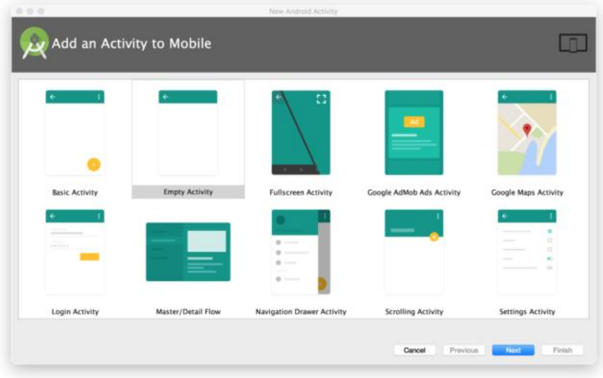
Android Development Lab

- Course no.: 61985

- Lecturer: Daniel Bouenos
- Email: dbouenos@braude.ac.il
- Topics: Activity life cycle

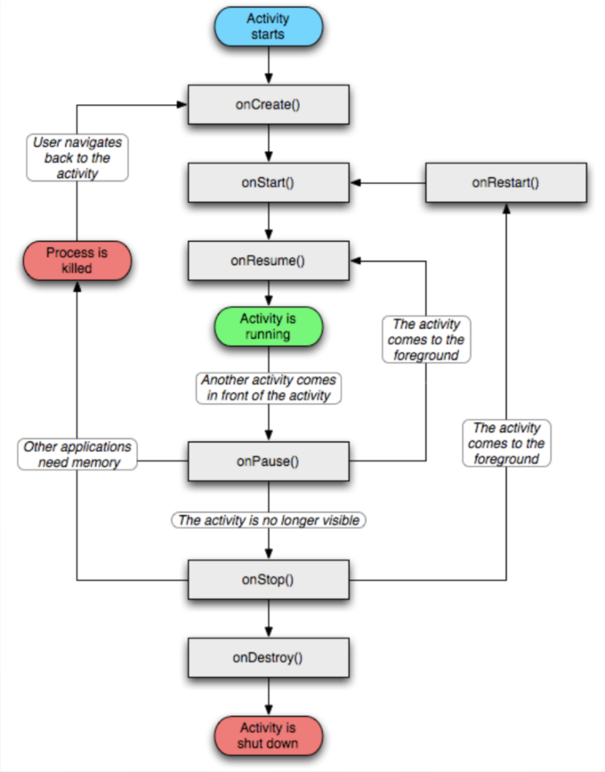
The Android “Activity”

- Represent one full screen of the app
  - You may have several activities in your app
- Includes:
  - The Java code file that handles the screen interactions
  - One or more UI layouts that defines the displayed widgets
- The Activity is acting as an “entry point” to the app
- Each such entry is defined in the manifest file
  - Public or not
  - Permissions
  - Actions filters that may invoke it



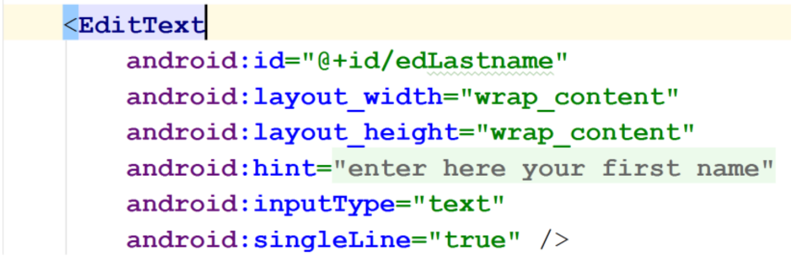
The Activity Life Cycle

- The activity is handled in Android by a Class that must extends the “Activity” superclass or one of its descendants
- The activity has a life cycle composed of several ordered “states”
  - See the diagram …
  - Each state is represented by a specific callback method
- As a developer, you may need to interfere in one or more of these states to change the default behavior or add your own handler
- You achieve that by overriding the specific callback method
- See more details in this link…


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
Other applications need memory
onPause()
The activity comes to the foreground
The activity is no longer visible
onStop()
onDestroy()
Activity is shut down

Accessing widgets from your Activity

Layout xml



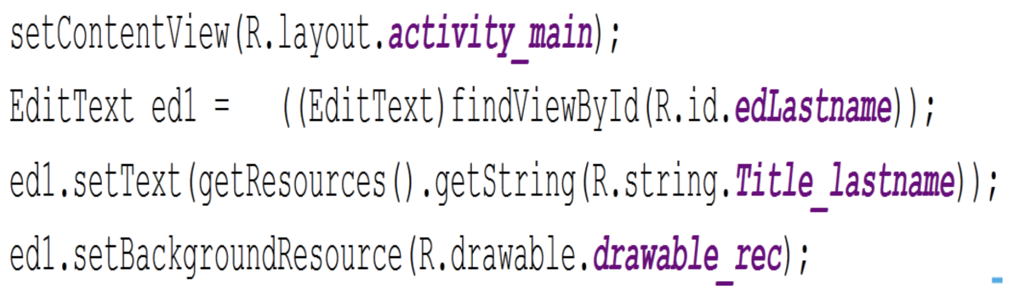
```
<EditText
    android:id="@+id/edLastname"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="enter here your first name"
    android:inputType="text"
    android:singleLine="true" />
```

R java class

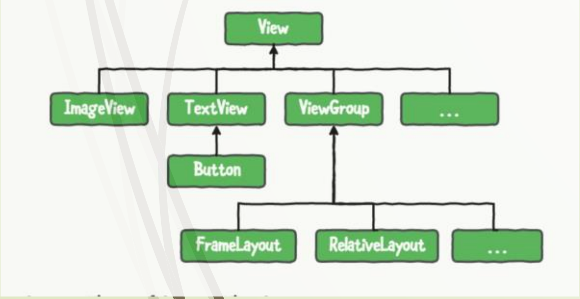
Various View objects in memory

View1..n

Activity



```
setContentView(R.layout.activity_main);
EditText ed1 =   ((EditText)findViewById(R.id.edLastname));
ed1.setText(getResources().getString(R.string.Title_lastname));
ed1.setBackgroundResource(R.drawable.drawable_rec);
```



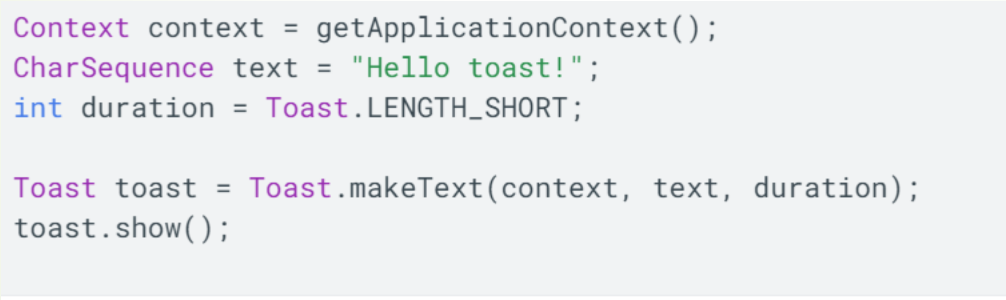
Tracing your code

LOGCAT

- Console that displays system messages and messages generated by the various apps
- Messages are categorized by importance levels
  - Verbose, Debug, Information, Warning, Error
- Filtering and searching capabilities
- Your app can write messages to the LogCat by invoking one of these methods depending the level:
  - Log.d(TAG, msg), Log.i(…), Log.w(…), Log.e(…)
  - The TAG is just a string literal to simplify the searching
- Let’s demonstrate…
- See details on this link…

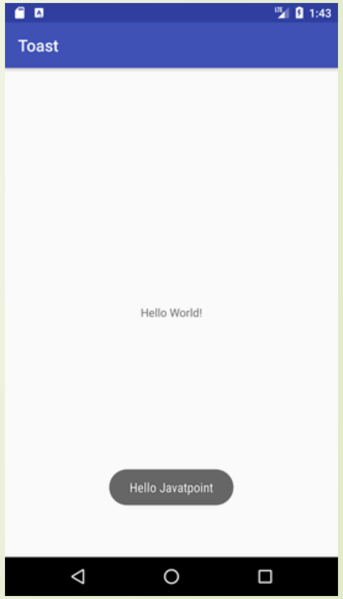
Use Toast messaging from your Activity

- Your code can also pop up a message to the user for a short period of time
- This is achieved through the Toast library, see below example:



```
Context context = getApplicationContext();
CharSequence text = "Hello toast!";
int duration = Toast.LENGTH_SHORT;

Toast toast = Toast.makeText(context, text, duration);
toast.show();
```



- You can control the duration and the display location
- You can also customize the layout of the message
- See details on this link…

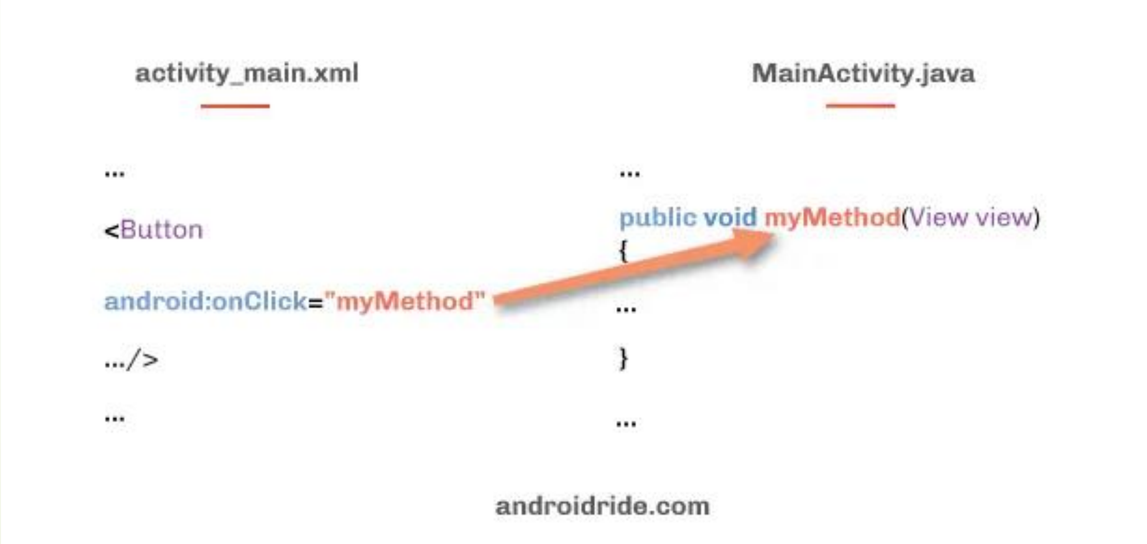
Responding to Widget ‘click’ events

- The fastest way to get a click handler for a widget is to:

1. Define a public method in your Activity with this signature

   public void myHandler(View v)

2. Add “onClick” attribute to your widget in the layout resource file and put a reference to the above method



activity_main.xml

```
...
<Button
android:onClick="myMethod"
.../>
...
```

MainActivity.java

```
...
public void myMethod(View view)
{
...
}
...
```

androidride.com

Saving/Restoring Activity state on orientation change

- The state/value of the widgets are automatically saved/restored during orientation change
  - …as long they have an ID
- Yet all code based field that you add to the Activity as class members will not be restored when changing the orientation
- One way to persist this data is within these two methods
  - onSaveInstanceState, onRestoreInstanceState
- See example here…

This week assignment – EX2

- Two parts
  1. Fill the lifecycle table, see next slide
  2. Develop the Calculator described next…

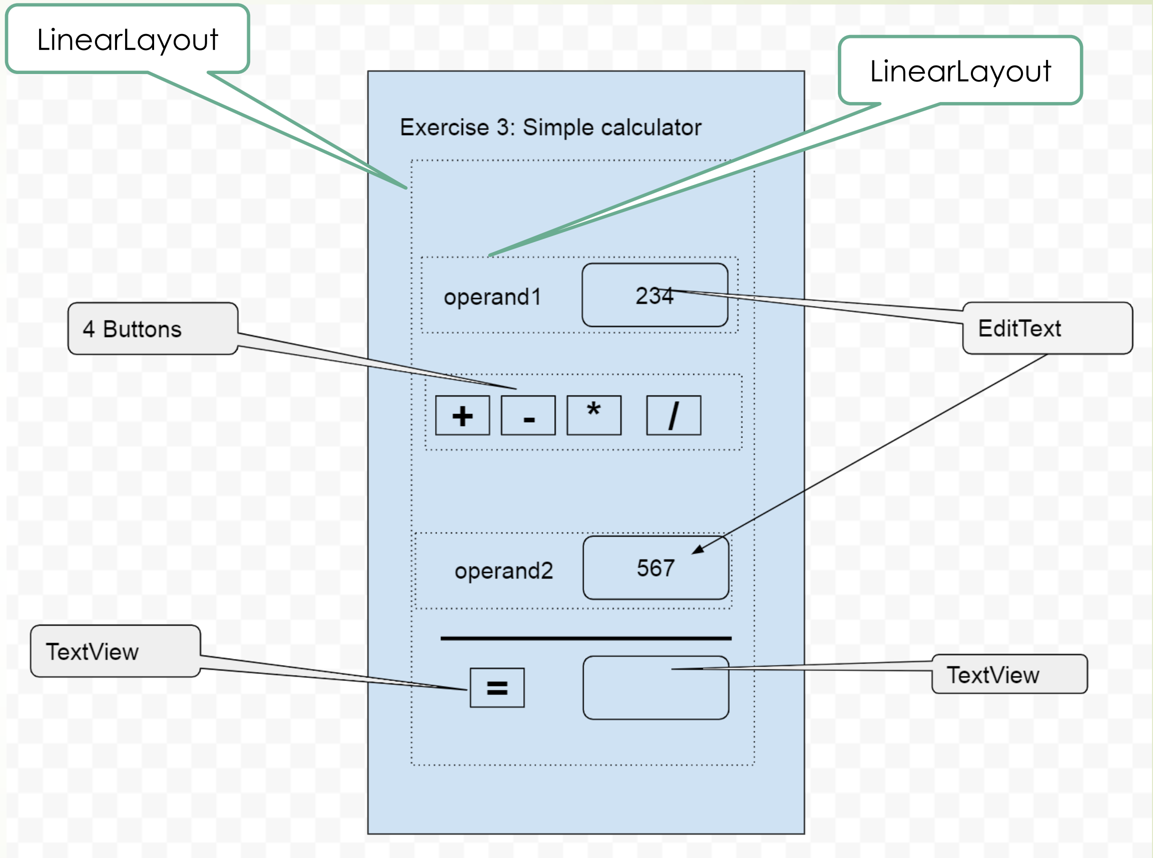
Lifecycle summary sheet

	First Time	Back	Home	Rotate
onCreate				
onStart				
onSaveInstanceState				
onResume				
onPause				
onStop				
onRestoreInstanceState				
onDestroy				

- Fill the above table by putting numbers staring from 1 … according the order of the calls
- Do it for the 4 difference scenarios

Basic Calculator

- Provide both portrait and landscape layouts variations
- Persist all data during orientation change
- Use Toast to display error message for these two cases:
  - Divide by zero
  - Pressing operator when operands are empty


LinearLayout
LinearLayout
Exercise 3: Simple calculator
operand1
234
EditText
4 Buttons
+
-
*
/
operand2
567
TextView
=
TextView