1 Introduction

An Android application consists out of the following parts:

- Activity - A screen in the Android application
- Services - Background activities without UI
- Content Provider - provides data to applications, Android contains a SQLLite DB which can serve as data provider
- Broadcast Receiver - receives system messages, can be used to react to changed conditions in the system

Intends allow the application to request and / or provide services . For example the application call ask via an intent for a contact application. Application register themself via an IntentFilter. Intends are a powerful concept as they allow to create loosely coupled applications.

An Android application is described the file "AndroidManifest.xml". This files contains all activities application and the required permissions for the application. For example if the application requires network access it must be specified here. "AndroidManifest.xml" can be thought as the deployment descriptor for an Android application.

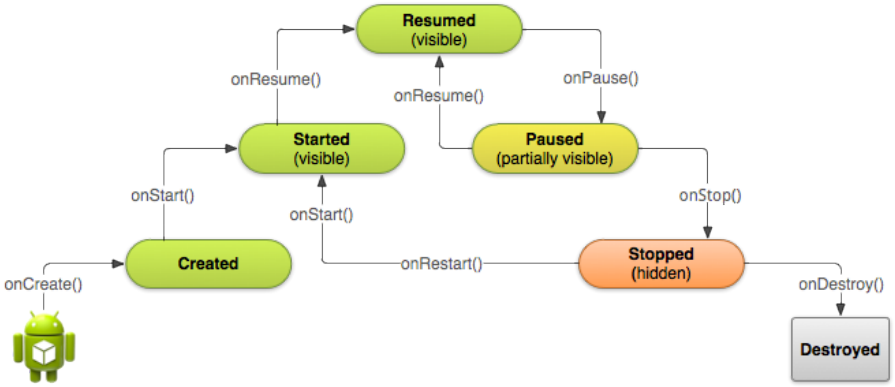
1.1 Activities and Layouts

The user interface for Activities is defined via layouts. The layout defines the UI elements, their properties and their arrangement. A layout can be defined via XML and via code at runtime. The XML way is usually preferred for a fixed layout while defining the layout via code is more flexible. You can also mix both approaches.

1.2 Activities and Lifecyle

The operating system controls the life cycle of your application. At any time the Android system may stop or destroy your application, e.g. because of an incoming call. The Android system defines a life cycle for activities via pre-defined methods. The most important methods are:

- onSaveInstanceState() - called if the activity is stopped. Used to save data so that the activity can restore its states if re-started
- onPause() - always called if the Activity ends, can be used to release ressource or save data
- onResume() - called if the Activity is re-started, can be used to initiate fields



You are encouraged to test the following two examples :

- http://developer.android.com/training/basics/activity-lifecycle/index.html
- http://www.vogella.com/tutorials/AndroidLifeCycle/article.html

2 HelloWorld [15 minutes]

- Open the android studio, /packages/mobserv/ android-studio/studio.sh
- Go the file > project structure
  - Set Android location SDK to /packages/mobserv/android-sdk (**cancel** if promoted to install an Android SDK)
  - Set the java path to /usr/jdk/latest
- Start a new Android Studio project, see http://developer.android.com/sdk/installing/create-project.html
  - App name: HelloWorld
  - Company: eurecom.fr
  - **Project location**: /home/Local_Data
  - Target: phones and tablets with min SDK API 15
  - Activity: Black Activity
  - Uncheck “use of Fragment”
  - Finish
- Build an Android Virtual Devices:
  - Tools > Android > AVD Manager or Click on AVD manager
  - Create a virtual device
  - Create a New Hardware profile or import the one in android-studio/extra/device.xml
    - Device name: myDevice
    - Screensize: 4.0
    - Resolution : 780x1280
    - RAM: 512
    - Check “Has Hardware Buttons”
    - Check “Has Hardware Keyboard”
    - Uncheck back and front camera
  - System image: marshmallow x86_64, Android 6.0 (with google API)

- Finish
- In Run > Edit Configurations, Emulator tab, Check Additional command line options and in the input field enter [1]:
  - -qemu –m 2047 -enable-kvm
  - Type the command “groups”, and make sure that you are in the kvm and libvirtd groups
- Run the application
  - Launch emulator : select “myDevice API 23”
  - You should see “HelloWorld”
- Change “HelloWorld” to “HelloMoon”, and check the results on the emulator

3 Implicit Intents (duration <60 minutes)

Android supports explicit intents and implicit intents. Explicit intent names the component while the implicit intents asked the system to perform a service without telling the system which component (java class) should do this service.
In an implicit Intent you specify

- Action: e.g. view
- URI : e.g. webpage

And the system will find an application which is registered for this event, e.g. a browser.
Useful methods:

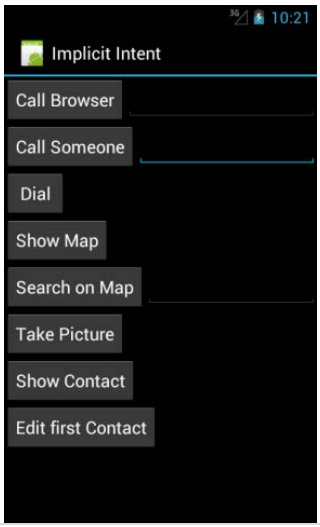
- startActivity(Intent) if you do not need a return value from the called activity
- startActivityForResult() and onActivityResult()

An intent can contain data when it requires results through the following methods:

- putExtra() if you want to add data to the intent
- getAction(), getData() and getExtras() if you want to receive the data and url from this Intent

3.1 Create a Project

1. Project name: fr.eurecom.android.implicit.intent
2. Application name : Implicit Intent
3. Package name : fr.eurecom.implicitintent
4. Create Activity : ImplicitIntent



3.2 Edit Layout

1. Add 8 Buttons:
   - Call browser
   - Call someone
   - Dial
   - Show map , position of Eurecom
   - Search map, e.g. valbonne
   - Take a picture
   - Show contact
   - Edit the first contact
2. Add 2 textboxes

- One for call browser
- One for call someone

3. Set the android:id, the android:text, and android:onClick function/handler
   - Keep the android:onClick handler the same for all the buttons, and name it ”callIntent”
4. Add three EditText for calling a browser, call someone, and search a map.

3.3 Add uses permission to the manifest.xml file

Open the AndroidManifest.xml file, and add the following permissions:

- Call_privilage
- Call_phone
- Internet
- Camera
- Read_contact
- Internet
- WRITE_EXTERNAL_STORAGE

For this, before the application tag, you need to add

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

3.4 Implement the onclick function for all buttons

Implementation of the first button is given below, you need to implement the remaining EditText and buttons.

```
public void callIntent(View view) {
            Intent intent = null;
            switch (view.getId()) {
            case R.id.button1:
                  EditText textBrow = (EditText) findViewById(R.id.edit_url);
                  String url = textBrow.getText().toString();
                  intent = new Intent(Intent.ACTION_VIEW,
                              Uri.parse("http:// "+ site));
                  startActivity(intent);
                  break;
      case R.id.button2:
            //…
}
```

You need the following actions (see developer.android.com/reference/android/content/Intent.html):

- ACTION_VIEW, ACTION_CALL, ACTION_DIAL, ACTION_EDIT

For the URI, see http://developer.android.com/training/basics/intents/sending.html
For the camera, you need "android.media.action.IMAGE_CAPTURE" intent, and then use startActivityForResult() instead of startActivity(), which allow us to specify a desired result code. Once the intent is finished the method onActivityResult() is called and you can perform actions based on the result of the activity. You also need to define common var "Uri imageURI = null;". You may need to have a AVD with camera support, and increase the size of sd card to 128 MiB.

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
  if (resultCode == Activity.RESULT_OK && requestCode == 0) {
      try {
Bitmap bitmap = MediaStore.Images.Media.getBitmap(
getApplicationContext().getContentResolver(),  imageURI);
      if(bitmap == null){
           System.out.println("FAIL");
       }else{
            System.out.println("SAVED!!!");
            Toast.makeText(this, "Picture stored in " + imageURI,
      Toast.LENGTH_LONG).show();
      }
      …
  }
}
```

3.5 Exercises

1. In HelloWorld example, could we use different OnClick function for each intent?
2. Through an example, illustrate the relationship between different application components.
3. Show an app chooser, see http://developer.android.com/training/basics/intents/sending.html#AppChooser
4. Add an event to the calendar, see http://www.vogella.com/articles/AndroidCalendar/article.html
5. Add intents to the mycontactlist application, to make a call, send an email to your contact list.

4 Menu, preference, and Intent (duration < 90minutes)

This lab session will demonstrate how to create and evaluate a menu, how to define preferences, to navigate between activities via an intent, and how to create toast, dialog, and notification message.

Android supports the usage of *Preferences* for persisting key-values pairs in the Android file system. The definition of Preferences can be done via an XML resource.

The `PreferenceManager` provides methods to get access to preferences stored in a certain file. The following code shows how to access preferences from a certain file

```
SharedPreferences settings = getSharedPreferences("Test",
Context.MODE_PRIVATE);
```

The `PreferenceManager` gives access to the preference values. The following code shows how to access your default preferences.

```
SharedPreferences preferences =
PreferenceManager.getDefaultSharedPreferences(this);
```

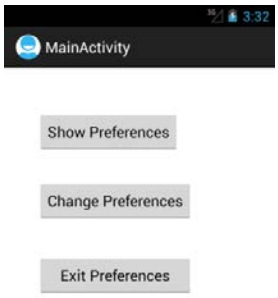
You may use `get Activity()` instead of `this`.

Values can get accessed via the key of the preference setting.

```
String username = preferences.getString("username", "n/a");
```

To create or change preferences you have to call the `edit()` method on the `SharedPreferences` object. Once you have changed the value you have to call the `commit()` method to apply your changes.

```
Editor edit = preferences.edit();
edit.putString("username", "new_value_for_user");
edit.commit();
```



4.1 Create a project

Create a project "fr.eurecom.android.preferences" with the activity "HelloPreferences".

Set the minimum API level to 16.

4.2 Add UI Elements

Change the UI in the file "/res/layout/activity_main.xml" in the following way.

- Delete the "Hello World, Hello!" via a right mouse click.
- From the "widgets" bar, drag three buttons
  - button1: change the “text” property to “Show Preferences”.
  - button2: change the “text” property to “Change ¨Preferences”
  - button3: change the “text” property to “Exit Preferences”

4.3 Add a Menu

4.3.1 Create preference file

Menus can be defined via XML files. Select your project, right click on “res” directory, and select “new > Android resource directory”.

1. Directory name: xml
2. Resource type: xml
3. Source set: main

Now create an Android XML resource called preferences.xml in the xml folder. Right click on the xml direcorty, and select “new > xml resource file ”

1. File name: preferences.xml
2. Add the following to the preferences.xml

```
<PreferenceCategory android:title="User Setting">
        <EditTextPreference android:summary="set the UserName"
android:title="User Name" android:key="username"/>
        <EditTextPreference android:summary="set the password"
android:title="Password" android:key="password"/>
    </PreferenceCategory>
```

Change your class "MainActivity" to the following. The OnCreateOptionsMenu method is used to create the menu. Please note that at the moment nothing happens if you select this menu. The behavior will be later implemented in the method "onOptionsItemSelected".

4.4 Create a Class “Preference”

Create the class "Preferences" with the superclass “`PreferenceActivity`”, which will load the "preference.xml". Add the following code:

```
package fr.eurecom.android.preferences;

import android.os.Bundle;
import android.preference.PreferenceActivity;

public class Preferences extends PreferenceActivity {
      /** Called when the activity is first created. */
      public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
//load the preferences from an xml rsource
          addPreferencesFromResource(R.xml.preferences);
      }
```

4.5 Update the manifestfile

Then Select "AndroidManifest.xml.

- Add the new activity "Preferences", and set the “Name” and “Label” attributes.

4.6 Implement the buttons

Change the coding of HelloPreferences to the following:

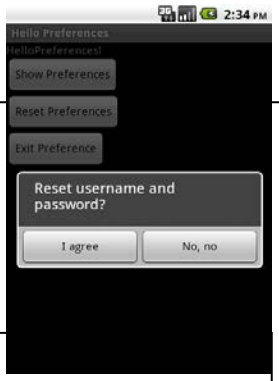
- Add the toast message to show user credentials without catching user focus

```
SharedPreferences preferences;
/** Called when the activity is first created. */
public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            …
            // Initialize preferences
            preferences = PreferenceManager.getDefaultSharedPreferences(this);

            Button button1 = (Button) findViewById(R.id.button1);
            button1.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
                        String username = preferences.getString("username", "n/a");
                        String password = preferences.getString("password", "n/a");
                        // A toast is a view containing a quick little message for the user.
                              showPrefs(username, password);
                  }
            });
      }
     private void showPrefs(String username, String password){
            Toast.makeText(MainActivity.this,
                        "You kept user: " + username + " and password: " + password,
                        Toast.LENGTH_LONG).show();
   }
```

- Store user credentials in the xml resource created above.

```
// This method is called once the menu is selected
      @Override
public boolean onOptionsItemSelected(MenuItem item) {
      switch (item.getItemId()) {
            // We have only one menu option
            case R.id.action_settings:
                  // Launch Preference activity
                  Intent i = new Intent(MainActivity.this, Preferences.class);
                  startActivity(i);
                  // A toast is a view containing a quick little message for the user.
                  Toast.makeText(HelloPreferences.this,
                              "Here you can store your user credentials.",
                              Toast.LENGTH_LONG).show();
                  Log.i("Main", "sent an intent to the Preference class")
                  break;
      }
      return true;
}
```



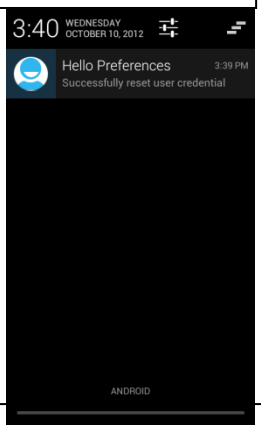
- Add the property "onClick" to Button02 "change Preferences", and set it to "openDialog". Note that this method is different from "Button button01 = (Button) findViewById(R.id.Button01);"
    - Note that a dialog is used to get the focus of the user.
- Add the following function to your code

```
public void openDialog(View v) {
            // Create out AlterDialog
            AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setMessage("Reset username and password?");
            builder.setCancelable(true);
            builder.setPositiveButton("I agree", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                  Toast.makeText(getApplicationContext(),"Resetting

      Credential",Toast.LENGTH_LONG).show();
                }
            });
            builder.setNegativeButton("No, no", new DialogInterface.OnClickListener() {
                  public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(getApplicationContext(),"Keep
                                          Credential",Toast.LENGTH_LONG).show();
                }
            });
            AlertDialog dialog = builder.create();
            dialog.show();
}
```

- Implement Reset Preference function

```
public void reset_preferences(){
            Editor edit = preferences.edit();
            edit.putString("username", null);
            edit.putString("password", null);
            edit.commit(); // Apply changes
            // A toast is a view containing a quick little message for the
            // user. We give a little feedback
            Toast.makeText(MainActivity.this,
                        "Reset user name and password",
                        Toast.LENGTH_LONG).show();
}
```



- Create a notification
  - Notification manager is the library used to add a notification into the title bar. The user can open the notification bar by sliding the title bar down. Furthermore, the associated activity can be launched through the notification bar.
  - "getSystemService()" is used to get the notification maneger. The "Hello Preference" will send a "PendingIntent" to the "notification manger" to execute the notification with the permission of "Hello preference".
- Add the following code

```
public void createNotification() {
      NotificationManager notificationManager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
            // prepare intent which is triggered if the notification is selected
            Intent intent = new Intent(this, HelloPreferences.class);
            PendingIntent activity = PendingIntent.getActivity(this, 0, intent, 0);
            Notification notification = new Notification.Builder(this)
                  .setContentTitle("Hello Preferences")
                  .setContentText("Successfully reset user Credential")
                  .setSmallIcon(R.drawable.icon)
                  .setContentIntent(activity)
                  .setAutoCancel(true)
                  .addAction(R.drawable.icon, "Call", activity)
                  .addAction(R.drawable.icon, "More", activity)
                  .build();
            notificationManager.notify(0,notification);

}
```

4.7 Run and test the application

- To test the app, follow this steps:
  - Menu >Settings and set the username and password
  - Back to the app, and click on "Show Preferences", you should see the value you just entered

- Click on “Reset Preferences”, then “No, no”, and then “ShowPreferences”, you should see the value you just entered
- Click on “Reset Preferences”, then “I agree”, and then “ShowPreferences”, you should see “n/a”, check the notification, click in the “Hello Preferences” notification
- Click on exit, will exit your app.
- Go to application, and check out the icon of your app.

4.8 Questions

- How the Preferences class is called ?
- You may be noticed those user credential is not actually reset and that the notification is not received. How this can be done in the code?
- Implement exit function for the button3 to exit the application.
- There are a some decrypted functions. Some of them are due to the notion of Fragment.
  - See http://developer.android.com/guide/components/fragments.html
  - Explain how to redesign the app to work with fragment
- Update the app icon, use any icon of your choice
- Give example of applications where such preferences are useful?

5 Reference

[1] http://developer.android.com/guide/topics/fundamentals.html
[2] http://www.vogella.de/android.html
[3] http://developer.android.com/resources/index.html
[4] http://code.google.com/p/android-for-gods/w/list