Hướng dẫn lập trình cơ bản với Android

List tutorial

Yêu cầu kiến thức cho lập trình Android:
Để lập trình android, mình nghĩ mọi người chỉ cần kiến thức java căn bản là hoàn toàn ok. Căn bản ở đây có nghĩa là hiểu được thế nào là class, package, biết ý nghĩa của các từ khóa như public, private, protected,... thành thạo các lệnh cơ bản như if, for(), switch(), while(), ... biết sd các lệnh như Integer.parseInt() hay String.valueOf()... Nên có thêm kiến thức về gói java.util vì đây là gói hỗ trợ nhiều lớp rất mạnh được sử dụng trên mọi nền, ngoài ra các gói như java.io, java.net... cũng được recommended

Các kiến thức về các gói lập trình cho desktop như java.awt, java.swing hoàn toàn không cần thiết (bản thân mình cũng chưa sd cái này bao giờ, nhảy vào học java là học J2ME luôn), hay các gói của J2ME cũng vậy Lập trình Android tuy cũng là lập trình di động, nhưng các điện thoại sử dụng hđh Android có cấu hình rất mạnh (Nexus One có VXL lên tới 1Ghz), vì vậy 2 nền tảng Android và J2ME cũng rất khác nhau. Android có những gói riêng hỗ trợ lập trình cho nó và không yêu cầu khắt khe về việc tối ưu code như J2ME. Thật đáng tiếc vì J2ME mình học ko ứng dụng được mấy vào lập trình Android (tuy nhiên 1 số kỹ thuật cơ bản cho lập trình game 2D như Sprite, double buffering, Tile... thì vẫn ko hề phí phạm chút nào)

Cài đặt Android để lập trình:
Để lập trình Android thì mỗi bộ SDK của Google là không đủ, bạn còn cần tích hợp nó vào một IDE như Eclipse. Anh Giáp đã có 2 bài hướng dẫn rất chi tiết về cài đặt Android trong Eclipse cũng như Netbeans, nhưng theo mình mọi người nên sử dụng Eclipse hơn vì nó có nhiều tính năng hỗ trợ lập trình Google, còn Netbeans thì plugin cho Android vẫn chưa hoàn thiện
Eclipse
Netbeans
Tiện thể mình nói luôn, mình học Android theo 2 cuốn *Professional Android Application Development* và *Unlocking Android*. Cả 2 cuốn đều dành cho beginner nhưng cuốn đầu code nhiều, giải thích ít, cuốn thứ 2 giải thích rõ ràng hơn. Nếu có ai có ý định tham khảo thì nên đọc cuốn UA trước để hiểu rõ hơn Android, sử dụng cuốn PAAD trong việc tham khảo các đoạn code cho lập trình.

Understanding Android Application:
Việc hiểu được các thành phần (component) tạo nên một ứng dụng Android là rất cần thiết cho việc lập trình. Các thành phần này được chia làm 6 loại bao gồm:
1.Activity: hiểu một cách đơn giản thì Activity là nền của 1 ứng dụng. Khi khởi động 1 ứng dụng Android nào đó thì bao giờ cũng có 1 main Activity được gọi, hiển thị màn hình giao diện của ứng dụng cho phép người dùng tương tác.
2.Service: thành phần chạy ẩn trong Android. Service sử dụng để update dữ liệu, đưa ra các cảnh báo (Notification) và không bao giờ hiển thị cho người dùng thấy.
3.Content Provider: kho dữ liệu chia sẻ. Content Provider được sử dụng để quản lý và chia sẻ dữ liệu giữa các ứng dụng.

4.Intent: nền tảng để truyền tải các thông báo. Intent được sử dụng để gửi các thông báo đi nhằm khởi tạo 1 Activity hay Service để thực hiện công việc bạn mong muốn. VD: khi mở 1 trang web, bạn gửi 1 intent đi để tạo 1 activity mới hiển thị trang web đó.
5.Broadcast Receiver: thành phần thu nhận các Intent bên ngoài gửi tới. VD: bạn viết 1 chương trình thay thế cho phần gọi điện mặc định của Android, khi đó bạn cần 1 BR để nhận biết các Intent là các cuộc gọi tới.
6.Notification: đưa ra các cảnh báo mà không làm cho các Activity phải ngừng hoạt động.

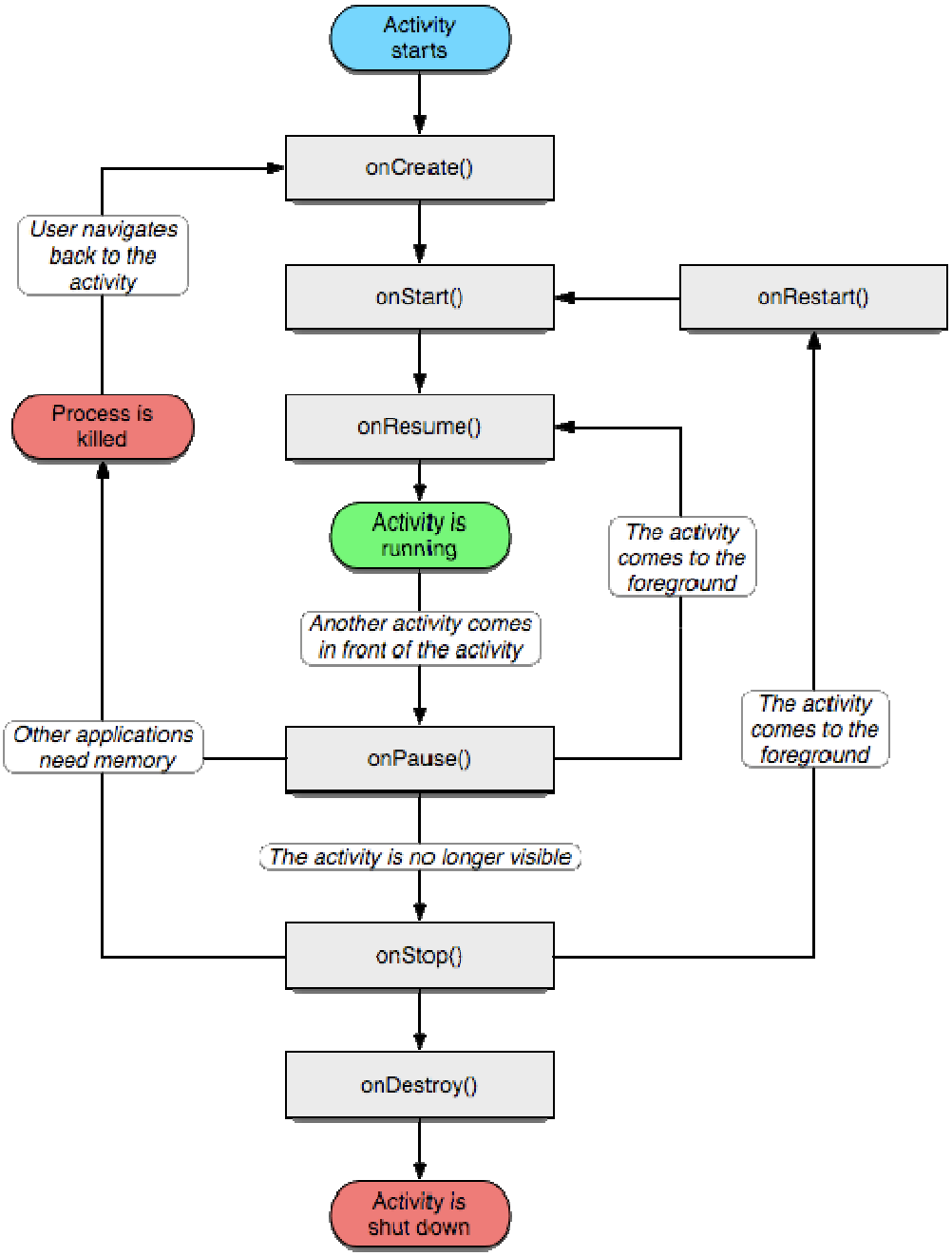
Activity, Service, Broadcast Receiver và Content Provider mới là những thành phần chính cấu thành nên ứng dụng Android, bắt buộc phải khai báo trong AndroidManifest (tham khảo bài 2 có giới thiệu đầy đủ về file này).

Understanding Android Application Life Cycle:
Android có cơ chế quản lý các process theo chế độ ưu tiên. Các process có priority thấp sẽ bị Android giải phóng mà không hề cảnh báo nhằm đảm bảo tài nguyên.
1.Foreground process: là process của ứng dụng hiện thời đang được người dùng tương tác.
2.Visible process: là process của ứng dụng mà activity đang hiển thị đối với người dùng (onPaused() của activity được gọi).
3.Service process: là Service đang running.
4.Background process: là process của ứng dụng mà các activity của nó ko hiển thị với người dùng (onStoped() của activity được gọi).
5.Empty process: process không có bất cứ 1 thành phần nào active.
Theo chế độ ưu tiên thì khi cần tài nguyên, Android sẽ tự động kill process, trước tiên là các empty process.

Android Activity Life Cycle:
Như mình đã giới thiệu ở trên , Actitvity là thành phần quan trọng nhất và đóng vai trò chính trong xây dựng ứng dụng Android. Hệ điều hành Android quản lý Activity theo dạng stack: khi một Activity mới được khởi tạo, nó sẽ được xếp lên đầu của stack và trở thành ***running activity***, các Activity trước đó sẽ bị tạm dừng và chỉ hoạt động trở lại khi Activity mới được giải phóng.

Activity bao gồm 4 state:
- ***active (running):*** Activity đang hiển thị trên màn hình (foreground).
- ***paused:*** Activity vẫn hiển thị (visible) nhưng không thể tương tác (lost focus). VD: một activity mới xuất hiện hiển thị giao diện đè lên trên activity cũ, nhưng giao diện này nhỏ hơn giao diện của activity cũ, do đó ta vẫn thấy được 1 phần giao diện của activity cũ nhưng lại không thể tương tác với nó.
- ***stop:*** Activity bị thay thế hoàn toàn bởi Activity mới sẽ tiến đến trạng thái ***stop***
- ***killed:*** Khi hệ thống bị thiếu bộ nhớ, nó sẽ giải phóng các tiến trình theo nguyên tắc ưu tiên. Các Activity ở trạng thái ***stop*** hoặc ***paused*** cũng có thể bị giải phóng và khi nó được hiển thị lại thì các Activity này phải khởi động lại hoàn toàn và phục hồi lại trạng thái trước đó.

Biểu đồ miêu tả Activity state



Vòng đời của Activity:
- ***Entire lifetime:*** Từ phương thức **onCreate( )** cho tới **onDestroy( )**
- ***Visible liftetime:*** Từ phương thức **onStart( )** cho tới **onStop( )**
- ***Foreground lifetime:*** Từ phương thức **onResume( )** cho tới **onPause( )**

Khi xây dựng Actitvity cho ứng dụng cần phải viết lại phương thức **onCreate( )** để thực hiện quá trình khởi tạo. Các phương thức khác có cần viết lại hay không tùy vào yêu cầu lập trình.

XML trong Android:
Không giống như lập trình java thông thường, lập trình android ngoài các lớp được viết trong *.java còn sử dụng XML để thiết kế giao diện cho ứng dụng. Tất nhiên bạn hoàn toàn có thể thiết kế 1 giao diện như ý muốn mà không cần tới bất cứ 1 dòng XML nào, nhưng sd XML sẽ đơn giản công việc đi rất nhiều. Đồng thời sd XML sẽ giúp việc chỉnh sửa ứng dụng sau này trở nên dễ dàng.

Về nguyên tắc, khi lập trình ứng dụng ta thiết kế giao diện bằng XML và cài đặt các xử lý khi tương tác với giao diện trong code.

1 số thành phần cơ bản trong Android:
1.Các layout:
Layout được dùng để quản lý các thành phần giao diện khác theo 1 trật tự nhất định.
- FrameLayout: Layout đơn giản nhất, thêm các thành phần con vào góc trên bên trái của màn hình.
- LinearLayout: thêm các thành phần con theo 1 chiều nhất định (ngang hoặc dọc). Đây là layout được sử dụng nhiều nhất.
- RelativeLayout: thêm các thành phần con dựa trên mối quan hệ với các thành phần khác hoặc với biên của layout.
- TableLayout: thêm các thành phần con dựa trên 1 lưới các ô ngang và dọc.
- AbsoluteLayout: thêm các thành phần con dựa theo tọa độ x, y.

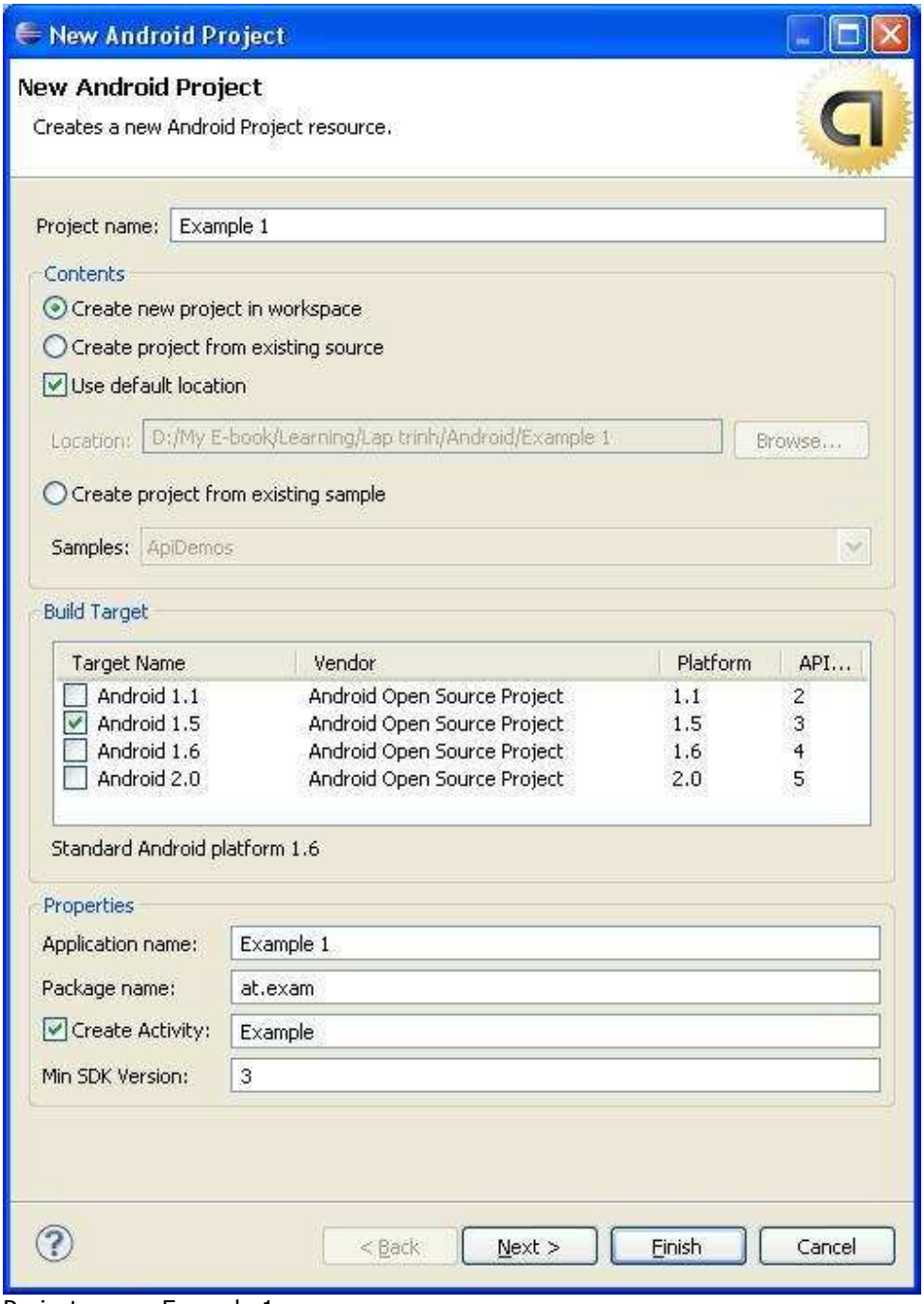
Layout được sử dụng nhằm mục đích thiết kế giao diện cho nhiều độ phân giải. Thường khi lập trình nên kết hợp nhiều layout với nhau để tạo ra giao diện bạn mong muốn.

2.XML unit:
Để hiểu được các thành phần cơ bản của XML cũng như việc sử dụng XML kết hợp với code, ta sẽ đi xây dựng thử một chương trình đơn giản.

Yêu cầu: Xây dựng 1 ứng dụng cho phép gõ 1 nội dung vào rồi hiển thị ra nội dung đó ở bên dưới.

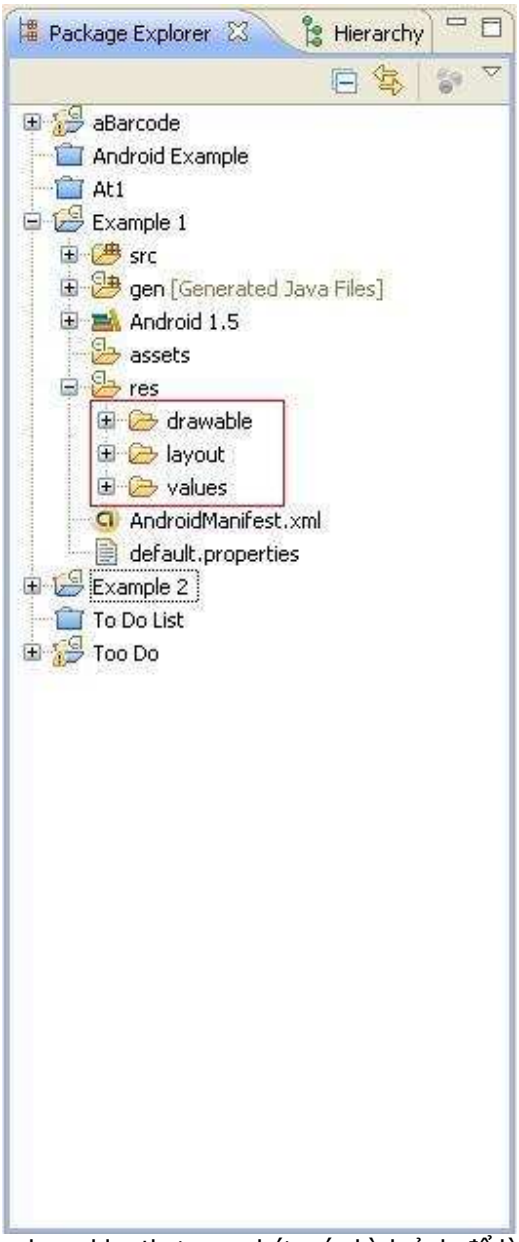
B1: Khởi tạo 1 project (ở đây sử dụng Eclipse để minh họa).
Vào thẻ File -> New -> Android Project. Nếu bạn mới lập trình Android lần đầu thì có lẽ dòng Android Project sẽ không hiện ra, khi đó xuống phía cuối chọn Other rồi vào Android -> Android Project.

B2: Điền thông tin cho project



Project name: Example 1
Build Target: Chọn Android 1.5 (mới nhất là 2.1 nhưng hiện tại bạn chưa cần quan tâm)
Application name: Example 1
Package name: at.exam
Create Activity: Example
=> Kích nút Finish.

B3: Bên khung Package Explore bên trái đi tới thư mục res, bạn sẽ thấy có 3 thư mục con:



- drawable: thư mục chứa các hình ảnh để làm icon hoặc tài nguyên cho giao diện...
- layout: chứa các file xml để thiết kế giao diện.
- values: chứa các giá trị sử dụng trong ứng dụng được bạn định nghĩa, như các dòng ký tự (string), các màu (color), các themes...

B4:Vào thư mục layout, chọn file main.xml và gõ đoạn code sau vào thay cho toàn bộ nội dung có sẵn (Eclipse hỗ trợ kéo thả cho xml nhưng theo mình không nên sử dụng):

Mã:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
```

```
    >
    <EditText
        android:id="@+id/edit_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="@string/edit_hint"
    />
    <TextView
        android:id="@+id/text_view"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="@color/text_color"
        android:textSize="28px"
        android:typeface="monospace"
    />
</LinearLayout>
```

Trong đoạn XML này chúng ta khai báo một Linear Layout với 2 thành phần con của nó là 1 Edit Text (dùng để gõ xâu ký tự) với 1 Text View (hiển thị xâu ký tự). Linear Layout được khai báo với từ khóa orientation nhằm chỉ ra chiều sắp xếp của 2 thành phần con là chiều dọc. Còn với layout_width, layout_height các bạn có thể cho giá trị bằng "fill_parent" hoặc "wrap_content" để thông báo thành phần này sẽ có chiều rộng (dài) phủ đầy thành phần cha hoặc chỉ vừa bao đủ nội dung.
Trong Edit Text và Text View các bạn có thể thấy có từ khóa id, từ khóa này cho phép khai báo id của các thành phần để lấy về trong code (sẽ đề cập sau).
Ngoài ra từ khóa hint trong Edit Text cho phép hiện ra phần nội dung mờ khi Edit Text vẫn chưa có ký tự nào. "@string/edit_hint" thông báo lấy trong file strings.xml xâu có tên là edit_hint.
Còn textColor của Text View thì thông báo đoạn ký tự sẽ được hiển thị với màu lấy trong file colors.xml, textSize chỉ ra cỡ chữ bằng 28 pixel và typeface chỉ ra kiểu chữ là monospace

B5: Vẫn trong thư mục res, vào values và chọn file strings.xml. Bổ sung thêm dòng định nghĩa cho edit_hint như sau:
Mã:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, Example!</string>
    <string name="app_name">Example 1</string>
    <string name="edit_hint">Enter the work here</string>
</resources>
```

B6: Trong thư mục values, tạo file colors.xml (chuột phải vào thư mục, chọn New -> Android XML File, và lưu ý chữ s, không phải là color.xml). Gõ nội dung cho file như sau:
Mã:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="text_color">#ff3300</color>
</resources>
```

OK, vậy là bạn đã tạo một màu mới cho dòng chữ sẽ được hiển thị trong Text View (ff3300 là mã hexa của màu đỏ). Thực chất bạn hoàn toàn có thể gõ thẳng
Mã:

```
android:textColor="#ff3300"
```

trong file main.xml mà không cần tạo mới file colors.xml, nhưng mục đích của XML trong Android chính là để hỗ trợ nâng cấp chỉnh sửa dễ dàng. Nếu sau này bạn muốn sửa màu của dòng text thì chỉ cần vào colors.xml thay đổi thay vì mò mẫm trong main.xml (có thể rất dài nếu giao diện

phức tạp).

Các thành phần trên mới chỉ là các phần cơ bản của XML. Ngoài ra các bạn có thể khai báo thêm về Animation, Style và Theme (phức tạp hơn nhiều nên mình không giới thiệu trong phần cơ bản này).

_____

B7: Vậy là chúng ta đã hoàn thiện phần giao diện với XML, giờ đến viết code để xử lý các sự kiện cho các thành phần:
=> vào thư mục src (source code của project) => at.exam => Example.java, gõ nội dung code sau vào:
Mã:

```
package at.exam;

import android.app.Activity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnKeyListener;
import android.widget.EditText;
import android.widget.TextView;

public class Example extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //Thiết lập giao diện lấy từ file main.xml
        setContentView(R.layout.main);

        //Lấy về các thành phần trong main.xml thông qua id
        final EditText edit = (EditText) findViewById(R.id.edit_text);
        final TextView text = (TextView) findViewById(R.id.text_view);

        //Thiết lập xử lý cho sự kiện nhấn nút giữa của điện thoại
        edit.setOnKeyListener(new OnKeyListener() {
            @Override
            public boolean onKey(View v, int keyCode, KeyEvent event) {
                if (event.getAction() == KeyEvent.ACTION_DOWN
                        && keyCode == KeyEvent.KEYCODE_DPAD_CENTER) {
                    text.setText(edit.getText().toString());
                    edit.setText("");
                    return true;
                }
                else {
                    return false;
                }
            }

        });
    }
}
```

Dạo qua một chút kiến thức cơ bản: Trong Android, các lớp sử dụng để tạo giao diện (Edit Text, Text View...) đều là lớp con của lớp View. Một số lớp thường xuyên được sử dụng để tạo giao diện:
- TextView
- EditText
- ListView
- Spinner
- CheckBox
- Button
- RadioButton

Ngoài ra bạn còn có thể tạo 1 View riêng của mình bằng cách kế thừa View có sẵn.

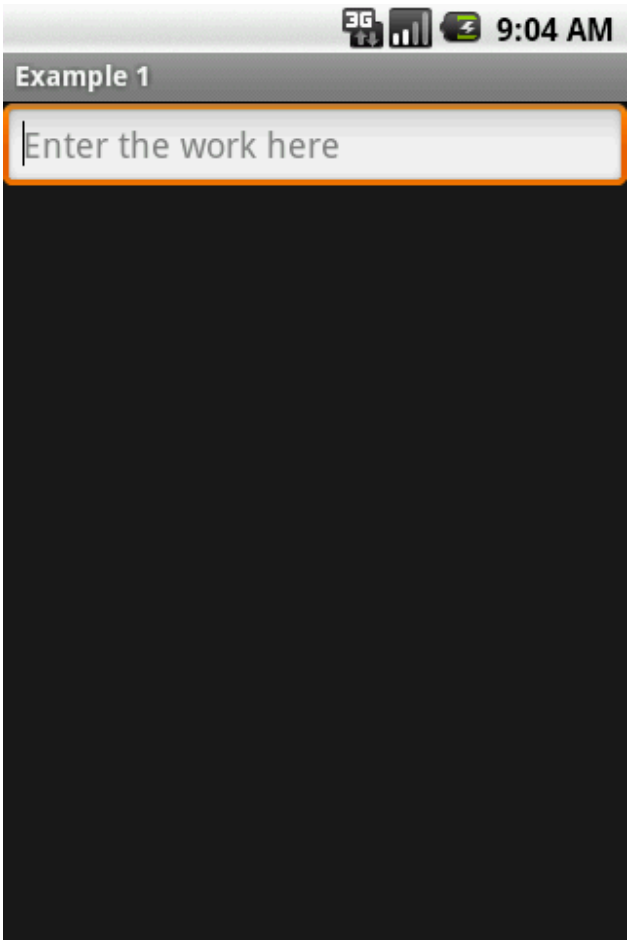
Các Listener được sử dụng để bắt 1 sự kiện nào đó. Ở đây mình sử dụng OnKeyListener dùng để bắt sự kiện khi nhấn 1 phím của điện thoại. Ngoài ra thường sử dụng OnClickListener để bắt sự kiện chạm vào 1 View đang hiển thị trên màn hình. Mỗi View đều phải set Listener riêng để xử lý cho sự kiện tương tác với nó, và mỗi loại View cũng lại có những Listener dành riêng cho nó (VD: CheckBox có OnCheckChangedListener)
Ở đây mình sử dụng hàm dạng inner để định nghĩa xử lý cho OnKeyListener nên có thể mọi người không quen lắm, nhưng nó cũng nằm trong phần cơ bản của Java đấy nhé.

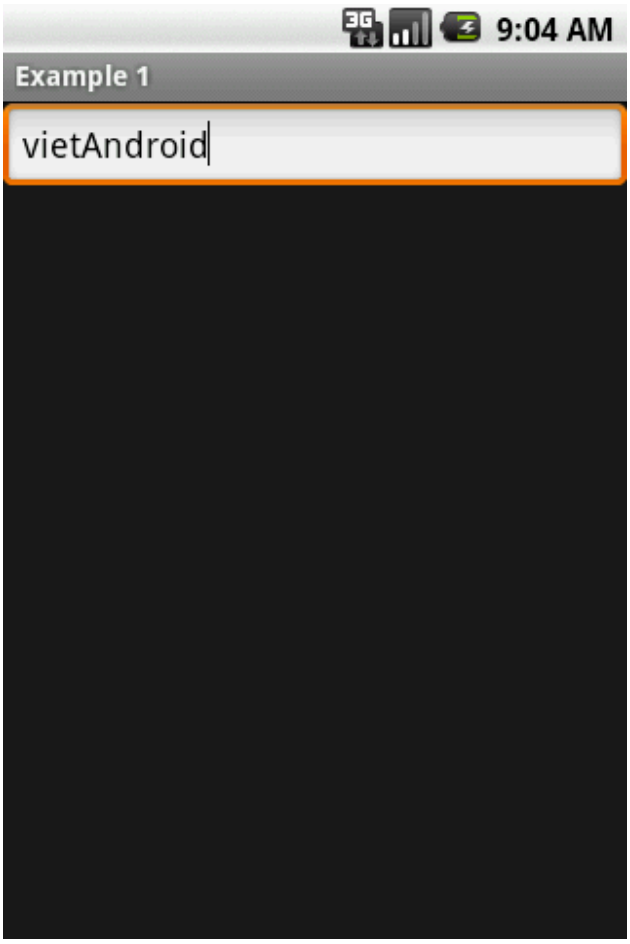
Đề nghị lưu ý thêm phần R.id.edit_text. Để lấy hoặc truy nhập các thành phần ta đã định nghĩa trong XML ta phải sử dụng R.* như R.layout.main, R.id.edit_text. Lệnh findViewById sẽ trả về 1 View có Id thiết lập trong phần XML. Do View là lớp cha của EditText với TextView nên ở đây ta phải ép kiểu.
Ngoài ra các string hay color cũng có thể lấy về bằng lệnh getResource() . Vd: getResource().getColor(R.color.text_color)

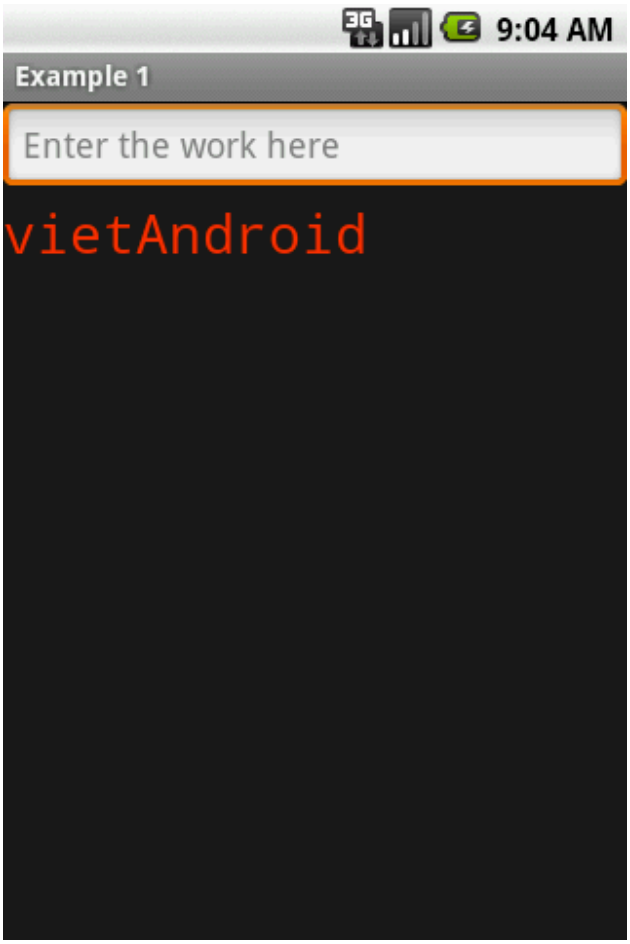
B8: Chạy chương trình. Chọn Run => Android Application và chờ cho emulator khởi động nhé. Ai có 1 Android thật có thể kết nối qua USB và thử nghiệm luôn. Tự chỉnh sửa trong code và trong XML để hiểu thêm về lập trình Android.
VD:
Mã:
```
edit.setOnClickListener(new OnClickListener() {
                        @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub

            }

        });
```
hoặc trong XML thêm vào phần Text View
Mã:
```
android:textSize="50px"
```
để xem chương trình thay đổi như thế nào nhé ^_^


9:04 AM
Example 1
Enter the work here


9:04 AM
Example 1
vietAndroid



Kết thúc bài 1