—— Button, Intent, Bundle ——

```
/*In MainActivity*/
btnreset.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MainActivity.this,
             SecondActivity.class);
        Bundle bundle = new Bundle();
        bundle.putString("name", name);
        intent.putExtras(bundle);
        startActivity(intent);
        /*int requestCode = 2;
        startActivityForResult(intent, requestCode);*/
    }
});
@Override
protected void onActivityResult(int requestCode, int
     resultCode, Intent data) {
    String newNo = data.getStringExtra("_newNo");
    switch (resultCode) {
        case 1:
    }
}
/*In SecondActivity*/
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second);
    Bundle bundle = getIntent().getExtras();
    String name = bundle.getString("name");
}
Intent intent = new Intent();
intent.putExtra("_newNo", newNo);
int resultCode = 1;
SecondActivity.this.setResult(resultCode, intent);
SecondActivity.this.finish();
```

—— ListView, List, ArrayAdapter ——

```
private ListView fruitListView;
private List<Fruit> fruitList = new ArrayList<Fruit>();
fruitListView = (ListView)this.findViewById(R.id.
     fruitListView);
fruitList.add(new Fruit("apple", R.mipmap.apple));
FruitAdapter adapter = new FruitAdapter(FruitList.this,
     R.layout.fruit_item, fruitList);
fruitListView.setAdapter(adapter);
fruitListView.setOnItemClickListener(new AdapterView.
     OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View
         view, int position, long id) {
        ListView listView = (ListView) parent;
        Fruit fruit = (Fruit) listView.getItemAtPosition
             (position);
    }
});
fruitListView.setOnItemLongClickListener(new AdapterView
     .OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> parent
         , View view, int position, long id) {
        try {
            ListView listView = (ListView) parent;
            Fruit fruit = (Fruit) listView.
                 getItemAtPosition(position);
            FruitAdapter adapter = (FruitAdapter)
                 listView.getAdapter();
            adapter.remove(fruit);
            adapter.notifyDataSetChanged();
            listView.invalidate();
        } catch (Exception e) {e.printStackTrace();
             return false;}
        return true;
    }
});
public class FruitAdapter extends ArrayAdapter<Fruit> {
    private int resourceId;
    public FruitAdapter(Context context, int
         textViewResourceId, List<Fruit> fruits){
        super(context, textViewResourceId, fruits);
        this.resourceId = textViewResourceId;
    }
    @Override
    public View getView(int position, View convertView,
         ViewGroup parent){
        Fruit fruit = getItem(position);
        View view = LayoutInflater.from(this.getContext
             ()).inflate(resourceId, null);
        ImageView fruitImage = (ImageView)view.
             findViewById(R.id.fruit_image);
        TextView fruitName = (TextView)view.findViewById
             (R.id.fruit_name);
        fruitImage.setImageResource(fruit.getImageId());
        fruitName.setText(fruit.getName());
        return view;
    }
}
```

—— AppWidgetProvider, BroadcastReceiver ——

```
public class MyWidgetProvider extends AppWidgetProvider
     {
    @Override
    public void onUpdate(Context context,
         AppWidgetManager appWidgetManager, int[]
         appWidgetIds){
        super.onUpdate(context, appWidgetManager,
             appWidgetIds);
        Intent intent = new Intent(context, MainActivity
             .class);
        PendingIntent pendingIntent = PendingIntent.
             getActivity(context, 0, intent, 0);
        RemoteViews remoteViews = new RemoteViews(
             context.getPackageName(), R.layout.
             widget_layout);
        remoteViews.setOnClickPendingIntent(R.id.
             widget_text, pendingIntent);
        appWidgetManager.updateAppWidget(appWidgetIds,
             remoteViews);
    }
}
public class MyBroadcastReceiver extends
     BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent
         ){
        RemoteViews remoteViews = new RemoteViews(
             context.getPackageName(), R.layout.
             widget_layout);
        remoteViews.setTextViewText(R.id.widget_text,
             intent.getStringExtra("message"));
        AppWidgetManager.getInstance(context).
             updateAppWidget(new ComponentName(context.
             getApplicationContext(),MyWidgetProvider.
             class), remoteViews);
    }
}
private boolean isRegister;
private MyBroadcastReceiver myBroadcastReceiver;
protected void onCreate(Bundle savedInstanceState) {
    myBroadcastReceiver = new MyBroadcastReceiver();
    unregisterReceiver(myBroadcastReceiver);
    registerReceiver(myBroadcastReceiver, new
         IntentFilter(getResources().getString(R.string.
         sysu)));
    /* To Send a broadcast*/
    Intent intent = new Intent(getResources().getString(
         R.string.sysu));
    intent.putExtra("message", mainEditText.getText().
         toString());
    sendBroadcast(intent);
}
```

—— Service ——

```
public class myService extends Service {
    public final IBinder binder = new MyBinder();
    public class MyBinder extends Binder{
        myService getService() {
            return myService.this;
        }
    }
    @Override
    public IBinder onBind(Intent intent) {
        return binder;
    }
}
private MusicService musicService = new musicService();
private ServiceConnection sc = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName
         componentName, IBinder iBinder) {
        musicService = ((MusicService.MyBinder)iBinder).
             getService();
    }
    @Override
    public void onServiceDisconnected(ComponentName
         componentName) {
        musicService = null;
    }
};
private void bindServiceConnection() {
    Intent intent = new Intent(MainActivity.this,
         MusicService.class);
    startService(intent);
    bindService(intent, sc, this.BIND_AUTO_CREATE);
}
```

—— SharedPreferences, CheckBox, AutoCompleteTextView ——

```
SharedPreferences sp = getSharedPreferences("xmlUserData
     ", MODE_PRIVATE);
SharedPreferences.Editor editor = sp.edit();
editor.putString("username", usr);
editor.commit();
chboxRem.setOnCheckedChangeListener(new CompoundButton.
     OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(CompoundButton
         compoundButton, boolean isChecked) {
        editor.putBoolean("isChecked", isChecked);
        editor.commit();
        if(isChecked && edtUsr.getText().toString().
             equals(sp.getString("username", null))) {
            edtPwd.setText(sp.getString("password", null
                 ));
        }
    }
});
chboxRem.setChecked(true);
AutoCompleteTextView fileName = (AutoCompleteTextView)
     this.findViewById(R.id.textViewFileName);
fileName.setAdapter(new ArrayAdapter<String>(this,
     android.R.layout.simple_dropdown_item_1line, this.
     fileList()));
```

—— SQLite ——

```
public class MyDatabaseHelper extends SQLiteOpenHelper {
    private static final String DB_NAME = "Contacts.db";
    private static final String TABLE_NAME = "Contacts";
    private static final int DB_VERSION = 1;
    public MyDatabaseHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_TABLE = "create table " +
             TABLE_NAME
```

```
                + " (_id integer primary key autoincrement, "
                + "_no text not null, "
                + "_name text not null, "
                + "_pnumber text);";
        db.execSQL(CREATE_TABLE);
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        String DROP_TABLE = "DROP TABLE IF EXITS " + TABLE_NAME;
        db.execSQL(DROP_TABLE);
        onCreate(db);
    }
    public long insert(Contact entity) {
        SQLiteDatabase db = getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put("_no", entity.getNo());
        values.put("_name", entity.getName());
        values.put("_pnumber", entity.getPhoneNumber());
        long id = db.insert(TABLE_NAME, null, values);
        db.close();
        return id;
    }
    public int update(Contact entity) {
        SQLiteDatabase db = getWritableDatabase();
        String whereClause = "_no = ?";
        String[] whereArgs = { entity.getNo() };
        ContentValues values = new ContentValues();
        values.put("_no", entity.getNo());
        values.put("_name", entity.getName());
        values.put("_pnumber", entity.getPhoneNumber());
        int rows = db.update(TABLE_NAME, values, whereClause, whereArgs);
        db.close();
        return rows;
    }
    public int delete(Contact entity) {
        SQLiteDatabase db = getWritableDatabase();
        String whereClause = "_no = ?";
        String[] whereArgs = { entity.getNo() };
        int rows = db.delete(TABLE_NAME, whereClause, whereArgs);
        db.close();
        return rows;
    }
    public Cursor query() {
        SQLiteDatabase db = getReadableDatabase();
        return db.query(TABLE_NAME, null, null, null, null, null, null);
    }
}
public class MySimpleAdapter extends SimpleAdapter {
    private ArrayList<Map<String, String>> mData;
    public MySimpleAdapter(Context context, List<? extends Map<String, ?>> data, int resource, String[] from, int[] to) {
        super(context, data, resource, from, to);
        this.mData = (ArrayList<Map<String, String>>) data;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        final int mPosition = position;
        return super.getView(position, convertView, parent);
    }
}
/*In MainActivity*/
private MyDatabaseHelper myDatabaseHelper = new MyDatabaseHelper(MainActivity.this);
private List<Map<String, String>> dataList = new ArrayList<Map<String, String>>();
listview.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
@Override
public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {
    ListView listView = (ListView) parent;
    final HashMap<String, String> map = (HashMap<String, String>)listView.getItemAtPosition(position);
    map.get("no");
    AlertDialog...
    return true;
}});
private void setData(List<Map<String, String>> mDataList) {
    Map<String, String> mData;
    Cursor c = myDatabaseHelper.query();
    while (c.moveToNext()) {
        mData = new HashMap<String, String>();
        mData.put("no", c.getString(c.getColumnIndex("_no")));...
        mDataList.add(mData);
    }
}
private void updateListView() {
    dataList.clear();
    setData(dataList);
    MySimpleAdapter mySimpleAdapter = new MySimpleAdapter(this, dataList, R.layout.contact_item,
            new String[] { "no", "name", "pnumber" },
            new int[] { R.id.contact_no, R.id.contact_name, R.id.contact_phonenumber });
    ListView lv = (ListView)this.findViewById(R.id.lv_contact);
    lv.setAdapter(mySimpleAdapter);
}
```

—— Message, Handler, Thread, Runnable ——

```
private static final int UPDATE_CONTENT = 0;
private static final int TIME_OUT = 1;
private Handler handler = new Handler() {
    @Override
    public void handleMessage(Message message) {
        switch (message.what) {
            case UPDATE_CONTENT: ...
            default: super.handleMessage(message);
        }
    }
};
private void aFunction() {
    Thread workThread = new Thread(new Runnable() {
    @Override
    public void run() {
        Message message = new Message();
        message.what = UPDATE_CONTENT;
        message.obj = parseXMLWithPull(response.toString());
        handler.sendMessage(message);
    }
    workThread.start();
});
}
```

—— Utils ——

```
public android.os.Handler handler = new android.os.Handler();
public Runnable runnable = new Runnable() {
    @Override
    public void run() {
        handler.postDelayed(runnable, 100);
    }
};
handler.removeCallbacks(runnable);
InputMethodManager imm = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
if(imm.isActive()) {
    imm.hideSoftInputFromWindow(EditFileActivity.this.getCurrentFocus().getWindowToken(), InputMethodManager.HIDE_NOT_ALWAYS);
}
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if(keyCode == KeyEvent.KEYCODE_BACK) {...}
    return true;
}
AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
builder.setMessage("Delete?");
builder.setTitle("Hint");
builder.setPositiveButton("Confirm", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {...}
});
builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {...}
});
builder.show();
String newNo = "32654984";
newNo.matches("^[1-9]\\d{7}$");
```

—— AndroidManifest.xml ——

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.wsine.west.finalexam" >
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="com.android.launcher.permission.READ_SETTINGS" />
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<application
    android:allowBackup="true" >
    <activity
        android:name=".MainActivity"
        android:label="@string/app_name" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity
        android:name=".SecondActivity"
        android:label="@string/title_activity_detail"
        android:launchMode="singleInstance" >
    </activity>
    <service android:name=".MyService" android:exported="true"></service>
    <receiver
        android:name=".MyWidgetProvider"
        android:label="@string/app_name">
        <intent-filter>
            <action android:name="android.appwidget.action.APPWIDGET_UPDATE" />
        </intent-filter>
        <meta-data
            android:name="android.appwidget.provider"
            android:resource="@xml/widget_provider" />
    </receiver>
</application>
</manifest>
```