Fpt University

FPT POLYTECHNIC

THIẾT KẾ GIAO DIỆN TRÊN ANDROID

Bài 6: Dialog –Input Keyboard-Search

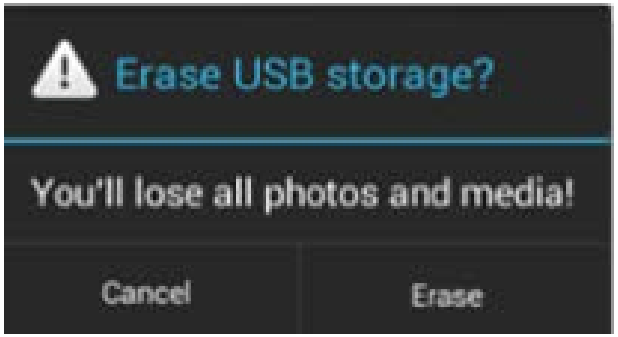
www.poly.edu.vn

Nội dung bài học

1. Toast & AlertDialog
2. Input Keyboard
3. Search

1. Toast & Alert Dialog

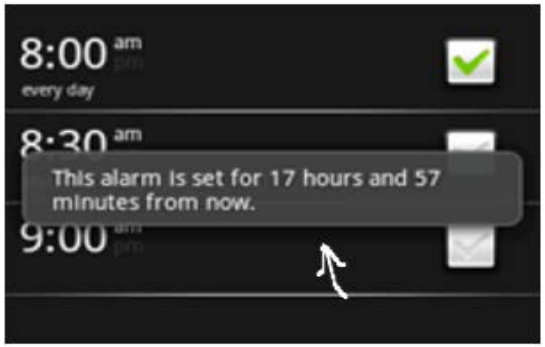
- Sử dụng Toast và Alert Dialog để kiểm tra một điều gì đó, hay đơn giản chỉ là xuất thông báo.
- Cả Toast và Alert Dialog khi hiển thị lên thì các tiến trình (hay các lệnh) khác vẫn cứ tiếp tục làm việc.



1. Toast & Alert Dialog

a) Toast

- Có thể sử dụng trong trường hợp hiển thông báo trong các mục thiết lập thông số cấu hình, hay đơn giản chỉ là hiển thị lên để xem thông tin tạm thời nào đó (giống như để kiểm tra một vấn đề xảy ra chẳng hạn).
- Toast có thể được tạo và hiển thị trong Activity hoặc trong Servive.
- Có thể thiết lập vị trí hiển thị toast.
  Ví dụ: Toast sẽ xuất hiện ở góc bên trái, trên cùng ta viết như sau:




```
toast.setGravity(Gravity.TOP|Gravity.LEFT, 0, 0);
```

1. Toast & Alert Dialog

a) Toast (tiếp)

- Không cho phép người sử dụng tương tác
- Khi hiển thị sau khoảng thời gian nào đó sẽ tự đóng lại.
- Có 2 giá trị mặc định (ta nên sử dụng 2 giá trị này, không nên gõ con số cụ thể vào):
  - **Toast.LENGTH_SHORT** hiển thị trong 2 giây,
  - **Toast.LENGTH_LONG** hiển thị trong 3.5 giây.
- Cách định nghĩa một Toast:

```
Context context = getApplicationContext();
CharSequence text = "Hello toast!";
int duration = Toast.LENGTH_SHORT;
Toast toast = Toast.makeText(context, text, duration);
toast.show();
```

1. Toast & Alert Dialog

a) Toast (tiếp): Tạo một Custom Toast

- Tạo một tệp toast_layout.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/
    res/android"
            android:id="@+id/toast_layout_root"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:padding="8dp"
            android:background="#DAAA"
            >
      <ImageView android:src="@drawable/droid"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:layout_marginRight="8dp"
             />
      <TextView android:id="@+id/text"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:textColor="#FFF"
             />
</LinearLayout>
```

1. Toast & Alert Dialog

a) Toast (tiếp): Tạo một Custom Toast

- Sử dụng custom toast bằng cách gọi id: toast_layout_root

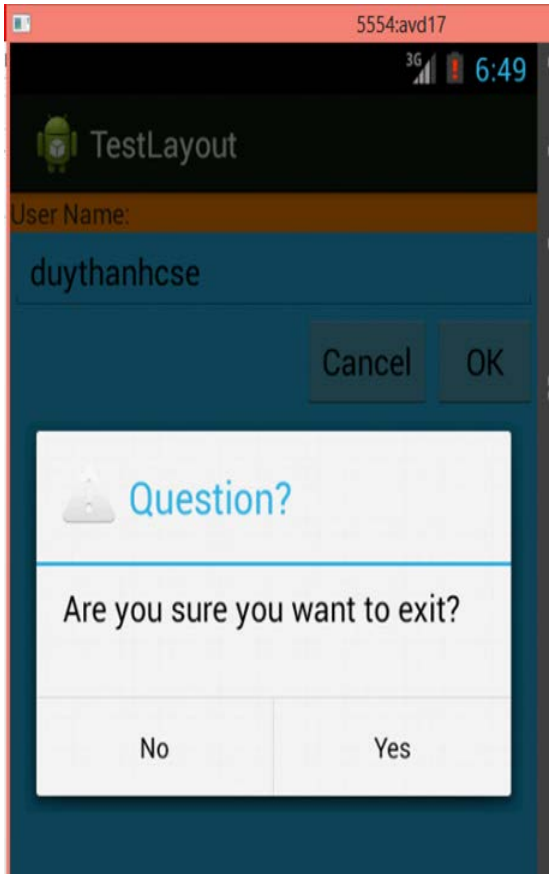
```
LayoutInflater inflater = getLayoutInflater();
    View layout = inflater.inflate(R.layout.custom_toast,
                                   (ViewGroup) findViewById(R.id.toast_layout_root));
    TextView text = (TextView) layout.findViewById(R.id.text);
    text.setText("This is a custom toast");

    Toast toast = new Toast(getApplicationContext());
    toast.setGravity(Gravity.CENTER_VERTICAL, 0, 0);
    toast.setDuration(Toast.LENGTH_LONG);
    toast.setView(layout);
    toast.show();
```

1. Toast & Alert Dialog

b) Alert Dialog

- Hiển thị và cho phép người dùng tương tác,
- Ví dụ như hình bên dưới, khi nhấn nút “Cancel”, chương trình sẽ hiển thị Alert Dialog hỏi xem có chắc chắn muốn xóa hay không? Bấm No thì không, bấm Yes thì tắt chương trình.



1. Toast & Alert Dialog

b) Alert Dialog (tiếp)

- **Cách tạo Alert Dialog:**

```
AlertDialog.Builder b=new AlertDialog.Builder(YourActivity.this);
 b.setTitle("Question");
b.setMessage("Are you sure you want to exit?");
b.setPositiveButton("Yes", new DialogInterface. OnClickListener()
 {
            @Override
            public void onClick(DialogInterface dialog, int which)
            {            finish();
}});
b.setNegativeButton("No", new DialogInterface.OnClickListener()
{
            @Override
            public void onClick(DialogInterface dialog, int which)
            {            dialog.cancel();}
});
b.create().show();
```

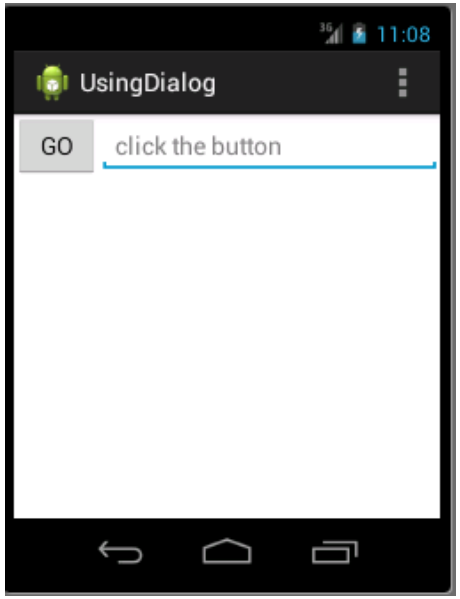
1. Toast & Alert Dialog

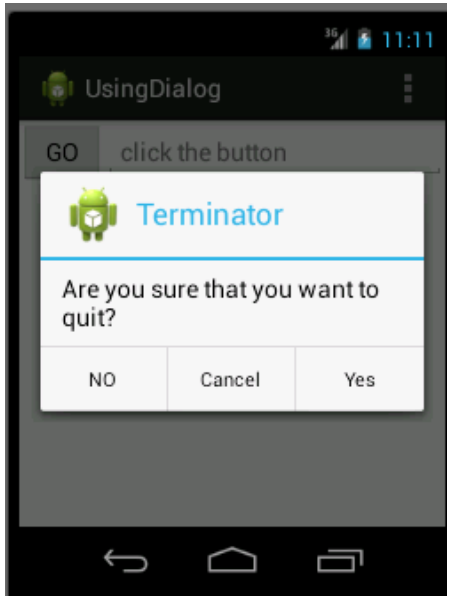
b) Alert Dialog (tiếp)

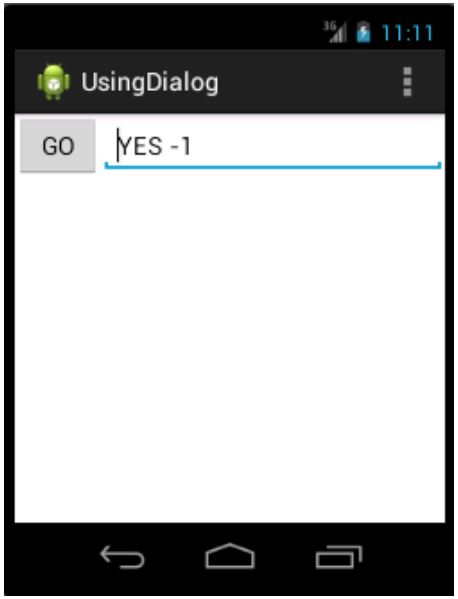
- **Ý nghĩa của các hàm :**
  - **setTitle** : thiết lập tiêu đề cho Dialog
  - **setMessage**: thiết lập nội dung cho Dialog
  - **setIcon** : thiết lập Icon
  - **setPositiveButton**, **setNegativeButton** thiết lập hiển thị nút chọn cho Dialog. Chú ý là ở đối số thứ 2 của các hàm này sẽ là **DialogInterface**.OnClickListener chứ không phải **View.**OnClickListener
  - **create**() để tạo Dialog
  - **show**() để hiển thị Dialog.

1. Toast & Alert Dialog

- **Demo → Toast & Alert Dialog**



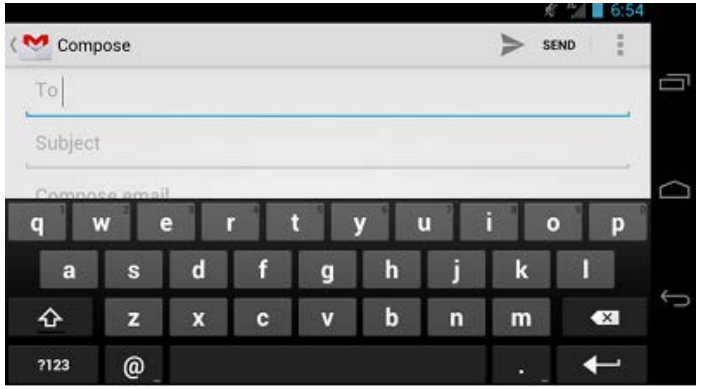




2. Input Keyboard

TextField:

- TextField cho phép nhập nội dung text vào ứng dụng.
- Khi đưa con trỏ vào ô nhập, một loại keyboard sẽ tự động hiển thị.
- Ngoài ra, text fields cũng cho phép thực hiện các hoạt động khác như: copy, cut, paste và tìm kiếm dữ liệu…
- Sử dụng thẻ <EditText>



2. Input Keyboard

(1)Keyboard Type

- TextFields có thể nhập nhiều loại khác nhau: number, date, password, email, adress..
- Với mỗi kiểu sẽ hiển thị dạng keyboard khác nhau.
- Ta có thể sử dụng thuộc tính android:inputType trong thẻ EditText để khai báo loại dữ liệu nhập :

```
<EditText
        android:id="@+id/email_address"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="@string/email_hint"
        android:inputType="textEmailAddress" />
```

2. Input Keyboard

(1)Keyboard Type

android:inputType =“….”

- "text" : bàn phím text thông thường.
- "textEmailAddress“: nhập text có chứa ký tự @ ở keyboard
- "textUri“ : nhập text có chứa ký tự /
- "number" : keyboard có chứa dạng số.
- "phone“: keyboard có dạng kiểu phone để nhập số điện thoại.

Figure 1. The default `text` input type.

Figure 2. The `textEmailAddress` input type.

Figure 3. The `phone` input type.

2. Input Keyboard

(1)Keyboard Type

Một số thuộc tính thông dụng khác:

- "textCapSentences“: viết hoa đầu mỗi câu
- "textCapWords“: viết hoa đầu mỗi từ. Ví dụ như họ tên
- "textAutoCorrect" : tự động chỉnh lỗi chính tả.
- "textPassword”: ký tự nhập vào chuyển sang dạng “.”
- "textMultiLine“: nhập ký tự dài.

```
<EditText
        android:id="@+id/postal_address"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="@string/postal_address_hint"
        android:inputType="textPostalAddress|textCapWords| textNoSuggestions" />
```

2. Input Keyboard

(2)Keyboard Actions: định nghia một Keyboard Action

- Cho phép hiển thị một số action như Search, Send sau khi nhập xong sẽ sử dụng.
- Sử dụng thuộc tính android:imeOptions = “actionSend” hoặc “actionSearch” hoặc “actionNone”
- Ví dụ: nếu khai báo thì keyboard sẽ có cả nút Send


```
<EditText
        android:id="@+id/search"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="@string/search_hint"
        android:inputType="text"
        android:imeOptions="actionSend" />
```

2. Input Keyboard

(2)Keyboard Actions: thực hiện hành động của nút action

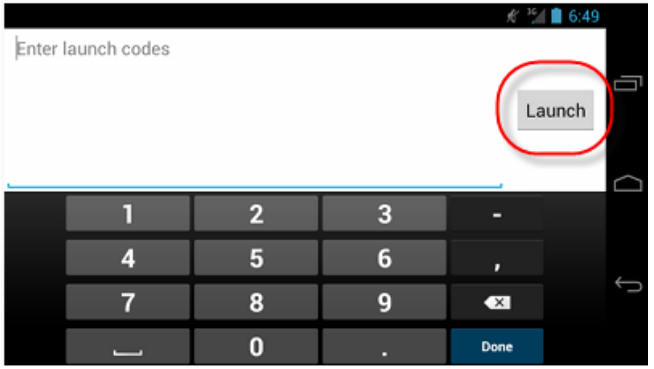
- Cho phép thực hiện hành động của nút action, tùy loại mà ta khai báo ID khác nhau: **IME_ACTION_SEND** hoặc **IME_ACITON_SEARCH:**

```
EditText editText = (EditText) findViewById(R.id.search);
editText.setOnEditorActionListener(new OnEditorActionListener()
 {
   @Override
   public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
      boolean handled = false;
      if (actionId == EditorInfo.IME_ACTION_SEND) {
         sendMessage();
         handled = true;
      }
      return handled;
   }});
```

2. Input Keyboard

(2)Keyboard Actions: tạo một nhãn riêng cho nút action

- Sử dụng thuộc tính android:imeActionLabel để tạo nhãn riêng cho nút action muốn thực hiện

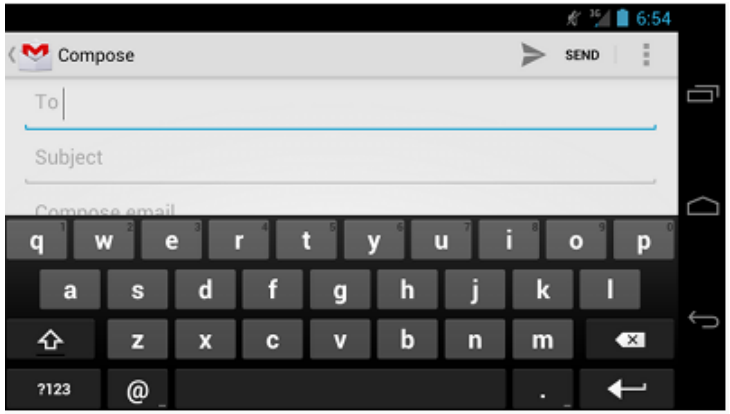



```
<EditText
        android:id="@+id/launch_codes"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="@string/enter_launch_codes"
        android:inputType="number"
        android:imeActionLabel="@string/launch" />
```

2. Input Keyboard

(3)Keyboard Flags

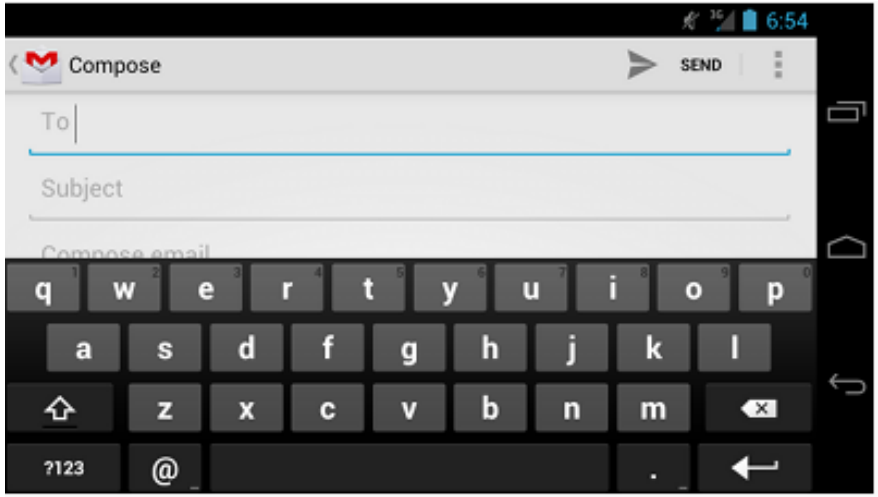
- Một vài flag được sử dụng với thuộc tính android:imeOptions
  - flagNoFullscreen;
  - flagNavigatePrevious;
  - flagNavigateNext;
  - flagNoExtractUi;
  - flagNoAccessoryAction;
  - flagNoEnterAction;
  - flagForceAscii



- Ví dụ: android:imeOptions="flagNoExtractUi“: làm cho màn hình nhập dữ liệu không bị fullscreen

2. Input Keyboard

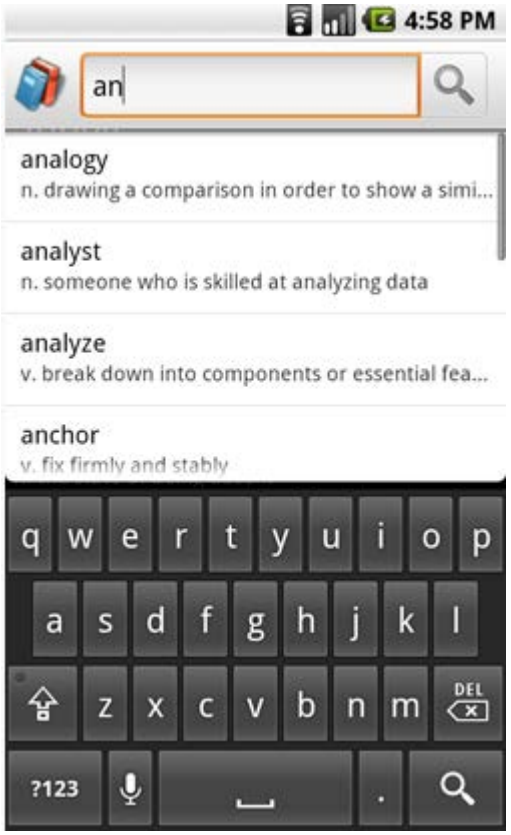
- **Demo → Input Keyboard**



3. Search

1. Tổng quan về Search

- Android cung cấp 2 loại search: search dialog và search widget
- Tính năng sẵn có:
  - Tìm kiếm theo voice
  - Cung cấp những từ gợi ý sau khi người dùng nhập vài ký tự.
  - Cung cấp từ gợi ý dựa trên dữ liệu đã xây dựng.



3. Search

a)Tạo một Searchable configuration

- Tạo một file searchable.xml trong thư mục /res/xml. Cấu hình để gán các thuộc tính như: voice search, search suggestion hoặc hint text cho hộp thoại search.

Ví dụ:

```
<?xml version="1.0" encoding="utf-8"?>
    <searchable
    xmlns:android="http://schemas.android.com/apk/res/android"
     android:label="@string/app_label"
       android:hint="@string/search_hint”
       android:searchSuggestAuthority="dictionary"
       android:searchSuggestIntentAction="android.intent.action.VIEW"
       android:includeInGlobalSearch="true"
       android:searchSettingsDescription="@string/settings_description" >
    </searchable>
```

3. Search

b)Tạo một Searchable Activity

(1)Khai báo searchable activity

- Khai báo activity để chấp nhận ACTION_SEARCH và thẻ <intent-filter>
- Chỉ định thẻ searchable để sử dụng, được định nghĩa trong thẻ <meta-data>

```
<application ... >
        <activity android:name=".SearchableActivity" >
          <intent-filter>
            <action android:name="android.intent.action.SEARCH" />
          </intent-filter>
          <meta-data android:name="android.app.searchable"
                  android:resource="@xml/searchable"/>
        </activity>
        </application>
```

3. Search

b)Tạo một Searchable Activity

- (2)Thực hiện search: 3 bước

Bước 1:Receiving the query

```
@Override
        public void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           setContentView(R.layout.search);
           // Get the intent, verify the action and get the query
           Intent intent = getIntent();
           if (Intent.ACTION_SEARCH.equals(intent.getAction())) {
            String query = intent.getStringExtra(SearchManager.QUERY);
            doMySearch(query);
           }}
```

Ở ví dụ trên, chuỗi truy vấn được lấy ra và truyền vào phương thức doMySearch() để thực hiện hành động tìm kiếm

3. Search

b)Tạo một Searchable Activity

Bước 2: Searching your data

Ta có thể khai báo chuỗi gợi ý cho tìm kiếm bằng cách sử dụng ArrayAdapter như đã học ở bài trước.(ListView)

Bước 3: Presenting the results

Hiển thị kết quả tìm kiếm bằng cách sử dụng ListView

3. Search

2. Search dựa trên những từ gợi ý gần đây đã sử dụng

Để tạo một truy vấn sử dụng những từ đã truy cập gần đây cần thực hiện:

- (1)Tạo một searchable activity
- (2)Tạo một content provider để chứa nội dung search, được kế thừa từ SearchRecentSuggestionsProvider và được khai báo trong manifest.
- (3)Chỉnh sửa cấu hình searchable
- (4)Lưu chuỗi truy vấn vào content provider mỗi lần thực hiện lệnh search.

3. Search

- **Tạo một content provider**

```
public class MySuggestionProvider extends SearchRecentSuggestionsProvider
{
  public final static String AUTHORITY = "com.example.MySuggestionProvider";
 public final static int MODE = DATABASE_MODE_QUERIES;
  public MySuggestionProvider() {
                setupSuggestions(AUTHORITY, MODE); }
}
```

- Khai báo provider trong manifest <application> với cùng chuỗi authorities:

```
<application>
   <provider android:name=".MySuggestionProvider"
          android:authorities="com.example.MySuggestionProvider" />
   ...
</application>
```

3. Search

Chỉnh sửa cấu hình thẻ searchable

```
<?xml version="1.0" encoding="utf-8"?>
<searchable xmlns:android="http://schemas.android.com/apk/res/android"
        android:label="@string/app_label"
        android:hint="@string/search_hint"
        android:searchSuggestAuthority="com.example.MySuggestionProvider"
         android:searchSuggestSelection=" ?" >
</searchable>
```

- Trong đó giá trị của **android:searchSuggestAuthority** phải trùng với tên của content provider đã khai báo ở trên.
- Giá trị của **android:searchSuggestSelection =“?”**

3. Search

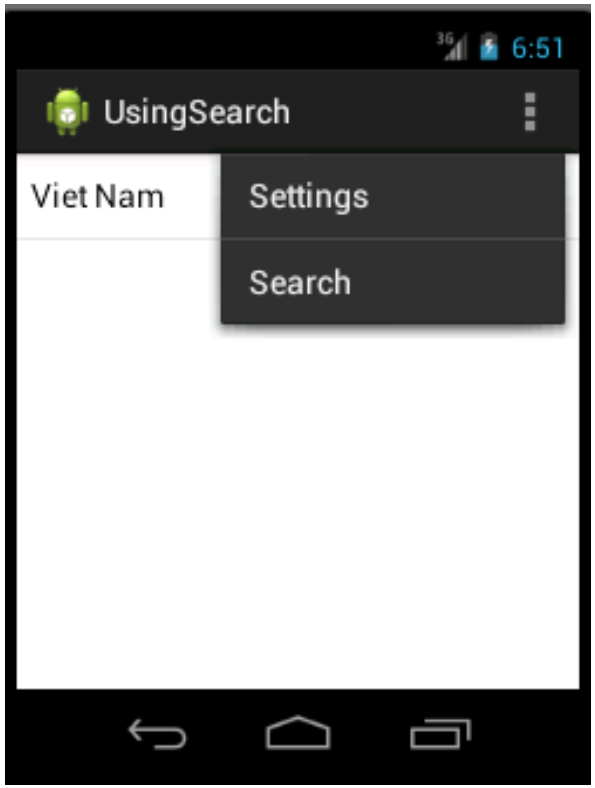
Lưu chuỗi truy vấn

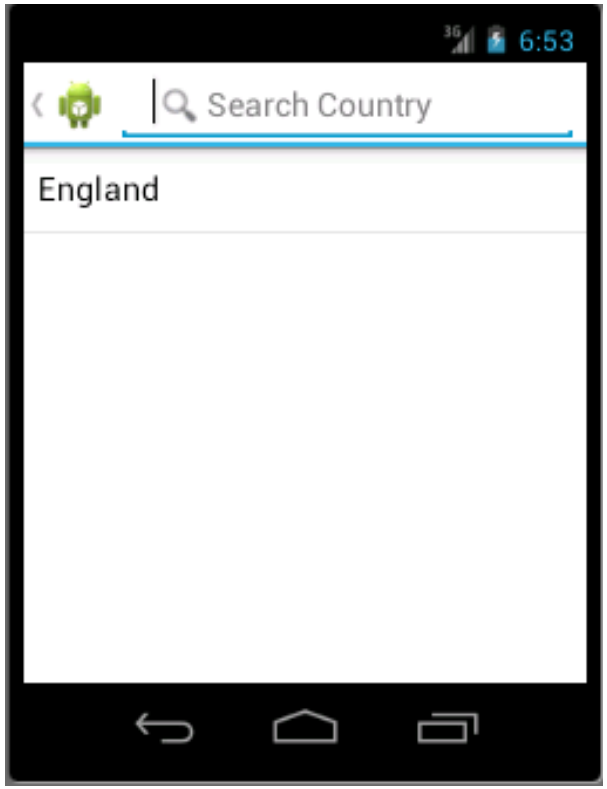
```
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
Intent intent  = getIntent();
if (Intent.ACTION_SEARCH.equals(intent.getAction())) {
String query = intent.getStringExtra(SearchManager.QUERY);
SearchRecentSuggestions suggestions = new SearchRecentSuggestions(this,
        MySuggestionProvider.AUTHORITY, MySuggestionProvider.MODE);
suggestions.saveRecentQuery(query, null);
}
}
```

- Chú ý phương thức **saveRecentQuery()**
- Để xóa lịch sửa truy vấn sử dụng hàm: **.clearHistory()**

3. Search

- **Demo → Search**





Tổng kết bài học

Tìm hiểu cách sử dụng:

- Toast & AlertDialog
- Input Keyboard
- Search


Kết thúc!