Distributed Systems HS2010 – Android live hacking

General hints

- Uninstall Application when switch to a different development PC
- Do not press Run with XML files
- Often no connection to debugger on emulator → restart emulator

Create Android project	
• Build target: Android 2.2 • Create Activity: `Main`	
src/Main.java	
• onCreate(): @Override → call super • setContentView():`R.layout.main`	
res/layout/main.xml	
• `fill_parent` –API8→ `match_parent` • Frontend • Strings: `@string/<name>`	`main.xml` `android:layout_width="match_parent"` http://www.droiddraw.org/
res/values/strings.xml	
• Frontend (cumbersome) • `app_name` from "Create project"	`strings.xml` `<string name="app_name">Live Hacking Demo</string>`
Create virtual device	
• Config: RAM 576 MB • Config: Keyboard no (default yes) • Config: Camera yes (default no) • Config: Trackball yes (otherwise missing controls) • Run emulator • Run App	
Play with strings	
• Change `hello` in XML	`strings.xml` `<string name="hello">This is VS!</string>`

• Add ID to TextView: `@+id/text`
• Change text via code in Main.java
	`main.xml` `android:id="@+id/text_main"`
	
	`Main.java`
	`TextView text = (TextView) findViewById(R.id.text);`
	`text.setText("I shound not do it this way!");`

• Add new string to xml / new setText()	`strings.xml` `<string name="welcome">That is the official way!</string>` `Main.java` `text.setText(R.string.welcome);`

gen/R.java	
• res folder as handle ints • Classes for layouts, ids, strings	
Manifest	
• App: Icon, name on the phone • Activities of an App • Intent-filter: first upon start, appear in launcher • `minSdkVersion` from "Create project"	
Debugging	
• Set breakpoint • Run debug	`Main.java` <Ctrl+Shift+B> `text.setText("I shound not do it this way!");`
• Run on device • Notice: no break	

• Set `debuggable="true"` in Manifest
• Run on device → break
	`Manifest`
	`<application android:icon="@drawable/icon"`
	`android:label="@string/app_name" android:debuggable="true">`

Debugging with "printf()"	
• Add setText() • Step through with F6 → no output	`Main.java` `text.setText(R.string.hello);`
• Use `android.util.Log` instead • Notice: does not work on emulator	Levels: VERBOSE > DEBUG > INFO > WARN > ERROR > ASSERT `Main.java` `Log.d("### Main ###", "1");`
Change layout	

• Add button
• ID naming convetion: `btn_test`
c.f., `R.layout.main`
	`main.xml`
	`<Button android:id="@+id/btn_test"`
	`android:layout_width="match_parent"`
	`android:layout_height="wrap_content"`
	`android:text="@string/btn_test"`
	`/>`
	
	`strings.xml` `<string name="btn_test">Click me</string>`

Inline Listener	

• Add inline on-the-fly Listener
• Quick & dirty
• Multiple per class possible
	Main.java
	`findViewById(R.id.btn_test).setOnClickListener( new OnClickListener() {`
	`  @Override`
	`  public void onClick(View v) {`
	`  ((Button)v).setText("Clicked");`
	`  }`
	`} );`

• Add button "Start"
• Store Listener in variable
• Assign to both buttons
• For reuse
	main.xml
	`<Button android:id="@+id/btn_action"`
	`  android:layout_width="match_parent"`
	`  android:layout_height="wrap_content"`
	`  android:text="@string/btn_action"`
	`  />`
	strings.xml `<string name="btn_action">Action</string>`
	
	Main.java
	`OnClickListener btnListener = new OnClickListener() {`
	`  @Override`
	`  public void onClick(View v) {`
	`  ((Button)v).setText("Clicked");`
	`  }`
	`};`
	`findViewById(R.id.btn_test).setOnClickListener(btnListener);`
	`findViewById(R.id.btn_action).setOnClickListener(btnListener);`

• Add branching with switch()-case for individual action	Main.java `onClick():`
	`switch (v.getId()) {`
	`case R.id.btn_test:`
	`  ((Button)v).setText("Clicked"); break;`
	`case R.id.btn_start:`
	`  ((Button)v).setText("Running"); break;`
	`}`

• Use implements Listener (with branching)
• Reusable
• Compact
• Centralized
• Only one listener per class
	Main.java
	`public class Main extends Activity implements OnClickListener {`
	
	`@Override`
	`public void onClick(View v) {`
	`  switch (v.getId()) {`
	`  case R.id.btn_test:`
	`  ((Button)v).setText("Clicked (this)");`
	`  break;`
	`  case R.id.btn_start:`
	`  ((Button)v).setText("Running (this)");`
	`  break;`
	`  }`
	`}`

XML linked Listener	

• Add `android:onClick` to XML (since 1.6)
• Implement functions
• Convenient
	`main.xml` `android:onClick="onClickTest"`
	
	`Main.java`
	`public void onClickTest(View v) {`
	`  ((Button)v).setText("Clicked (XML)");`
	`}`
	
	`public void onClickAction(View v) {`
	`  ((Button)v).setText("Running (XML)");`
	`}`

Other buttons	

• ToggleButton	`main.xml`
	`<ToggleButton android:id="@+id/btn_start"`
	`  android:layout_width="wrap_content"`
	`  android:layout_height="wrap_content"`
	`  android:text="@string/btn_action"`
	`  android:onClick="onClickStart"`
	`  />`

• `android:text` not supported
• Add off/on string to XML
• isChecked() - changed before onClick oder after?
	`strings.xml`
	`<string name="btn_toggle_off">Stopped</string>`
	`<string name="btn_toggle_on">Running</string>`
	
	`Main.java` `onCreate():`
	`((Button)findViewById(R.id.btn_start)).setText(R.string.btn_toggle_off);`
	
	`Main.java`
	`public void onClickStart(View v) {`
	`  ToggleButton tb = (ToggleButton) v;`
	`  if (tb.isChecked())`
	`  ((Button)v).setText(R.string.btn_toggle_off);`
	`  else`
	`  ((Button)v).setText(R.string.btn_toggle_on);`
	`}`

<table>
<tr><th colspan="2">New Activity, Intents</th></tr>
<tr><td>

- Create new Class: `Actuators`
- Same package
- Add onCreate() → call super
- Manifest: Add Activity .Actuators (e.g., via frontend)
- Add layout -> new Android XML File
- Add TextView, play with attributes
- Add string with HTML formatting
- Add Intent to launch new Activity

</td><td>

Actuators.java

```
package ch.ethz.inf.vs.android.g<##>.<nethz-login>.livehacking;

import android.app.Activity;
import android.os.Bundle;

public class Actuators extends Activity {
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.actuators);
	}
}
```

actuators.xml

```
<TextView
	android:id="@+id/txt_actuators"
	android:layout_width="match_parent"
	android:layout_height="wrap_content"
	android:gravity="center_horizontal"
	android:textStyle="bold"
	android:text="@string/txt_actuators"
	/>
```

strings.xml

```
<string name="txt_actuators">Actuators <tt>Activity</tt><br /><tt>TextView</tt>s
<i>understand</i> HTML <b>formatting</b>!
```

Main.java

```
public void onClickTest(View v) {
	((Button)v).setText("Clicked (XML)");

	Intent myIntent = new Intent(this, Actuators.class);
	this.startActivity(myIntent);
}
```

</td></tr>
<tr><td>

- Notice: no `<br />`, text style only
- Play with back and home buttons
- Notice: App resumes last activity when launched from phone menu after home button was used

</td><td>

strings.xml

```
<string name="txt_actuators">Actuators <tt>Activity</tt><br /><tt>TextView</tt>s
<i>understand</i> HTML <b>formatting</b>!\n\nBut no HTML breaks</string>
```

</td></tr>
</table>

<table>
<tr><td colspan="2">Vibrator</td></tr>
<tr><td>

- Fix break with \n
- Add button
- Add onClickVibrate() method
- Index at which to start to repeat!
Not how often.

</td><td>

actuators.xml

```
<Button
      android:id="@+id/btn_vibrate"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="@string/btn_vibrate"
      android:onClick="onClickVibrate"
      />
```

Actuators.java

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

</td></tr>
<tr><td>

- Notice: where is the button → missing orient.
- Run → crash → why?

</td><td>

actuators.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="vertical"
```

</td></tr>
<tr><td>

- Add uses-permission

</td><td>

Manifest

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

</td></tr>
<tr><td colspan="2">Seekbar</td></tr>
<tr><td>

- Add SeekBar to XML
- Make vib a member
- Add inline Listener
- Keep pattern in onClickVibrate
- Add duration vibrate() to onStopSeek()
- Notice:setContentView() before findViewById()

</td><td>

actuators.xml

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

Actuators.java Members:

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

Actuators.java onCreate():

```
SeekBar seekDuration = (SeekBar) findViewById(R.id.seek_duration);
seekDuration.setOnSeekBarChangeListener( new SeekBar.OnSeekBarChangeListener() {
      @Override
      public void onProgressChanged(SeekBar seekBar, int progress, boolean user) {
            duration = progress;
      }
      public void onStartTrackingTouch(SeekBar seekBar) {}
      public void onStopTrackingTouch(SeekBar seekBar) {
            vib.vibrate(duration*10);
      }
});
```

</td></tr>
</table>

Flashlight

- Add title TextViews (paddingTop)
- Look up unit *dip*
- Add ToggleButton (no text)
- Add Camera member
- Implement onClickFlash()
- Add uses-permission
- Notice: works only since 2.2

actuators.xml

```
<TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:textStyle="bold"
      android:text="@string/txt_flashlight"
      android:paddingTop="30dip"
      />
<ToggleButton
      android:id="@+id/btn_flash"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:onClick="onClickFlash"
      />
```

Actuators.java

```
import android.hardware.Camera;
private Camera cam = null;

public void onClickFlash(View v) {
      ToggleButton tb = (ToggleButton) v;
      if (tb.isChecked()) {
            cam = Camera.open();
            Camera.Parameters parameters = cam.getParameters();
            parameters.setFlashMode(Camera.Parameters.FLASH_MODE_TORCH);
            cam.setParameters(parameters);
      } else {
            cam.release();
      }
}
```

- Check available modes: `list!=null` and for `TORCH`
- Try camera
- Notice: camera still open
  Camera App will not start

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

Actuators.java

```
List<String> list = parameters.getSupportedFlashModes();
if (list==null || !list.contains(Camera.Parameters.FLASH_MODE_TORCH)) {
      Log.i("Actuators", "TORCH mode not supported");
      cam.release();
      tb.setChecked(false);
      ((TextView)findViewById(R.id.txt_flash_modes)).setText("No camera support");
      return;
}
((TextView)findViewById(R.id.txt_flash_modes)).setText(list.toString());
```

- And release to `onPause()`
- Display a `Toast`
- Notice: transition diagrams from introduction

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

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

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

	((ToggleButton)findViewById(R.id.btn_flash)).setChecked(false);
}
```

Media/Sound

- Add title and button
- Implement onClickSound()
  Use `MediaPlayer`
- Add soundfiles to `res/raw/` directory

`actuators.xml`

```
<TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center_horizontal"
      android:textStyle="bold"
      android:text="@string/txt_sound"
      android:paddingTop="30dip"
      />
<Button
      android:id="@+id/btn_sound"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="@string/btn_sound"
      android:onClick="onClickSound"
      />
```

`Actuators.java`

```
public void onClickSound(View v) {
      MediaPlayer mp = MediaPlayer.create(this, R.raw.ready);
      mp.setVolume(10.0f, 10.0f);
      mp.start();
}
```

- Change to looping player
- Make `mp` a member
- Check `isPlaying()` for action
- Reset player after stopping: prepareAsync()

`Actuators.java`

```
private MediaPlayer mp = null;
```

`Actuators.java` onCreate():

```
mp = MediaPlayer.create(this, R.raw.work);
mp.setLooping(true);

public void onClickSound(View v) {
      if (!mp.isPlaying()) {
            mp.start();
            if (mp.isLooping()) {
                  ((Button)v).setText(R.string.btn_sound_stop);
            }
      } else {
            mp.stop();
            try {
                  mp.prepareAsync();
            } catch (IllegalStateException e) {
                        // TODO Auto-generated catch block
            }
            ((Button)v).setText(R.string.btn_sound);
      }
}
```

Menu button

- Add code for menu button
- Options: looping or once
- Add menu option: back
  `finish()` ends Activity

Actuators.java

```
private static final int LOOPING_MENU_ID = Menu.FIRST;
private static final int ONCE_MENU_ID = Menu.FIRST + 1;
private static final int BACK_MENU_ID = Menu.FIRST + 2;

@Override
public boolean onCreateOptionsMenu(Menu menu) {
      super.onCreateOptionsMenu(menu);

      menu.add(0, LOOPING_MENU_ID, 0, "Looping");
      menu.add(0, ONCE_MENU_ID, 0, "Once");
      menu.add(0, BACK_MENU_ID, 0, "Back");

      return true;
}

@Override
public boolean onPrepareOptionsMenu(Menu menu) {
      super.onPrepareOptionsMenu(menu);
      if (mp.isPlaying()) return false;
      else return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {

      switch (item.getItemId()) {
            case LOOPING_MENU_ID:
                  mp = MediaPlayer.create(this, R.raw.work);
                  mp.setLooping(true);
                  return true;
            case ONCE_MENU_ID:
                  mp = MediaPlayer.create(this, R.raw.ready);
                  mp.setLooping(false);
                  return true;
            case BACK_MENU_ID:
                  finish();
                  return true;
      }
      return super.onOptionsItemSelected(item);
}
```

Fancy Activities	

• Create `Dialogs` Activity
• Set dialogs layout to TextView
• Set Translucent:
WindowManager
Manifest theme
• Change `onClickAction()`
	`dialogs.xml`
	`<?xml version="1.0" encoding="utf-8"?>`
	`<TextView xmlns:android="http://schemas.android.com/apk/res/android"`
	`android:id="@+id/txt_dialogs"`
	`android:layout_width="match_parent"`
	`android:layout_height="match_parent"`
	`android:gravity="center_vertical|center_horizontal"`
	`android:text="@string/txt_dialogs"`
	`/>`
	
	`strings.xml`
	`<string name="txt_dialogs">Isn\'t this <i>fancy</i>?\n\nA blurry <tt>Activity</tt> in the background</string>`
	
	`Dialogs.java` `onCreate():`
	`getWindow().setFlags(WindowManager.LayoutParams.FLAG_BLUR_BEHIND, WindowManager.LayoutParams.FLAG_BLUR_BEHIND);`
	
	`setContentView(R.layout.dialogs);`
	
	`Manifest`
	`<activity android:name=".Dialogs" android:theme="@android:style/Theme.Translucent"></activity>`
	
	`Main.java`
	`public void onClickAction(View v) {`
	`Intent myIntent = new Intent(this, Dialogs.class);`
	`this.startActivity(myIntent);`
	`}`

- Change theme to dialog
- Add icon
- Notice: `android.R.` provides default resources

Manifest

```
<activity android:name=".Dialogs"
android:theme="@android:style/Theme.Dialog"></activity>
```

Dialogs.java onCreate():

```
requestWindowFeature(Window.FEATURE_LEFT_ICON); // must be before setContentView()
setContentView(R.layout.dialogs);
// Set icon for dialog
getWindow().setFeatureDrawableResource(Window.FEATURE_LEFT_ICON,
                                                android.R.drawable.ic_dialog_info);
```

- Send data with Intent
- Append to dialogs text
- Notice: `Html.fromHtml()` supports
- Notice only for custom dialogs

Main.java

```
public void onClickAction(View v) {
      Intent myIntent = new Intent(this, Dialogs.class);
      myIntent.putExtra("text", "This text was sent with the <tt>Intent</tt>");
      this.startActivity(myIntent);
}
```

Dialogs.java onCreate():

```
((TextView) findViewById(R.id.txt_dialogs)).append(Html.fromHtml("<br />" +
getIntent().getExtras().getString("text")));
```

- Add button for real dialog (OK/Cancel)
- Continued…

dialogs.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="wrap_content" android:layout_height="wrap_content"
      android:orientation="vertical"
      android:gravity="center_horizontal">
<TextView android:id="@+id/txt_dialogs"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:gravity="center_vertical|center_horizontal"
      android:text="@string/txt_dialogs"
      />
<Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/btn_dialog"
      android:onClick="onClick"
      />
</LinearLayout>
```

strings.xml

```
<string name="btn_dialog">Real dialog</string>
```

- …
- Use these dialogs by default
- Notice: builder pattern

Dialogs.java

```
private static final int DIALOG_YES_NO_MESSAGE = 1;

@Override
protected Dialog onCreateDialog(int id) {

switch (id) {
case DIALOG_YES_NO_MESSAGE:
      return new AlertDialog.Builder(Dialogs.this)
            .setIcon(android.R.drawable.ic_dialog_alert)
            .setTitle("Title: Alert")
            .setMessage("A simple alert dialog")
            .setPositiveButton("OK/Yes", new DialogInterface.OnClickListener() {
                  public void onClick(DialogInterface dialog, int whichButton) {
                        /* User clicked OK so do some stuff */
                  }
            })
            .setNegativeButton("Cancel/No", new DialogInterface.OnClickListener() {
                  public void onClick(DialogInterface dialog, int whichButton) {
                        /* User clicked Cancel so do some stuff */
                  }
            })
            .create();
      }
      return null;
}

public void onClick(View v) {
      showDialog(DIALOG_YES_NO_MESSAGE); // ID
}
```

- Add progress dialog ID and case
- Add dialog members
- Handler that progresses
  Init in onCreate()
- Trigger handler in onClick() via message

Dialogs.java

```
private ProgressDialog mProgressDialog;
private int mProgress;
private Handler mProgressHandler;

mProgressHandler = new Handler() {
	@Override
	public void handleMessage(Message msg) {
		super.handleMessage(msg);
		if (mProgress >= 100) {
			mProgressDialog.dismiss();
		} else {
			mProgress++;
			mProgressDialog.incrementProgressBy(1);
			mProgressHandler.sendEmptyMessageDelayed(0, 100);
		}
	}
};

private static final int DIALOG_PROGRESS = 2;

case DIALOG_PROGRESS:
	mProgressDialog = new ProgressDialog(Dialogs.this);
	mProgressDialog.setIcon(android.R.drawable.ic_dialog_alert);
	mProgressDialog.setTitle("Title");
	mProgressDialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
	mProgressDialog.setMax(100);
	mProgressDialog.setButton("Button", new DialogInterface.OnClickListener() {
		public void onClick(DialogInterface dialog, int whichButton) {
			/* User clicked Yes so do some stuff */
		}
	});
	mProgressDialog.setButton2("Cancel", new DialogInterface.OnClickListener() {
		public void onClick(DialogInterface dialog, int whichButton) {
			/* User clicked No so do some stuff */
		}
	});
	return mProgressDialog;

public void onClick(View v) {
	showDialog(DIALOG_PROGRESS);
	mProgress = 0;
	mProgressDialog.setProgress(0);
	mProgressHandler.sendEmptyMessage(0);
}
```