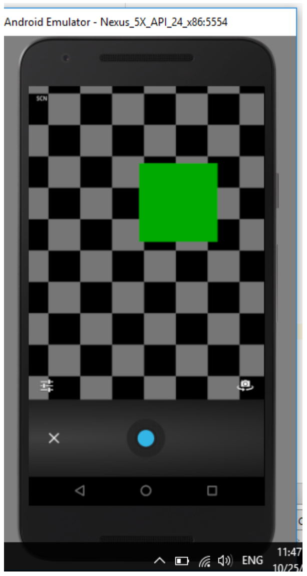
In order to enable our app to use the camera we need to add in the mainfast:

```
<uses-feature android:name="android.hardware.Camera" android:required="true"></uses-
feature>
```

<u>Before clicking the button:</u>

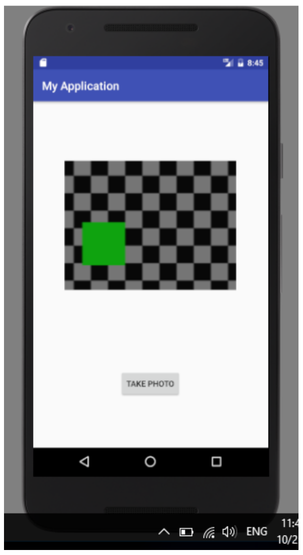


After clicking the button: (the camera is on)



After taking a picture: (the image is displayed in the image box)

Android Emulator - Nexus_5X_API_24_x86:5554



MainActivity class:

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.provider.MediaStore;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

    //a way to adentify each intent (if we use different intents in the app)
        static final int REQUEST_IMAGE_CAPTURE = 1;
        ImageView myImageView;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            Button buckyButton = (Button) findViewById(R.id.myButton);
            myImageView = (ImageView) findViewById(R.id.myImageView);

            //Disable the button if the user's device has no camera
            if(!hasCamera())
                buckyButton.setEnabled(false);
        }

        //Check if the user has a camera and return the boolean result
        private boolean hasCamera(){
            return
getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA_ANY);
```

```
        }

        //Launching the camera. this method is called onclicking the activity's button
        public void launchCamera(View view){
            Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

            //Take a picture and pass results along to onActivityResult method
            startActivityForResult(intent, REQUEST_IMAGE_CAPTURE);
        }

        //returning the taken image, and adding it to the image box
        @Override
        protected void onActivityResult(int requestCode, int resultCode, Intent data)
{

            //checking if a picture were taken, and the result was ok
            if(requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK){

                //Get the photo
                Bundle extras = data.getExtras();
                Bitmap photo = (Bitmap) extras.get("data");
                myImageView.setImageBitmap(photo);
            }
        }

    }
```

MainActivity xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Take Photo"
        android:id="@+id/myButton"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="72dp"
        android:onClick="launchCamera" />

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/myImageView"
        android:layout_alignParentTop="true"

        android:layout_centerHorizontal="true"
        android:layout_marginTop="50dp"
        android:minHeight="300dp"
        android:minWidth="300dp" />
</RelativeLayout>
```

Mainfest:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.anna.myapplication">
<uses-feature android:name="android.hardware.Camera" android:required="true"></uses-
feature>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <service
            android:name=".MyService"
            android:enabled="true"
            android:exported="true"></service>
    </application>

</manifest>
```

Notification

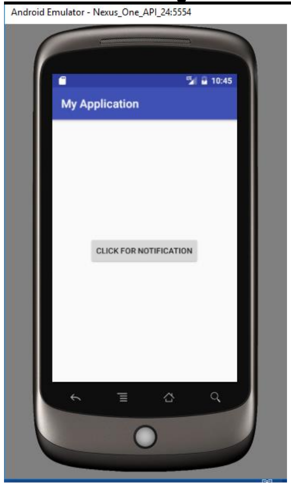
MainActivity class:

```
package com.anna.myapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Intent;
import android.view.View;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.support.v4.app.NotificationCompat;


public class MainActivity extends AppCompatActivity {

        NotificationCompat.Builder notification;

        private static final int uniqueID = 45612;  //the uniqueID is a number that we
chose

        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            notification = new NotificationCompat.Builder(this);
            //here we cancel the notification that was displayed in the user's screen
            //because after a user clicks on the notification he
            notification.setAutoCancel(true);
        }

        public void ButtonClicked(View view){
            //Build the notification
            notification.setSmallIcon(R.mipmap.ic_launcher);

            //this is the text that appers when the notification appears
            notification.setTicker("This is the ticker");

            //this setts the time that will appear on the notification (as a receive
time)
            notification.setWhen(System.currentTimeMillis());

            notification.setContentTitle("the notification's title");
            notification.setContentText("body text notification");

            //here we create an intent, and send the class of the activity that we
want to access by the notification
            Intent intent = new Intent(this, MainActivity.class);

            //gives our app an access to the device's home screen
            PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent,
PendingIntent.FLAG_UPDATE_CURRENT);

            notification.setContentIntent(pendingIntent);

            //Builds notification and issues it
            NotificationManager nm = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
            nm.notify(uniqueID, notification.build());

        }

    }
```

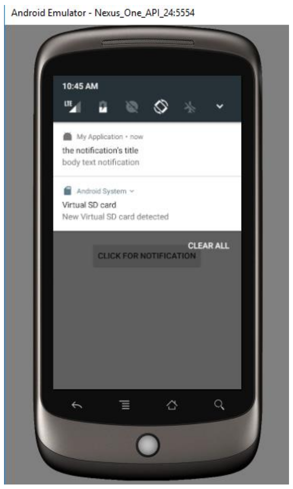
MainActivity xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.anna.myapplication.MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click for Notification"
        android:id="@+id/Button1"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:onClick="ButtonClicked" />
</RelativeLayout>
```

The result by default



The result after clicking the activities' button



the notification's title
body text notification

Android System
Virtual SD card
New Virtual SD card detected