Distributed Systems HS2017 – Android Tutorial

Getting Started

Start a new Android Project	
• Configure your new project	Application Name: `Android Tutorial` (will be the name when managing applications) Company Domain: `<nethz-login>.vs.inf.ethz.ch` Package name: `ch.ethz.inf.vs.<nethz-login>.androidtutorial`
• Target Android Devices	Phone/Tablet > Minimum SDK: `API 21: Android 5.0 (Lollipop)`
• Add am activity to Mobile > Empty Activity	Activity Name: `MainActivity` Layout Name: `activity_main`
Project Structure > Modules > app (Alternatively, edit build.gradle (Module app)	
• Properties	• Compile Sdk Version: 26 • Build Tools Version: 26.0.1
• Flavors	• Min Sdk Version: 21 • Target Sdk Version: 23
• Dependencies	• com.android.support:appcompat-v7:26.0.2 (Compile) * • junit:junit:4.12 (Test compile) * Requires adding Maven URL to Gradle (Top Level Gradle file. Build.gradle (Project AndroidTutorial): `allprojects { repositories { jcenter() maven { url 'https://maven.google.com' } } }`
Create virtual device: Nexus 5	
• Configure an AVD • Start emulator • Run as > Android Application	System Image: API 23. ABI: x86_64 Emulated Performance: Hardware RAM: 768 MB. VM heap: 64 MB SD Card: Studio-managed 20 MB
res/layout/activity_main.xml	
• Check palette • Play with preview	Replace ConstraintLayout with LinearLayout
res/values/strings.xml	
• Use frontend to add new strings or edit XML • Create *hello_world*	`strings.xml` `<string name="hello_world">This is VS!</string>`

• Replace literal string with @string/<name>	layout/activity_main.xml `android:text="@string/hello_world"`
src/…/MainActivity.java	
• onCreate()	State change handlers are @Override → **always remember to call super first**! The layout in `activity_main.xml` is set via constant in generated resource class R

• Add automatic ID to TextView: *@+id/text_main*
The *+* says "create an automatic ID"
• Change text via code in MainActivity.java
	layout/activity_main.xml
	`<TextView android:id="@+id/text_main"`
	MainActivity.java/onCreate():
	`TextView text = (TextView) findViewById(R.id.text_main);`
	`text.setText("I should not do it this way!");`

• Add new string to XML
• Update setText() to use the string resource
	strings.xml
	`<string name="welcome">This is the correct way.</string>`
	MainActivity.java
	`text.setText(R.string.welcome);`

Debugging with UI elements	

• Set breakpoint at different setText()
• Run debug
• Step through with F8 → no output
	MainActivity.java
	`text.setText(R.string.hello_world);`
	`text.setText(R.string.app_name);`
	`text.setText(R.string.welcome);`

Debugging with logcat	

• Use `android.util.Log` instead
VERBOSE > DEBUG > INFO > WARN > ERROR > ASSERT
• Put Log call after each `setText()`
• Create a LogCat filter on tag
	MainActivity.java
	`public static final String ACTIVITY_TAG = "### Main ###";`
	
	`Log.d(ACTIVITY_TAG, "1");`

Buttons and OnClick Listeners

Extend layout

- Change layout to LinerLayout (vertical)
- Add button *@+id/btn_test* “Click me”
- ID and string naming convention: [a-z0-9_] (general for Android-XML identifiers)

layout/activity_main.xml

```
<LinearLayout
    ...
    android:orientation="vertical"

<Button android:id="@+id/btn_test"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/btn_click"/>
```

strings.xml

```
<string name="btn_click">Click me</string>
```

Listener

- Add string *@string/btn_clicked* “Clicked”
- Implement onClickListener (IDE: Code -> Implement Methods…)
- Register OnClickListener

strings.xml

```
<string name="btn_clicked">Clicked</string>
```

MainActivity.java

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    ...
    protected void onCreate(Bundle savedInstanceState) {
        ...
        Button btn_test = (Button) findViewById(R.id.btn_test);
        btn_test.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        ((Button) v).setText(R.string.btn_clicked);
    }
}
```

- Add button *@+id/btn_action* “Action”
- Register OnClickListener

layout/activity_main.xml

```
<Button android:id="@+id/btn_action"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/btn_click"/>
```

MainActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    ...
    Button btn_test = (Button) findViewById(R.id.btn_test);
    btn_test.setOnClickListener(this);
    Button btn_action = (Button) findViewById(R.id.btn_action);
    btn_action.setOnClickListener(this);
}
```

- Add branching with switch-case for individual actions

strings.xml

```
<string name="btn_running">Running</string>
```

MainActivity.java

```
@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.btn_test:
            ((Button) v).setText(R.string.btn_clicked);
            break;
        case R.id.btn_action:
            ((Button) v).setText(R.string.btn_running);
            break;
    }
}
```

XML linked Listener

- Add `android:onClick` to XML (since 1.6)
- Implement functions (depending on the name specified in `android:onClick`
- Remember to remove `setOnClickListener()`

layout/activity_main.xml

```
<Button android:id="@+id/btn_test"
    ...
    android:onClick="onClickTest"/>
<Button android:id="@+id/btn_action"
    ...
    android:onClick="onClickAction"/>
```

MainActivity.java

```
public void onClickTest(View v) {
    ((Button) v).setText(R.string.btn_clicked);
}

public void onClickAction(View v) {
    ((Button) v).setText(R.string.btn_running);
}
```

Toggle button

- Add ToggleButton *@+id/btn_toggle* "Stopped"
- Add string *btn_stopped*

layout/activity_main.xml

```
<ToggleButton android:id="@+id/btn_toggle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textOff="@string/btn_stopped"
    android:textOn="@string/btn_running"
    android:onClick="onClickToggle"/>
```

strings.xml

```
<string name="btn_stopped">Stopped</string>
```

MainActivity.java

```
public void onClickToggle(View v) {
    ToggleButton tb = (ToggleButton) v;
    if (tb.isChecked()) {
        Log.d(ACTIVITY_TAG, "Toggle ON");
    } else {
        Log.d(ACTIVITY_TAG, "Toggle OFF");
    }
}
```

Actuation and Permissions

New Activity, Intents

- Create new Activity: File > New > Activity > Empty Activity
  Name: ActuatorsActivity
- Replace again with LinearLayout, add orientation
- Play with back and home buttons
- Notice: App resumes last activity when launched from phone menu after home button was used

layout/activity_actuators.xml

```
<TextView android:id="@+id/txt_actuators"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello_world"/>
```

MainActivity.java/onClickTest():

```
this.startActivity(new Intent(this, ActuatorsActivity.class));
```

Vibrate

- Add button *@+id/btn_vibrate* "Vibrate"
- Add and link onClickVibrate() method

ActuatorsActivity.java

```
public void onClickVibrate(View v) {
    Vibrator vib = (Vibrator) getSystemService(VIBRATOR_SERVICE);
    long[] pattern = { 0, 100, 100, 200, 100, 100 };
    vib.vibrate(pattern, -1);
}
```

- Run → crash → why?
- Add uses-permission to Manifest

AndroidManifest.xml

```
<uses-permission android:name="android.permission.VIBRATE"></uses-permission
```

Seekbar

- Add SeekBar to XML
- Make vib a member
- Anonymous inline implementation of OnSeekBarChangeListener (use anonymous classes only with care!)
- Keep pattern in onClickVibrate
- Add duration vibrate() to onStopSeek()
- Notice: setContentView() before findViewById()

layout/activity_actuators.xml

```
<SeekBar android:id="@+id/seek_duration"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android android:layout_margin="20sp"
    android:max="100"
    android:progress="50"/>
```

ActuatorsActivity.java

```
private Vibrator vib = null;
private int duration = 50;
```

ActuatorsActivity.java/onCreate():

```
vib = (Vibrator) getSystemService(VIBRATOR_SERVICE);
SeekBar seekBar = (SeekBar) findViewById(R.id.seek_duration);
seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
        duration = progress;
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
```

```
    public void onStopTrackingTouch(SeekBar seekBar) {
        vib.vibrate(duration*10);
    }
});
```

Media/Sound

- Add button *@+id/btn_sound* “Play”
- Implement and link onClickSound()
  Use `MediaPlayer`
- Add file sound.mp3 to `res/raw/` directory

strings.xml

```
<string name="sound">Play</string>
```

layout/activity_actuators.xml

```
<Button android:id="@+id/btn_sound"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/sound"/>
```

ActuatorsActivity.java

```
public void onClickSound(View v) {
    MediaPlayer mp = MediaPlayer.create(this, R.raw.sound);
    mp.setVolume(1.0f, 1.0f);
    mp.start();
}
```

- Change to looping player
- Make `mp` a member
- Add file loop.mp3 to `res/raw/` directory
- Check `isPlaying()` for action

ActuatorsActivity.java

```
private MediaPlayer mp = null;

private void initPlayer() {
    mp = MediaPlayer.create(this, R.raw.loop);
    mp.setLooping(true);
}

protected void onCreate(Bundle savedInstanceState) {
    ...
    initPlayer();
}

public void onClickSound(View v) {
    if (!mp.isPlaying()) {
        mp.start();
        if (mp.isLooping()) {
            ((Button) v).setText(R.string.btn_running);
        }
    } else {
        mp.pause();
        ((Button) v).setText(R.string.sound);
    }
}
```

Add menu and settings

- Add menu item for settings
- Implement `onCreateOptionsMenu()`
- Implement `onOptionsItemSelected()`
- Add preference for audio looping

menu/menu_actuators.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menu_settings"
        android:title="@string/menu_settings"
        android:orderInCategory="1"/>
```

- Add Settings Activity and Fragment and register Activity in Manifest
- Implement `SharedPreferenceChangeListener`
- Register `ActuatorsActovity` as `OnSharedPreferenceChangeListener`
- Add `loop` argument to `initPlayer()`

```
</menu>
```

ActuatorsActivity.java

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_actuators, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menu_settings:
            Intent myIntent = new Intent(this, SettingsActivity.class);
            this.startActivity(myIntent);
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

xml/preferences.xml

```
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">
    <CheckBoxPreference
        android:key="audio_loop"
        android:title="@string/setting_loop"
        android:defaultValue="false"
        />
</PreferenceScreen>
```

SettingsActivity.java

```
public class SettingsActivity extends AppCompatActivity {

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

        getFragmentManager().beginTransaction()
                .replace(android.R.id.content, new SettingsFragment())
                .commit();

    }
}
```

SettingsFragment.java

```
public class SettingsFragment extends PreferenceFragment {

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

        addPreferencesFromResource(R.xml.preferences);
    }
```

```
}
```

ActuatorsActivity.java

```
public class ActuatorsActivity extends AppCompatActivity implements
SharedPreferences.OnSharedPreferenceChangeListener {
    private final String KEY_PREF_AUDIO_LOOP = "audio_loop";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        PreferenceManager.getDefaultSharedPreferences(this)
            .registerOnSharedPreferenceChangeListener(this);

        initPlayer(false);
    }
    ...
    @Override
    public void onSharedPreferenceChanged(SharedPreferences sharedPreferences, String key) {
        if (key.equals(KEY_PREF_AUDIO_LOOP)) {
            SharedPreferences sharedPref = PreferenceManager
                .getDefaultSharedPreferences(this);
            boolean loop = sharedPref.getBoolean(KEY_PREF_AUDIO_LOOP, true);
            initPlayer(loop);
        }
    }
    ...
    private void initPlayer(boolean loop) {
        if (loop)
            mp = MediaPlayer.create(this, R.raw.loop);
        else
            mp = MediaPlayer.create(this, R.raw.sound);
        mp.setLooping(loop);
    }

}
```

Flashlight (optional as device-specific)

- Add title TextView “Flashlight” (paddingTop)
- Add ToggleButton `@+id/btn_toggle` (no text)
- Add Camera member
- Implement and link `onClickTorch()`
- Add uses-permission
- Some devices require `cam.setPreviewDisplay()` with `SurfaceTexture` and/or `SurfaceHolder` and `cam.startPreview();` e.g., Nexus 5

layout/activity_actuators.xml

```
<ToggleButton android:id="@+id/btn_torch"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textOn="@string/btn_torch_on"
    android:textOff="@string/btn_torch_off"
    android:onClick="onClickTorch"/>
```

ActuatorsActivity.java

```
import android.hardware.Camera;
private Camera cam = null;
public void onClickTorch(View v) {
    ToggleButton tb = (ToggleButton) findViewById(R.id.btn_torch);
    if (tb.isChecked()) {
        cam = Camera.open();
        Camera.Parameters parameters = cam.getParameters();
```

```
        parameters.setFlashMode(Camera.Parameters.FLASH_MODE_TORCH);
        cam.setParameters(parameters);
        cam.startPreview();
    } else {
        cam.release();
        cam = null;
    }
}
```

- Goes off or crashes when rotating screen: Add release to `onPause()`
- Display a `Toast`
- Also allows other apps to access camera when switching apps
- See transition diagrams from introduction

ActuatorsActivity.java

```
@Override
protected void onResume() {
    super.onResume();
    ToggleButton tb = (ToggleButton) findViewById(R.id.btn_torch);
    tb.setChecked(false);
}

@Override
public void onPause() {
    super.onPause();

    if (cam!=null) {
        cam.release();
        cam = null;
        Toast.makeText(this, "Camera released", Toast.LENGTH_LONG).show();
    }
}
```

Async Task

AsyncTask

- Note: Do not do heavy processing in onCreate()
- Never do blocking I/O on UI/main thread
- Create new Activity: `WorkerActivity`
- Add ProgressBar: *`@+id/progress_bar`*
- Add id to TextView: *`@+id/txt_progress`*
- Extend AsyncTask<Integer, Integer, Void>
- Add string *`@string/done`* "Done"
- Execute it in onCreate()
- Link activity to the action button in MainActivity
- Make sure to call `publishProgress()` when **updating the GUI** in `onProgressUpdate()`
- Task: Loop with *N* = 20 steps
- Show progress in percent

MainActivity.java:

```
public void onClickAction(View v) {
        Intent myIntent = new Intent(this, WorkerActivity.class);
        this.startActivity(myIntent);
    }
```

layout/activity_worker.xml

```
<TextView android:id="@+id/txt_progress"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="0"/>

<ProgressBar android:id="@+id/progress_bar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

WorkerActivity.java

```
public class WorkerActivity extends AppCompatActivity {

    private ProgressBar progressBar;
    private TextView textView;

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

        progressBar = (ProgressBar) findViewById(R.id.progress_bar);
        textView = (TextView) findViewById(R.id.txt_progress);

        new MyWorker().execute(20);
    }

    private int calculateProgress(int index, int max) {
        return index * 100 / max;
    }

    class MyWorker extends AsyncTask<Integer, Integer, Void> {
        private int progress;
        @Override
        protected void onPreExecute() {
            progressBar.setMax(100);
            progressBar.setProgress(0);
        }

        @Override
        protected Void doInBackground(Integer... params) {
            for (int i = 0; i < params[0]; i++) {
```

```
                try {
                    Thread.sleep(100);
                } catch (InterruptedException e) { }
                publishProgress(i, params[0]); // run onProgressUpdate on UI thread
            }
            return null;
        }

        @Override
        protected void onProgressUpdate(final Integer... values) {
            progress = calculateProgress(values[0], values[1]);
            Log.d("###", Integer.toString(progress));
            textView.setText(Integer.toString(progress));
            progressBar.incrementProgressBy(progress);
        }

        @Override
        protected void onPostExecute(final Void result) {
            textView.setText(R.string.done);
        }
    }
}
```

JUnit Tests

JUnit

- Make `calculateProgress()` in `WorkerActivity public static`
- Add Tests in `ExampleUnitTests`
- Add Run Configuration: Android JUnit
- Classpath of module: app
- Class: ExampleUnitTest
- Run

ExampleUnitTests.java:

```
public class ExampleUnitTest {
    @Test
    public void integer_fraction_is_correct() throws Exception {
        assertEquals(WorkerActivity.calculateProgress(7, 10), 70);
    }

    @Test
    public void float_fraction_is_correct() throws Exception {
        assertEquals(WorkerActivity.calculateProgress(2, 3), 66);
    }

    @Test
    public void max_is_hundred() throws Exception {
        assertEquals(WorkerActivity.calculateProgress(3, 1), 100);
    }
}
```