Introduction to Android

F. Mallet

Frederic.Mallet@unice.fr

Université Nice Sophia Antipolis

M1 IFI – Embedded & Mobile Computing

Android

- Objectives
  - Basic concepts: package, component, manifest
  - Activity: lifecycle
  - Model/View/Controller
- Practical: traffic light

ANDROID FUNDAMENTALS

Android Package

- Android **applications** are deployed within a .apk
  - Contains all the bytecode
  - Contains all the resources
- Android applications
  - Each app has its own Linux user ID (by default)
    - Each app is a different user
    - All the files in the .apk are accessible only by this ID
  - Each process has its own virtual machine
    - Isolation to other processes
    - By default, each app runs in its own Linux process
    - Process starts when one of its app needs to run
    - Process finishes when no app is running or need memory

Sharing information

- Possible to share data between apps
  - Assign the same Linux user ID
    - => access each other's file
  - Can run in the same Linux process
    - => must be signed with the same certificate
- Possible to ask access to device data
  - E.g., contacts, SMS, SD card, camera, …
  - Needs to have access granted before installation
    - Manifest: declare the required permissions

Components

- An *app* is a set of **components**
  - Each component is independent of the other ones
  - Some are entry points to the user
  - Some are entry points to other components
- Four types of components
  - **Activity**: *single* screen with a user interface [android.app.Activity]
    - Can reuse activities of others to build an app (camera, map, …)
  - **Service**: runs in the background to perform long-running operations (no UI) [android.app.Service]
  - **Content provider**: manages a (shared or private) set of data [android.content.ContentProvider]
  - **Broadcast receiver**: responds to system-wide announcement
    - E.g., battery is low, picture was captured
    - No UI, but may have status bar notification, or initiate a service
    - [android.content.BroadcastReceiver]

Communication between components

- Not a single entry point
  - An app has many components that can be shared between apps
  - Each app has its own Linux user ID (default)
  - Each app runs in its own process (default)
- **Intents**: asynchronous messages [android.content.Intent]
  - Between components
    - Activity or service: action to perform or data to exchange
    - Broadcast receiver: just a message “battery is low”
    - Content provider: do not use intent but *content resolver*
  - May be used to *activate* an activity, a service or a broadcast receiver
    - May belong to another *app* (then runs in the process of the other *app*)
  - May be used to convey data (e.g., result of an activity)

Explicit vs. Implicit Intents

- Explicit Intents
  - Access a component by name (fully qualified class name)
    - Internal use between components of your own app
  - Ex: Start Service **`DownloadService`**
    - ```
      Intent downloadIntent = new Intent(this, DownloadService.class);
      downloadIntent.setData(Uri.parse(fileUrl));
      startService(downloadIntent);
      ```
- Implicit Intents
  - Describe a general action to perform without selecting which app should be used
  - Ex: Share some text with someone
    - ```
      Intent sendIntent = new Intent(Intent.ACTION_SEND);
      sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
      sendIntent.setType(HTTP.PLAIN_TEXT_TYPE); // "text/plain"

      // Verify that the intent will resolve to an activity
      if (sendIntent.resolveActivity(getPackageManager()) != null)
          startActivity(sendIntent);
      ```

Manifest

- Declare the components and their properties
- AndroidManifest.xml: <manifest …> </manifest>
  - Declare the components of an application
    - <activity>, <service>, <provider>: must be declared to run
    - <receiver>: may be declared and registered at runtime
  - Identify user permissions (e.g., internet access)
  - Minimum, maximum, target API level
    - Package and classes
    - XML elements for the manifest
    - XML elements for resources
    - Allowed intents
    - Allowed permissions
  - HW/SW features used (camera, bluetooth, multitouch)
  - API libraries (other than Android)
    - E.g., Google Map Library

API Level

```
<uses-sdk android:minSdkVersion="integer" android:maxSdkVersion="integer"
          android:targetSdkVersion="integer" />
```

Platform Version	API Level	VERSION_CODE
Android 4.4	19	KITKAT
Android 4.3	18	JELLY_BEAN_MR2
Android 4.2, 4.2.2	17	JELLY_BEAN_MR1
Android 4.1, 4.1.1	16	JELLY_BEAN
Android 4.0.3, 4.0.4	15	ICE_CREAM_SANDWICH_MR1
Android 4.0, 4.0.1, 4.0.2	14	ICE_CREAM_SANDWICH
Android 3.2	13	HONEYCOMB_MR2
Android 3.1.x	12	HONEYCOMB_MR1
Android 3.0.x	11	HONEYCOMB
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD
Android 2.2.x	8	FROYO
Android 2.1.x	7	ECLAIR_MR1
Android 2.0.1	6	ECLAIR_0_1
Android 2.0	5	ECLAIR
Android 1.6	4	DONUT
Android 1.5	3	CUPCAKE
Android 1.1	2	BASE_1_1
Android 1.0	1	BASE

FIRST ANDROID APPLICATION

Simple Android Application

- Application
  - Can define new activities and reuse existing ones
  - Can define new layout or build libraries
- Activity
  - Responsible for user interactions with screens
  - MyActivity extends Activity
  - May have many interacting activities
- Layout
  - Defines the set of user interface objects and their position
  - Several definitions within an XML file

Download, Install, Set-Up

- Need to download Eclipse ADT + SDK (all in one)
  - http://developer.android.com/sdk/index.html
- Need to configure the SDK (optional)
  - Launch the SDK manager
- Need to build
  - A Device
    - Properties of the targeted device
  - An Android Virtual Device (AVD)
    - Emulator for the device

- Getting Started
  - http://developer.android.com/training/index.html
- API
  - http://developer.android.com/reference/packages.html

MAIN activity

- The main activity is identified with an **Intent Filter**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.unice.m1ifi"    android:versionCode="1" android:versionName="1.0" >
    <uses-sdk       android:minSdkVersion="8"       android:targetSdkVersion="19" />
    <application
        android:allowBackup="true"  android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="fr.unice.m1ifi.light.MainLightActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

String resources

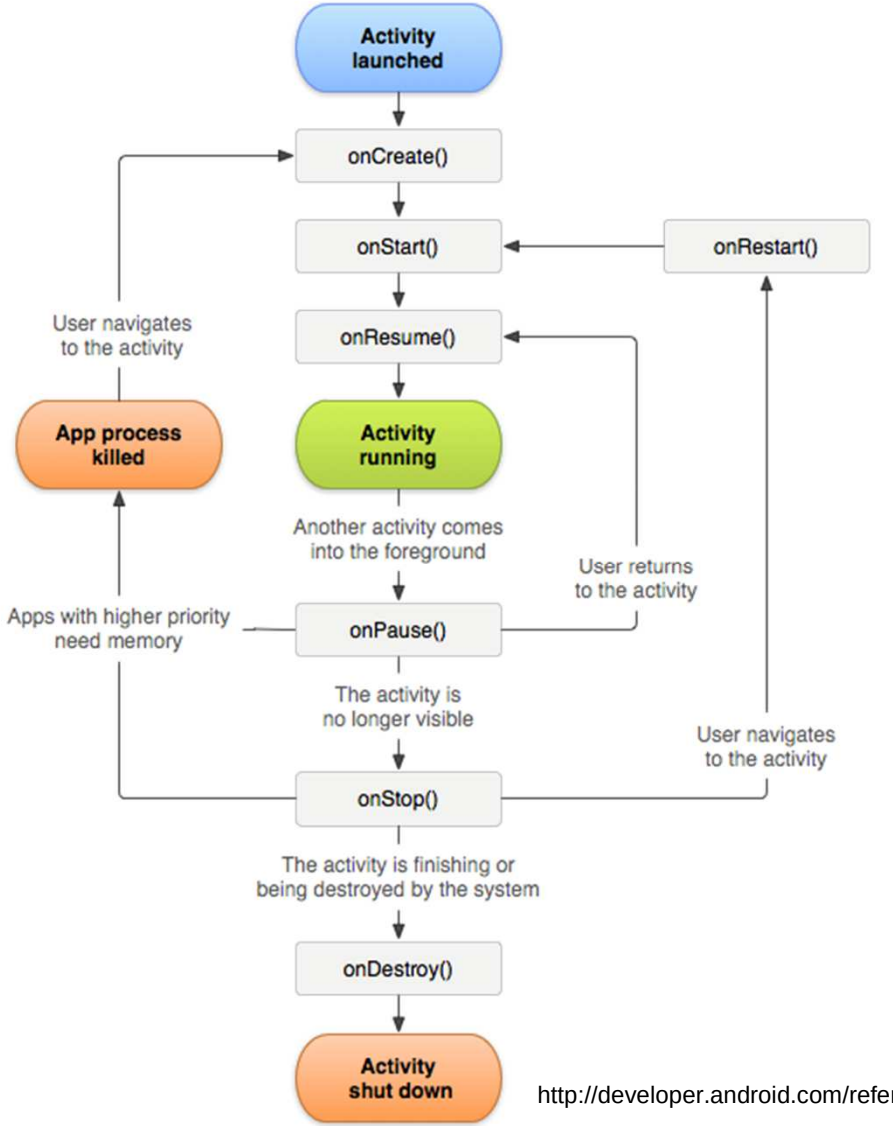
- Should be declared in separated .xml files
  - In the folder res/values (By default: res/values/strings.xml)
  - Example

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">M1 Car Light</string>
    <string name="action_settings">Settings</string>
    <string name="light_text">Pick your light</string>
    <string name="pedlight_button">Pedestrian</string>
    <string name="carlight_button">Car</string>
</resources>
```

- Reference from the manifest
  - android:label="@string/app_name"

Activity Life cycle

- Override the methods to control what happens



http://developer.android.com/reference/android/app/Activity.html#ActivityLifecycle

Activity: example

- Example of generated code

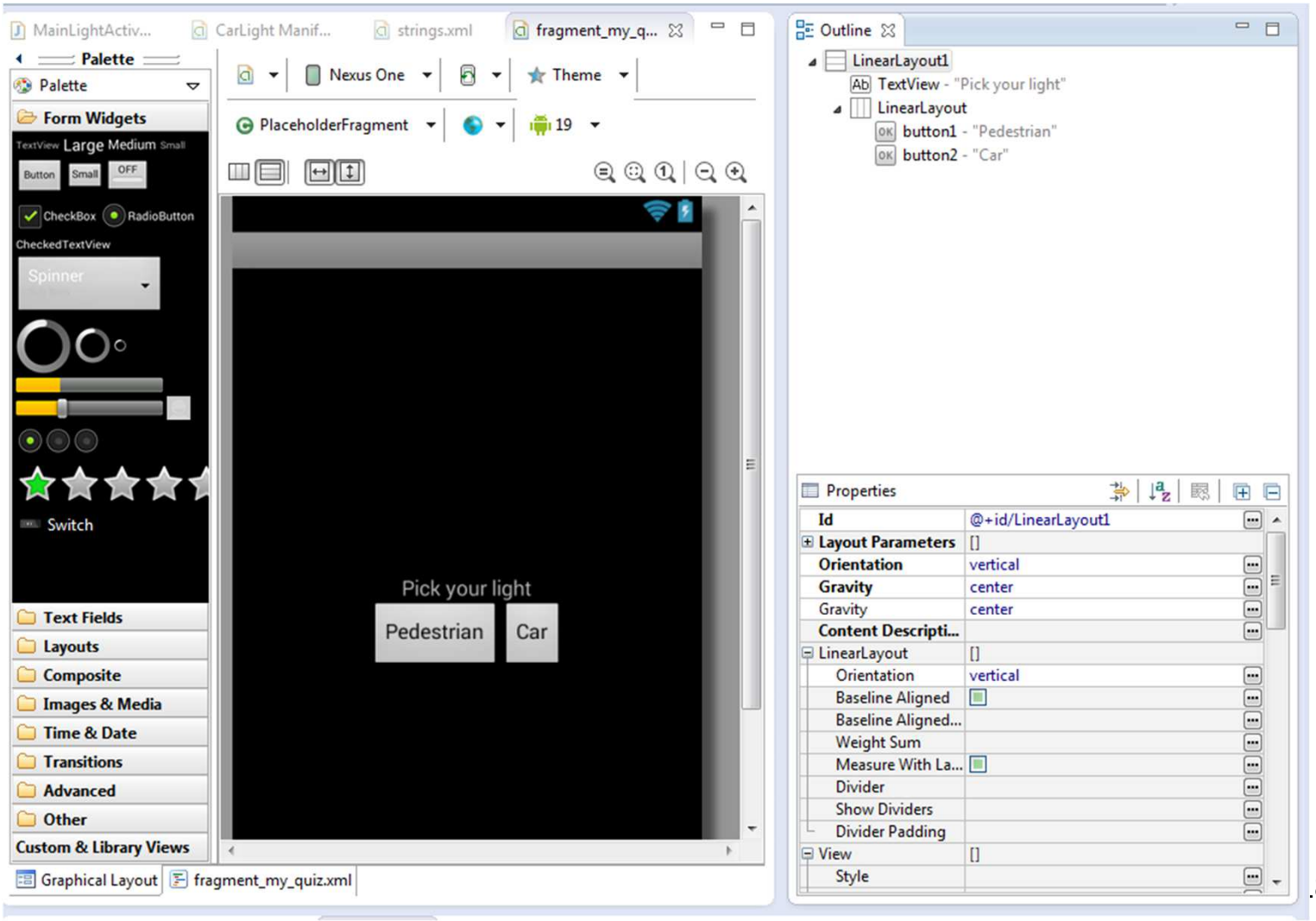
```
public class MainLightActivity extends ActionBarActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my_light);
    }

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

Refers to res/layout/activity_my_light.xml

Refers to res/menu/my_light.xml

Layout

- Use the IDE to create widgets and set up their position



Accessing widgets

- Access through Integer ID
- Declaration in the layout

```
<Button
        android:id="@+id/ped_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/pedlight_button" />
```

- When the project is built

```
package fr.unice.m1ifi.light;
public final class R {
        public static final class id {
                public static final int ped_button=0x7f05003e;
```

- From the code

```
private android.widget.Button mPedButton;
…
mPedButton = (Button)findViewById(R.id.ped_button);        (within onCreate method)
```

OnClickListener

- Use Android-specific listeners for Android Widget

```
mPedButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
                Log.d("MainLightActivity", "Click on ped button");
        }
});
```

- Log system (Use LogCat) ⇔ println
  - System.out.println

    Log.**d**(String task, String msg);
  - System.err.println

    Log.**e**(String task, String msg);

Building your own View

- Extends the class android.view.**View**

```
public class PedCanvas extends View {
   private Paint paint = new Paint(); // reuse
   public PedCanvas(Context context) {
      super(context);
   }

   @Override
   protected void onDraw(Canvas canvas) {
      super.onDraw(canvas);

      paint.setColor(Color.RED);
      paint.setStyle(Paint.Style.FILL);
      canvas.drawCircle(getWidth ()/ 2, getHeight()/2, 20, paint);
   }
}
```

Activity: example

- Example of generated code

```
public class PedLightActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new PedCanvas(this));
    }

}
```

References

- Android Developers
  - http://developer.android.com/training/index.html
- Other lectures at UNS
  - P. Renevier
  - E. Amosse
- Books
  - Beginning Android, M. L. Murphy, APress
  - Android Programming: The Big Nerd Ranch Guide, B. Phillips, B. Hardy, http://www.bignerdranch.com