LIVEhouse.in

Android

Zhuohao Lee(李卓皓)

Manager of Embedded Team, iKala

LOVEhouse.in

About Me

- iKala, Embedded Team Manager (2012 - Present)
- MediaTek, Software Engineer (2007 - 2011)
- Phison, Software Engineer (2006 – 2007)
- NCTU Computer Science, Master (2005 – 2006)
- NCTU Computer Science (2001 - 2005)

LOVEhouse.in

- Today`s Topic is focus on Developing an Android Application
- I will demo an example which get and parse data from server
- Assume no experience on developing Android code
- Pre-required: JAVA
- Two breaks: 15.10~15.30, 16.20~16.30

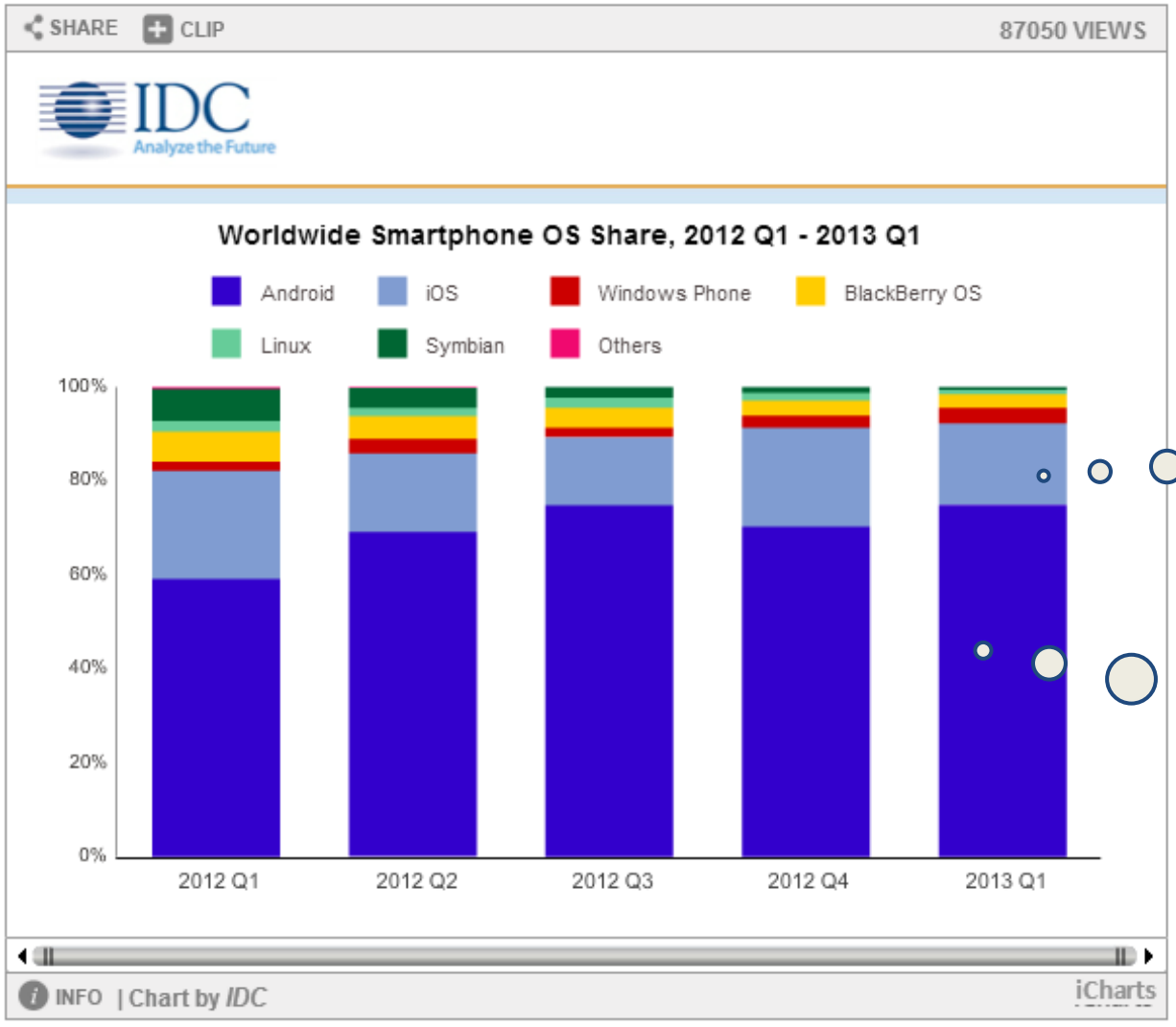
LOVEhouse.in

Overview

- Why Android?
- What is Android?
- Environment setup
- How to write an Android App?
- A very easy example
- AVD hot keys
- Android Components
- Sample Code & Demo
- Experience Sharing

LOVEhouse.in

Why Android?



iOS~15%

Android ~70%
Many Manufacturers
Multi-Device issues
Challenge!

What is Android?

- Based on the Linux kernel
- Designed primarily for touchscreen mobile devices such as smartphones and tablet computers
- Applications are developed in the Java language using the Android software development kit (SDK)

LOVEhouse.in

Environment setup

1. Install JDK 7
2. Develop Environment setup, eclipse + Android SDK
3. Install Android 4.4 SDK and Android 2.3.3 SDK
4. Install AVDs (Android 4.4 and 2.3.3)
5. (Option) Install android mobile phone's USB driver for App running

LOVEhouse.in

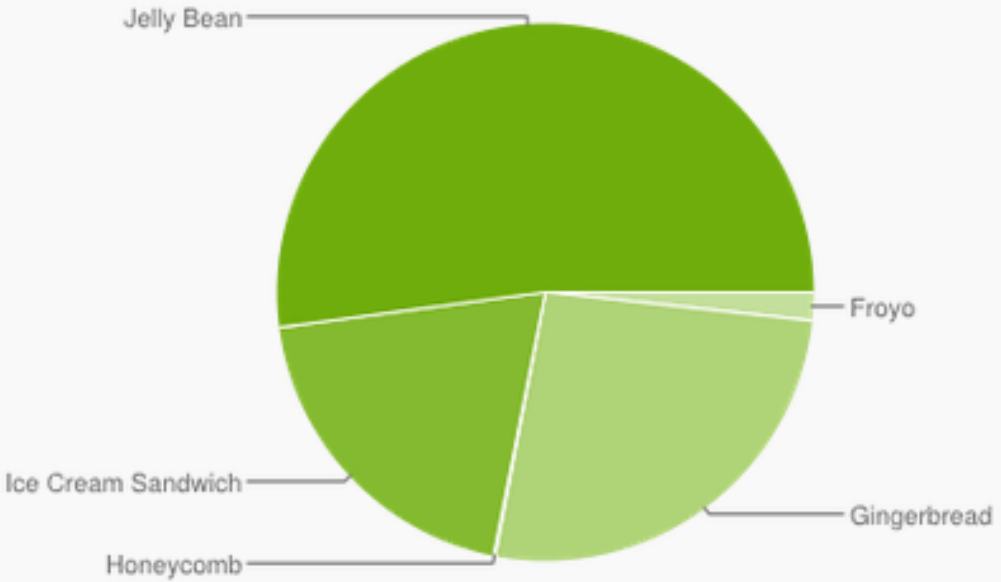
Environment setup

- To compatible to others platform, please setting your eclipse
  - Window->Preferences->General->Editors->Text Editors, check “Insert space for tab”
  - Window->Preferences->Java->Code Style->Formatter->new profile, select Tab Policy to “Spaces Only”
  - Window->Preferences->Workspace->Text file encoding, check Other and select UTF-8

LOVEhouse.in

Why API Level 10

Version	Codename	API	Distribution
2.2	Froyo	8	1.7%
2.3.3 - 2.3.7	Gingerbread	10	26.3%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	19.8%
4.1.x	Jelly Bean	16	37.3%
4.2.x		17	12.5%
4.3		18	2.3%



Data collected during a 7-day period ending on November 1, 2013.
Any versions with less than 0.1% distribution are not shown.

LOVEhouse.in

How to write an Android App?

1. Open Eclipse
2. File->New->Android Application Project
3. Fill in Application Name, Project Name, Package Name
4. Next->Next->…->Finish

LOVEhouse.in

A very easy example

1. Project -> clean
2. Run-> Run Configurations -> Always prompt to pick device
3. Run-> Run -> Launch a new Android Virtual Device
4. Select AVD 2.3.3
5. Unlock the screen you will see your App

LOVEhouse.in

AVD hot keys

Table 1. Emulator keyboard mapping

Emulated Device Key	Keyboard Key
Home	HOME
Menu (left softkey)	F2 *or* Page-up button
Star (right softkey)	Shift-F2 *or* Page Down
Back	ESC
Call/dial button	F3
Hangup/end call button	F4
Search	F5
Power button	F7
Audio volume up button	KEYPAD_PLUS, Ctrl-F5
Audio volume down button	KEYPAD_MINUS, Ctrl-F6
Camera button	Ctrl-KEYPAD_5, Ctrl-F3
Switch to previous layout orientation (for example, portrait, landscape)	KEYPAD_7, Ctrl-F11
Switch to next layout orientation (for example, portrait, landscape)	KEYPAD_9, Ctrl-F12
Toggle cell networking on/off	F8
Toggle code profiling	F9 (only with `-trace` startup option)
Toggle fullscreen mode	Alt-Enter
Toggle trackball mode	F6
Enter trackball mode temporarily (while key is pressed)	Delete
DPad left/up/right/down	KEYPAD_4/8/6/2
DPad center click	KEYPAD_5
Onion alpha increase/decrease	KEYPAD_MULTIPLY(*) / KEYPAD_DIVIDE(/)

Bug!!

Switch Second times will fail

Android Components

- **Activities**
  - An *activity* represents a single screen with a user interface.
  - An activity is implemented as a subclass of Activity
- **Services**
  - A *service* is a component that runs in the background to perform long-running operations or to perform work for remote processes.
  - A service is implemented as a subclass of Service
- **Content providers**
  - A *content provider* manages a shared set of application data
  - A content provider is implemented as a subclass of ContentProvider
- **Broadcast receivers**
  - A *broadcast receiver* is a component that responds to system-wide broadcast announcements.
  - A broadcast receiver is implemented as a subclass of BroadcastReceiver

LOVEhouse.in

Activity Life Cycle

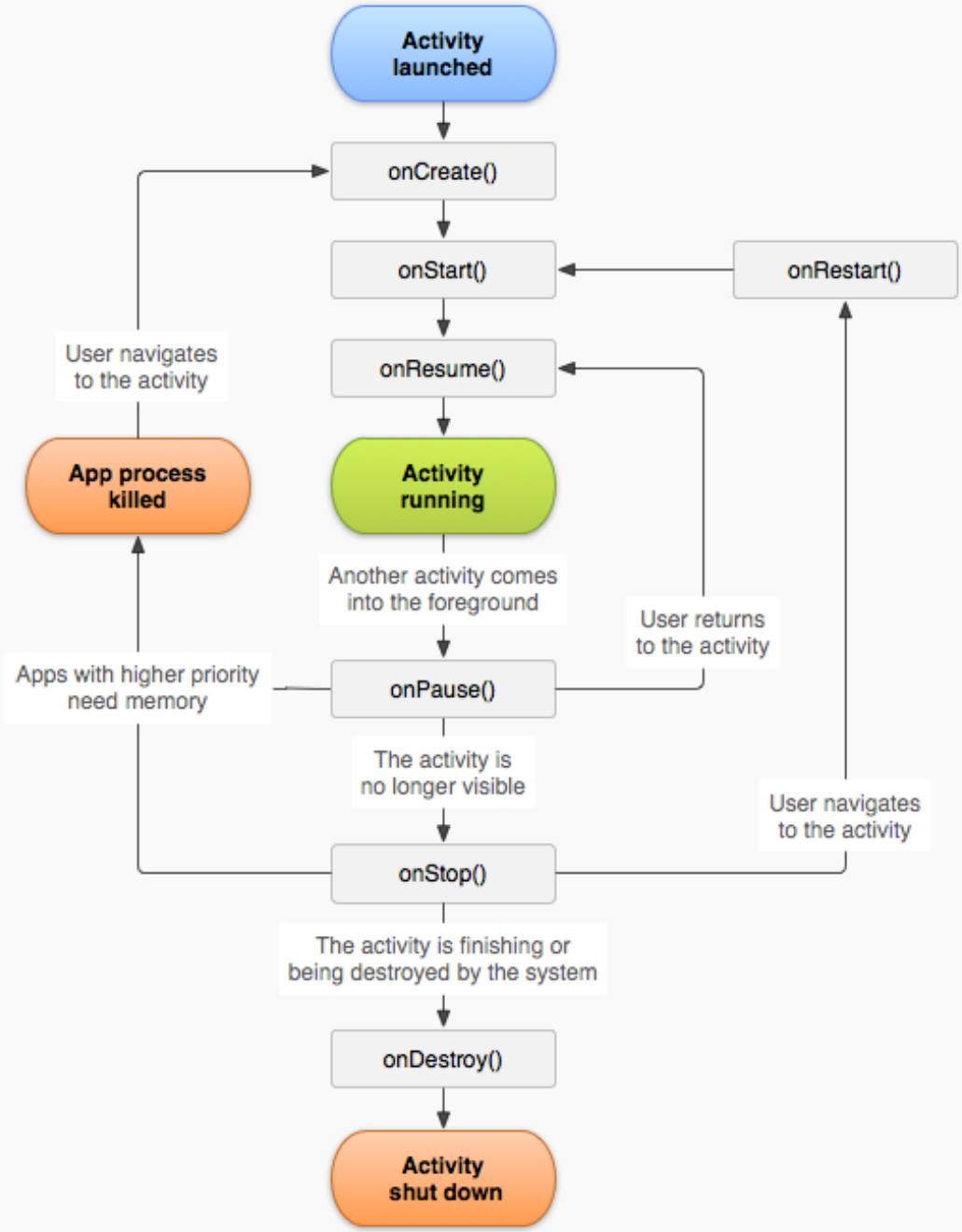


Figure 1. The activity lifecycle.

LOVEhouse.in

illustration 1

- Application: Example2
- Add 3 Text View and setText to statusCode, country, ipAddress respectively

LOVEhouse.in

Creating An Application

- Create an Activity name Example2
- Min. SDK: API 10
- Target SDK: API 10
- Compile With: API 19

LOVEhouse.in

Add Views & members

- res/layout/activity_main.xml
  - Add 3 TextView
    - id:statusCode , country, ipAddress
- src/com/example/example2/ExampleActivity.java
  - Add members: 3 TextView (name: statusCode, country, ipAddress)

Link view's ID & manipulate

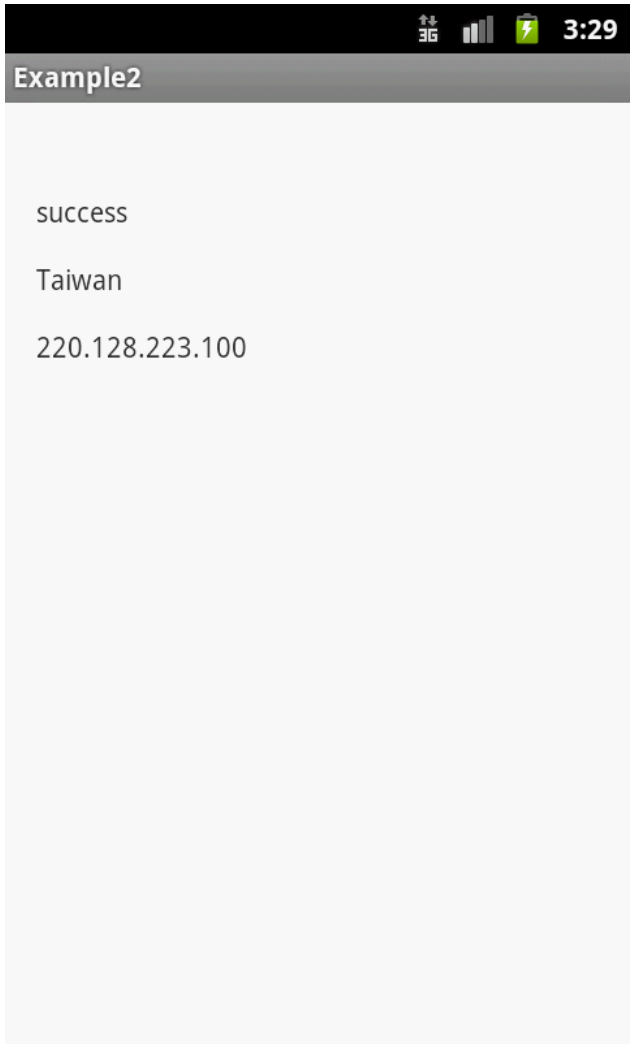
- src/com/example/example2/ExampleActivity.java
- Add code in function onCreate() {

```
setContentView(R.layout.activity_main);
statusCode = (TextView)findViewById(R.id.statusCode);
country = (TextView)findViewById(R.id.country);
ipAddress = (TextView)findViewById(R.id.ipAddress);
statusCode.setText("statusCode");
country.setText("country");
ipAddress.setText("ipAddress ");
}
```

- Run AVD with Target 2.3.3

LOVEhouse.in

illustration 2



- Access Network and get JSONData from http://ip-api.com/json
- Show status, country & query in textview

LOVEhouse.in

Connect Network & Parse JSON Data

- get JSON data from: http://ip-api.com/json
  - {"status":"success","country":"Taiwan",...,"query":"220.128.223.100"}
- Target: parse “status”, “country”, “query” and show in TextView respectively
- Use HttpClient, HttpResponse class to get data
- Use JSONObject to get the data
- Add two functions queryData() & parseAndFillData(), and call function in onCreate() function

LOVEhouse.in

Connect Network (Source Code)

```
// 省略try catch及一些boundary case
 private JSONObject queryData(String URL){
      // Network access
       HttpClient lClient = new DefaultHttpClient();
       HttpGet lGetMethod = new HttpGet(URL);
       HttpResponse lResp = null;
       lResp = lClient.execute(lGetMethod);

       // Read JSON data from http response
       ByteArrayOutputStream lBOS = new ByteArrayOutputStream();
       String lInfoStr = null;
       JSONObject jSONObjResponse = null;
       lResp.getEntity().writeTo(lBOS);
       lInfoStr = lBOS.toString("UTF-8");
       jSONObjResponse = new JSONObject(lInfoStr);
       return jSONObjResponse;
   }
```

LOVEhouse.in

Parse JSON Data (Source Code)

```
// 省略try catch及一些boundary case
private void parseAndFillData(JSONObject jObj){
        // parse JSONObject
        sts = jObj.getString(STATUS_KEY);
        ctry = jObj.getString(COUNTRY_KEY);
        ip = jObj.getString(IP_KEY);

        // show parse data on TextView
        statusCode.setText(sts);
        country.setText(ctry);
        ipAddress.setText(ip);
}
```

LOVEhouse.in

LogCat + Break Point Debug

- Run previous code on AVD 2.3.3
  - Error!!
  - not check in parseAndFillData
  - not permission on Internet
- Add Uses Permission in AndroidManifest.xml
  - android.permission.INTERNET
  - re-run again, you will see the correct data
  -

LOVEhouse.in

Blocking API

- If running on AVD 4.4,
  - Error!!! Why?
  - Network Access is Blocking API
  - Two Solutions:
    - Bypass Blocking API Check
    - Run in worker thread
-

LOVEhouse.in

Bypass Blocking API Check

- Add code in onCreate() before super.onCreate()

```
StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()
        .detectDiskReads()
        .detectDiskWrites()
        .detectNetwork()
        .penaltyLog()
        .build());
```

- Not Recommended!!

LOVEhouse.in

Run in Worker Thread

- Move queryData & parseAndFillData inside thread's run function

```
new Thread(new Runnable(){
        @Override
         public void run() {
              JSONObject jObj = queryData("http://ip-api.com/json");
              parseAndFillData(jObj);
         }
}).start();
```

- In Function parseAndFillData(), add runOnUiThread()

```
runOnUiThread(new Runnable(){
         @Override
          public void run() {
                  // origin code in parseAndFillData();
         }
});
```

LOVEhouse.in

UI Thread

- If missing second step in previous slide, the app
  - Crash!
  - Logcat output CalledFromWrongThreadException
  - This exception can’t be bypass
- Android SDK provide AsyncTask for easy use for UI thread

AsyncTask

- An asynchronous task is defined by a computation that runs on a background thread and whose result is published on the UI thread.
- Use AsyncTask when
  - Data which calculated by Thread is shown on UI
  - Progress UI is shown when Data is processing by Thread

LOVEhouse.in

How AsyncTask work?

- When an asynchronous task is executed, the task goes through 4 steps:
  a. onPreExecute() : run on UI Thread
  b. doInBackground(Params...): run on worker Thread
  c. onProgressUpdate(Progress...): run on UI Thread
  d. onPostExecute(Result): run on UI Thread
-

LOVEhouse.in

Copy MainActivity and naming MainActivity2

Modify MainActivity2

```
protected void onCreate(Bundle savedInstanceState) {
        ...
        ipAddressAsyncTask = new AsyncTask<String, Integer, JSONObject>() {
                @Override
                protected void onPreExecute() {
                    super.onPreExecute();
                }
                @Override
                protected JSONObject doInBackground(String... params) {
                    JSONObject result;
                    result =queryData(params[0]);
                    return result;
                }
                @Override
                protected void onProgressUpdate(Integer... values) {
                    super.onProgressUpdate(values);
                }
                @Override
                protected void onPostExecute(JSONObject result) {
                    super.onPostExecute(result);
                    parseAndFillData(result);
                }
    };
    ipAddressAsyncTask.execute("http://ip-api.com/json");

}
```

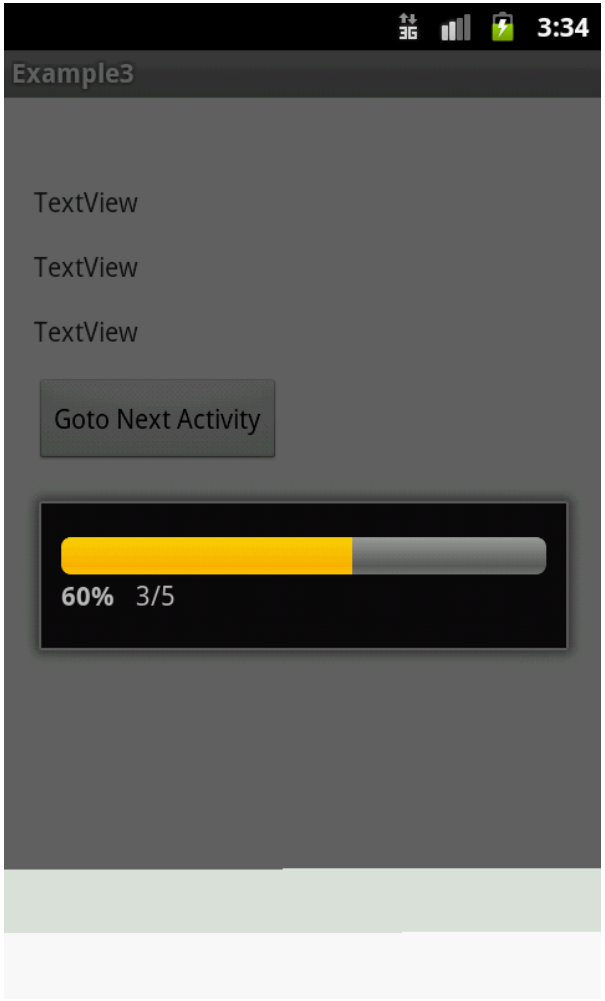
LOVEhouse.in

Add new Activity in Manifest & Set to default Activity

- In AndroidManifest.xml
  - Application->Add->Activity->Select Activity Name (MainActivity2.java)
  - Move <intent-filter> ... </intent-filter> into MainActivity2 tag
- Save & Run on AVD

LOVEhouse.in

illustration 3



- Show Loading bar when data not ready

LOVEhouse.in

onPreExecute() & onProgressUpdate()

```
onCreate(){
      …
   progressDialog = new ProgressDialog(this);
    progressDialog.setIndeterminate(false);
    progressDialog.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
    progressDialog.setMax(5);
    ...
}
@Override
 protected void onPreExecute() {
      super.onPreExecute();
      progressDialog.show();
}
@Override
      protected void onProgressUpdate(Integer... values) {
      super.onProgressUpdate(values);
      progressDialog.setProgress(values[0].intValue());
}
```

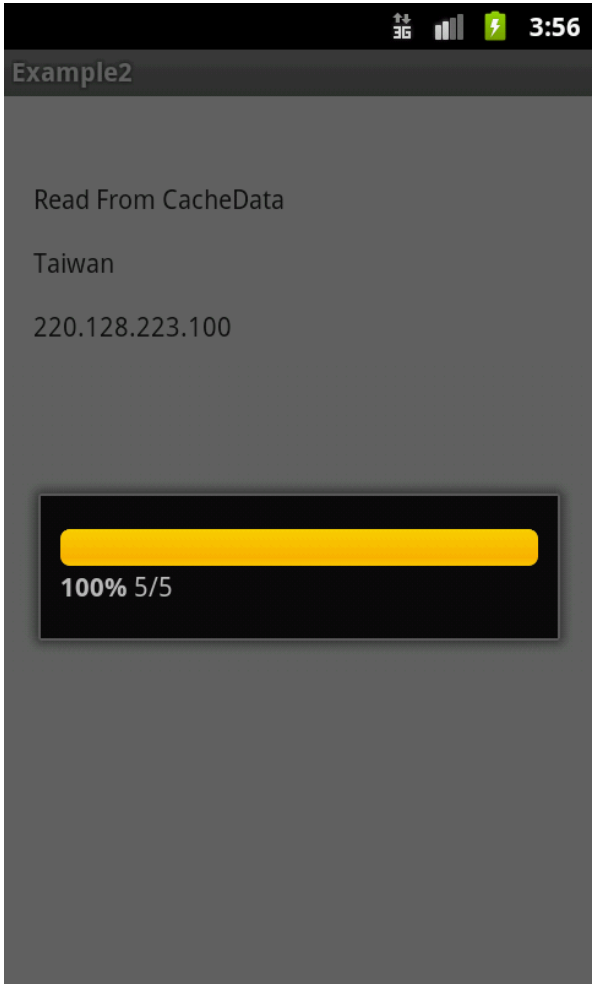
LOVEhouse.in

onPostExecute() & doInBackground()

```
@Override
    protected void onPostExecute(JSONObject result) {
    super.onPostExecute(result);
    progressDialog.dismiss();
    parseAndFillData(result);
}
@Override
 protected JSONObject doInBackground(String... params) {
    JSONObject result;
    result =queryData(params[0]);
    publishProgress(0);
    for(int i=1;i<=5;i++){
        try{
          Thread.sleep(1000);
         }catch (InterruptedException e) { }
        publishProgress(i);
    }
    return result;
 }
```

LOVEhouse.in

illustration 4



- Use SharedPreferences to cache Data
- Show status to “Read From CacheData” if read data from cache

LOVEhouse.in

Can i cache the data for later used?

- Definitely Yes!
- Method
  - Shared Preferences
  - Write File
  - SQLite Databases
  - ...
- Today introduce SharedPreferences
  - function used: getSharedPreferences(), contains(), getString(), edit(), Editor. putString(), Editor.apply()

LOVEhouse.in

Check the preference data

```
protected void onCreate(Bundle savedInstanceState) {
        …
        // get the preferencese
        cacheData = getSharedPreferences(PREFERENCE_NAME, Context.MODE_PRIVATE);
        // if data exists, we can use the cache data
        if(cacheData.contains(CACHE_DATA_KEY)){
                String jSONString = cacheData.getString(CACHE_DATA_KEY, null);
                JSONObject jObj = null;
                try {
                        jObj = new JSONObject(jSONString);
                } catch (JSONException e) {
                        e.printStackTrace();
                }

                if(jObj!=null){
                        parseAndFillData("Read From "+CACHE_DATA_KEY, jObj);
                }
        }
}
```

LOVEhouse.in

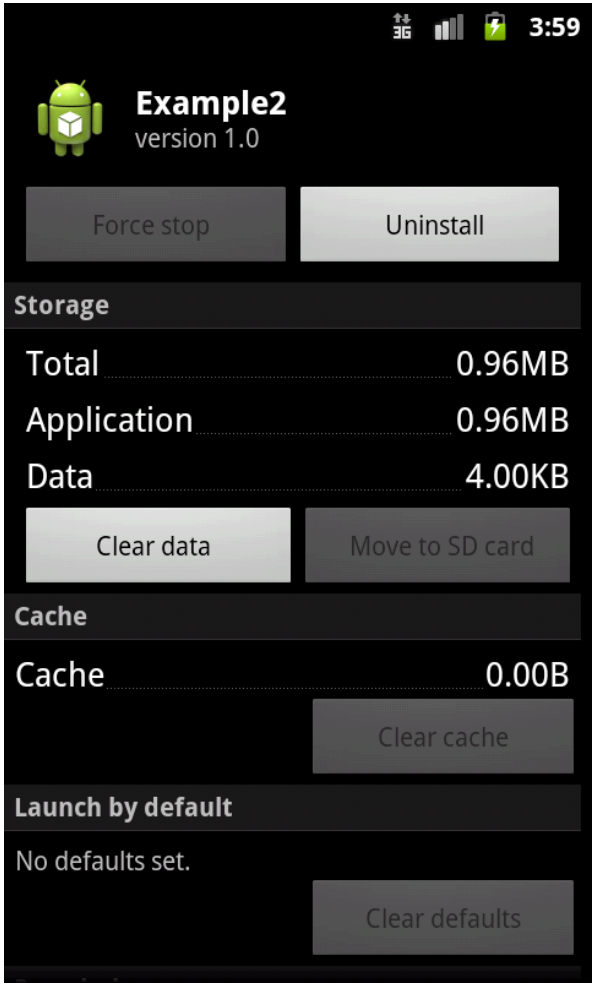
Save the data to preference data

```
private void parseAndFillData(JSONObject jObj){
      if(cacheData!=null && jObj!=null){
         Editor editor = cacheData.edit();
         if(null != editor){
            editor.putString(CACHE_DATA_KEY, jObj.toString());
            editor.apply();
         }
      }
      parseAndFillData(null, jObj);
 }
private void parseAndFillData(String loadingStatus, JSONObject jObj){
      …
      if(loadingStatus!=null)
            sts=loadingStatus;
      else
            sts = jObj.getString(STATUS_KEY);
     …
}
```

Move original code to parseAndFillData(null, jObj)

LOVEhouse.in

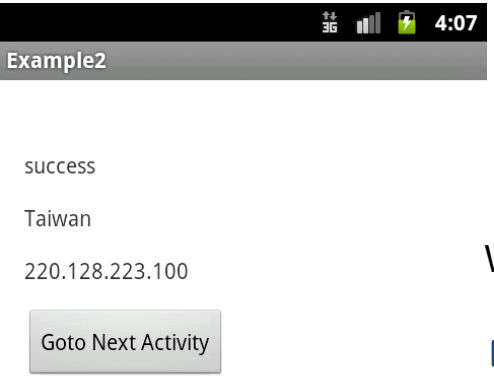
Clear Shared Preferences Data



- Settings→Applications→ Manage applications→ APP NAME→Clear data

LOVEhouse.in

illustration 5



When Button Click

- Add a button
- Switch to the another activity when button click and show Button Text to “No Function”

Start Another Activity

- Add a button below ipAddress with text=”Goto Next Activity” in activity_main.xml
- Use bundle.putString(), intent.setClass(), startActivity() to start and pass parameter the next activity
-

LOVEhouse.in

Start Another Activity (Sample Code)

```
protected void onCreate(Bundle savedInstanceState) {
...
 gotoNextActivity = (Button)findViewById(R.id.gotoNextActivity );
        gotoNextActivity .setOnClickListener(new Button.OnClickListener(){
           @Override
            public void onClick(View v) {
               Intent intent = new Intent();
               intent.setClass(MainActivity2.this, MainActivity.class);
                Bundle bundle = new Bundle();
                bundle.putString(MainActivity.CHANGE_BUTTON_TEXT_KEY,"No Function");
                intent.putExtras(bundle);
               startActivity(intent);
            }
        });
}
// in next Activity
protected void onCreate(Bundle savedInstanceState) {
          …
           Bundle bundle = getIntent().getExtras();
           String showButtonText = bundle.getString(MainActivity2.CHANGE_BUTTON_TEXT_KEY);
           gotoNextActivity.setText(showButtonText);
}
```

LOVEhouse.in

Service

- A Service is an application component that can perform long-running operations in the background and does not provide a user interface.
- When use Service? Do something user no need aware it
  - Recieve Message
  - Download/Upload Data
  - ...

LOVEhouse.in

Service Lifecycle

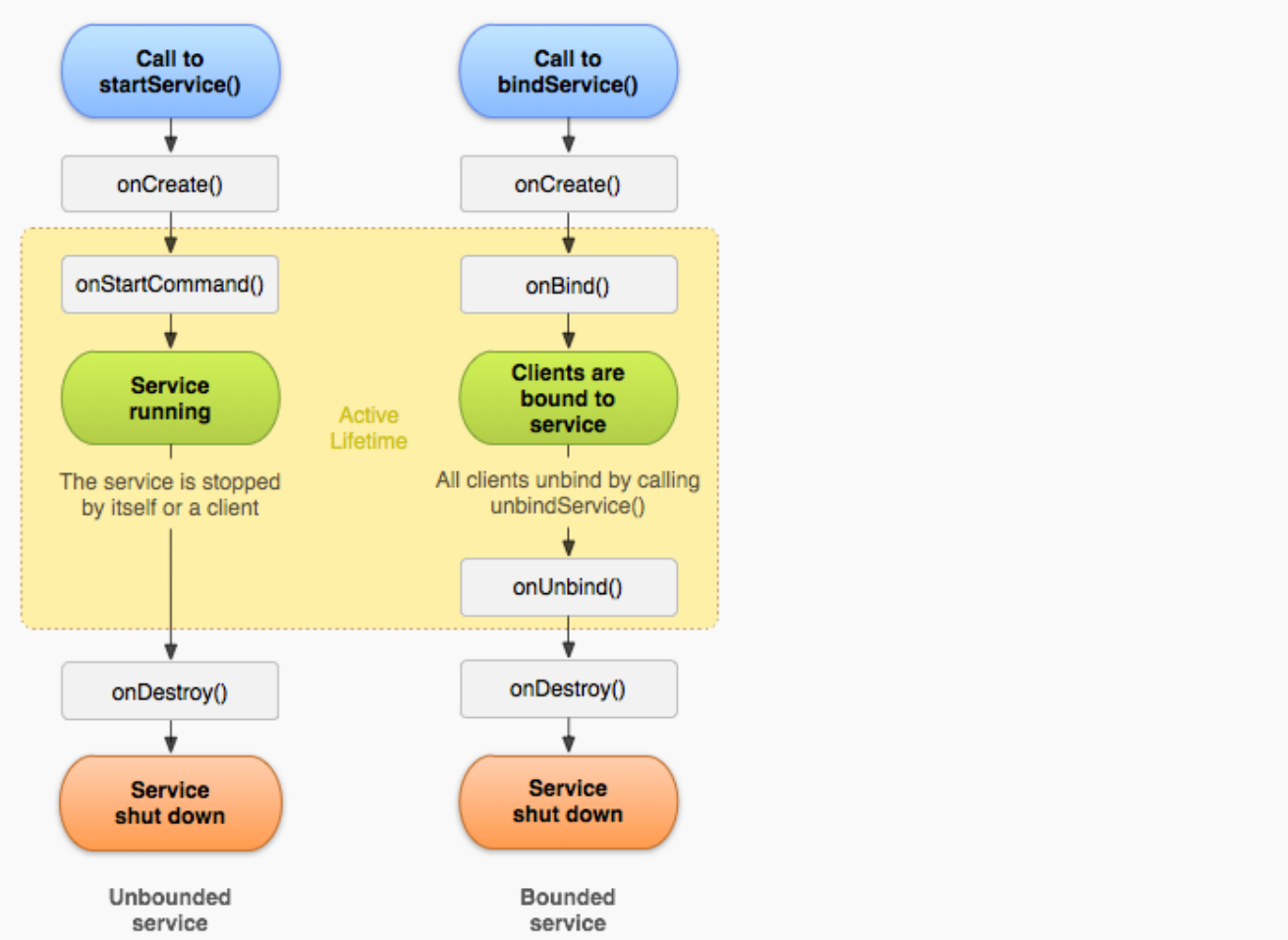
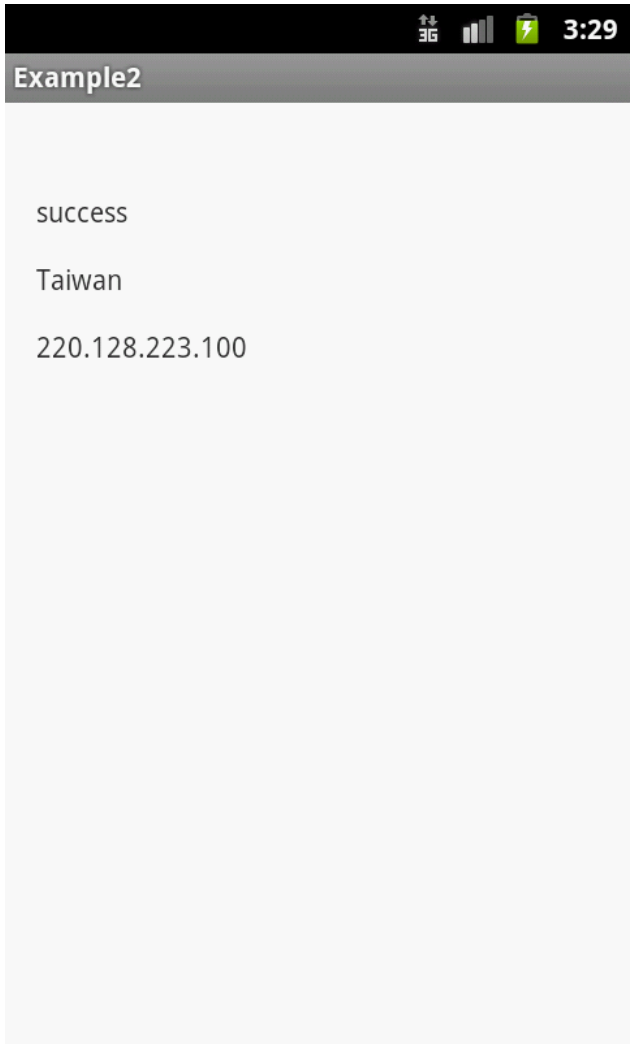


Figure 2. The service lifecycle. The diagram on the left shows the lifecycle when the service is created with `startService()` and the diagram on the right shows the lifecycle when the service is created with `bindService()`.

LOVEhouse.in

illustration 6



- Similar to illustration 2 but using Service + Broadcast receiver

LOVEhouse.in

IntentService & Broadcast message

- Add 3 classes which extended IntentService, BroadcastReceiver and Activity respectively
- In IntentService,

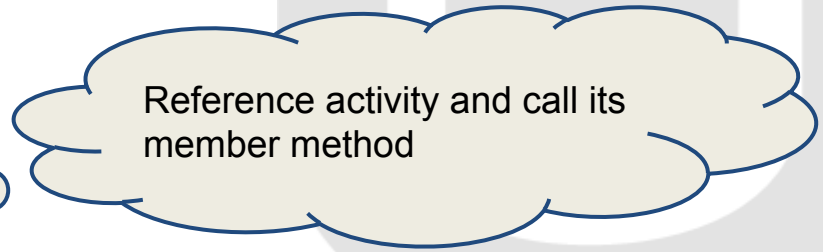
```
@Override
 protected void onHandleIntent(Intent intent) {
        // function called when someone call StartService(intent)
        String url = intent.getExtras().getString(URL_KEY);
        JSONObject jObj = queryData(url);
        publishResults(jObj.toString());
    }
private void publishResults(String result) {
        // broadcast message
        Intent intent = new Intent(NOTIFICATION);
        intent.putExtra(MainReciever.JSONSTRING_KEY, result);
        sendBroadcast(intent);
    }
```

LOVEhouse.in

BroadcastReciever

- In BroadcastReceiver

```
@Override
    public void onReceive(Context context, Intent intent) {
        // Show a message in UI
        Toast.makeText(context, "Broadcast Intent Detected.", Toast.LENGTH_LONG).show();
        if(activity!=null){
            String jString = intent.getExtras().getString(JSONSTRING_KEY);
            JSONObject jObj=null;
            try {
                jObj = new JSONObject(jString);
            } catch (JSONException e) {
                e.printStackTrace();
            }
            activity.parseAndFillData(jObj);
        }
    }
```



LOVEhouse.in

How to Use Service & Broadcast receiver

- In Activity

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
            …
            // Start Service with intent
            Intent intent = new Intent(this, MainService.class);
             intent.putExtra(MainService.URL_KEY, "http://ip-api.com/json");
            startService(intent);
            // Register Reciever
            mainReciever = new MainReciever(this);
            registerReceiver(mainReciever, new IntentFilter(MainService.NOTIFICATION));
}
@Override
    protected void onDestroy() {
        // Unregister Reciever
        unregisterReceiver(mainReciever);
        super.onDestroy();
    }
```

LOVEhouse.in

Add Intent-Filter to activate corresponding function

- Add activity, service and reciever in Manifest
- Because reciever need process Intent, so we need to add intent filter for reciever

```
<intent-filter>
        <action android:name="com.example.example2.MainReceiver"/>
</intent-filter>
```

- Set Activity to Startup Activity

```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- Run App

LOVEhouse.in

IDE Provide Some Useful Functions

- Right click select source → generate xxx to generate source code
- Import related library (Ctrl+Shift+O)
- F3: jump to the function or variable definition
- Use Breakpoints/Logcat to debug code
- In debug mode
  - F5: step into
  - F6: step over
  - F8: resume

LOVEhouse.in

How to deploy a good App?

- Treat seriously about the try-catch block
- Use lint tool to scan potential error
- Do not ignore any compiler warning!
- Must run App on a real platform
- Test multi-device, ex: screen size, cpu speed, RAM size,...
- Do not skip any logcat's warning (including system log)
- Do not fix bug on a specify phone if it is a multi-device issue
- Search multi-device issues on Google and find solution!

LOVEhouse.in

iKala Android App

LOVEhouse.in

- 想學習更多手機、後端技術嗎?
- 想與一流的團隊一起工作嗎?
- 想展現你的實力嗎?

實習生
招募中

iKala 徵才中!

履歷請寄朱佩霜小姐 shelly@ikala.tv

- 系　　所:大專或專科資訊相關科系 (強者不受此限制)
- 工作內容:iOS app、Android app、後端系統開發
- 熟悉的程式語言:C/C++, Java, Objective-C
- 工作時間:非常彈性, 一個禮拜一天或是每天來都可以
- 時　　薪:170 元以上
- http://event.ikala.tv/2013/11/intern/index.html