LẬP TRÌNH ANDROID

1. Intent

* Ba trong số những thành phần ứng dụng cốt lõi trong một dự án Android là: Activity, Service và Broadcast Receiver, được kích hoạt thông qua các thông điệp bất đồng bộ được gọi là Intent. Thông điệp này dùng để kết nối các thành phần trong một dự án với nhau, hoặc giữa các dự án với nhau.
* Intent rất quan trọng, nó là linh hồn của Android, là hạt nhân để Android tồn tại
* Intent có 5 công dụng chính

1. Dùng Intent để mở một Activity
2. Dùng Intent để mở một Activity khác và truyền dữ liệu
3. Mở Activity, có kiểm soát kết quả trả về
4. Liên kết Application Components: Activity, Service và Broadcast Receiver trong 1 ứng dụng
5. Kết nối các thành phần trong các ứng dụng khác nhau và giữa HĐH với ứng dụng

* Phân loại Intent

1. **Explicit Intent :** Hay còn gọi là Intent công khai. Với loại Intent này, tên của Application Component đích phải được chỉ định và nêu rõ tên

Ví dụ : Intent myintent = new Intent(MainActivity.this, ChildActivity. class);

1. **Implicit Intent:** Hay còn gọi là Intent ẩn, với loại Intent này, tên của Application component đích không được chỉ định, chúng ta cần thiết lập thông tin về hành động (Action), Dữ liệu (Data), hay phân loại (category)

Ví dụ: Intent myintent = new Intent(Intent.Action\_Call, Uri.parse(“tel:555-2368”));

* Một số hành động thường dùng trong Implicit Intent

**ACTION\_CALL**: Mở 1 phone Dialer và ngay lập tức thực hiện cuộc gọi dựa vào thông tin số điện thoại trong data URI

**ACTION\_DIAL**: Mở 1 phone Dialer (mặc định là PD của Android) và điền thông tin số điện thoại chứa trong data URI

**ACTION\_SEND**: Mở 1 Activity cho phép gửi dữ liệ lấy từ data URI, kiểu của dữ liệu xác định trong thuộc tính type

**ACTION\_SENDTO**: Mở 1 Activity cho phép gửi thông điệp (Message) tới địa chỉ lấy từ data URI

**ACTION\_VIEW**: Action thông dụng nhất, khởi chạy Activity thích hợp để hiển thị dữ liệu trong data URI (hình ảnh, web, map,…)

**ACTION\_IMAGE-CAPTURE/ACTION\_VIDIEO\_CAPTURE** : Mở ứng dụng cho phép chụp ảnh, qay vidieo

1. Intent Filter

* Với Explicit Intent : chúng ta chỉ cần cung cấp đối tượng nguồn và đối tượng đích cụ thể
* Với Implicit Intent : Android phải tim Application Component phù hợp để thực hiện Action chuyển phát thông điệp (Broadcast)
* Để làm được điều này Android sử dụng Intent Filter như những bộ lọc để cho biết một Application Component có thể thực hiện được những Action nào
* Ví dụ: Như khi bạn mở một link web, android sẽ đưa ra một thông báo với các ứng dụng có thể mở đươc link này như Google, Chorme, Brower,...Vậy các ứng dụn đó đã được cài Intent Filter để thông báo cho Android rằng “Tôi có thể thực hiện được việc này”

//Khai báo intent  
Intent myintent = new Intent(Intetn.ACTION\_VIEW, Uri.parse(https://+edtlink.getText().toString()));  
//Khởi động Intent  
startActivity(myintent);

1. Intent và Service

* Service dùng để thực hiện những công việc chạy nền, mang tính chất lâu dài và không cần cung cấp giao diện cho người dùng tương tác, có thể làm việc ngay cả khi ứng dụng bị hủy
* Cần phải có 1 Component khác để khởi động và ngừng service (Intent)
* Sử dụng các thông điệp bất đồng bộ Intent để start và stop một service
* Service có 2 loại Unbounded service và Bounded service

1. Broadcast Receiver

* Broadcast Receicer là bộ tiếp nhận và xử lý các Intent dựa trên các chỉ định của chúng ta hoặc hệ thống
* Thành phần này sẽ có phép mở các ứng dụng để xử lý các Intent tương ứng được gửi lên từ hệ thống
* Trong các ứng dụng, đôi khi bạn cần phải nhận được tín hiệ nếu như một sự kiện nào đó trong thiết bị được kích hoạt để xử lý cho các ứng dụng tốt hơn
* Các sự kiện ví dụ như: Điện thoại sắp hết pin, nguồn được cắm sạc, rút sạc, bộ nhớ sắp hết. Thay đổi vị trí, ngày giờ, máy vừa được khởi động, một ứng dụng nào đó vừa cài đặt lên máy, Internet được thay đổi,... khi đó, sử dụng BroadcastReceiver sẽ gips chúng ta bắt được sự kiện này khi chúng bị kích hoạt
* Tạo Broadcast Receiver

+ Để tạo một Broadcast Receiver, ta sẽ tạo một lớp kế thừa từ Broadcasr Receiver

Public class MySmsReceive extends BroadcastReceiver{

@Override

Public void onReceive (Context context, Intent intent){

//To do thing

}

+ **Context:** là context của ứng dụng hoặc hệ thống mà Broadcast này được kích hoạt

+ **Intent:** Toàn bộ dữ liệu nhận được đều nằm trong đối số này. Tùy thuộc vào bộ lọc khi đăng ký mà cách lấy đối số này sẽ khác nhau

* Đăng ký Broadcast Receiver

+ Đăng ký trong onResume của Activity: Khi tắt Activity thì Receiver không còn hoạt động nữa

IntentFilter filter = new IntentFilter(“Tên bộ lọc khai báo ở đây);

MySmsReceive receive = new MySmsReceive();

registerReceiver(receiver, filter);

+ Đăng ký trong AndroidManifest: Trở thành 1 dịch vụ chạy ngầm, cho dù tắt phần mềm nó vẫn luôn lắng nghe được bộ lọc

<receiver

android:name = “.MySmsReceive”>

<intent-filter>

<action android:name = “Tên bộ lọc khai báo tại đây” />

</intent-filter>

</receiver>

* Hủy Broadcast Receiver

+ Muốn hủy BroadcastReceiver ta dùng lệnh unregisterReceiver trong onPause

@Override

Protected void onPause(){

super.onPause();

unregisterReceiver(MySmsReceive);

}

+ Chú ý: Lệnh hủy chỉ áp dụng cho BroadcastReceiver được đăng ký trong Activity. Nếu đăng ký trong AndroidManifest thì khi tháo ứng dụng khỏi thiết bị thì Receiver mới bị hủy

* Tên một số bộ lọc Receiver

1. Android.provider.Telephony.SMS\_RECEIVE
2. Android.intent.action.PHONE\_STATE
3. Android.net.conn.CONNECTIVITY\_CHANGE
4. Android.intent.action.ACTION\_POWER\_CONNECTED
5. Android.intent.action.ACTION\_POWER\_DISCONNECTED
6. Android.intent.action.BATTERY\_CHANGED
7. Android.intent.action.BATTERY\_LOW
8. Android.intent.action.DEVICE\_STORAGE\_LOW
9. Android.intent.action.CAMERA\_BUTTON
10. Android.intent.action.DATE\_CHANGED
11. Android.intent.action.SCREEN\_ON/OFF
12. ListView

B1. Khởi tạo dữ liệu cho mảng arr (còn gọi là data source)

String arr[] = {“name1”, “name2”, “name3”, …};

B2. Khai báo Array Adapter và gán data source vào ArrayAdapter

ArrayAdapter<String> adapter = new ArrayAdapter<>(MainAcivity.this, android.R.layout.simple.list\_item\_1, arr) ;

B3. Khai báo ListView và đưa Adapter vào ListView

ListView lv.

lv.setAdapter(adapter) ;

1. Tab

* Tab Host : Là Container chính chứa các Tab buttons và Tab contents
* Tab Widget : Để định dạng cho các Tab buttons : Nhãn, Icon,…
* FrameLayout : Là Container để chứa các layout cho Tab contents, ta chỉ có thể dùng FrameLayout cho Tab contents, không thể dùng các loại layout khác. Tại vì sao lại là FrameLayout mà không phải là các Layout khác ? Cho dù bạn có nhấn vào Tab nào đi nữa thì layout tương ứng với mỗi Tab mà bạn vừa nhấn cũng chỉ xuất hienj cùng một chỗ trên màn hình điện thoại, điều này chỉ có FrameLayout mới giải quyết được

1. Cách lưu trữ dữ liệu trong Android

* Trong Android có nhiều các để lưu trữ dữ liệu với nhiều mục đích khác nhau :

+ Sử dụng SQLite để lưu trức các data dưới dạng table hay nói các khách chính là database

+ Lưu trữ file ở Internal Storage, External Stroeage.

+ Lưu trữ dữ liệu trên server và thông qua webservice để thao tác với dữ liệu đó

+ Lưu trạng thái ứng dụng với Shared Preferences

* SharedPreferences là một cách lưu trữ những dữ liệu dạng nguyên thủy (bool, int, float, double, String) trong android dưới dạng file xml với những cặp key-value

+ Ví dụ như lưu trữ high score, level, status của game,…

* Trong sự kiện onPause() hoặc các sự kiện khác lưu lại thông tin trạng thái trong đối tượng SharedPreferences, và trong sự kiện onCreate(Bundle), ta sẽ đọc các thông tin lưu trữ và hiển thị trên giao diện
* Lưu trữ và lấy dữ liệu với SharedPreferences
* Lưu trữ

+ Để làm việc với SharedPreferences thì Android SDK có cung cấp cho chúng ta hai class chính đó là SharedPrefences và SharedPrefences.Editor

+ Tạo và lưu trữ dữ liệu trong onPause();

B1: Tạo đối tượng SharedPrefences

SharedPrefences prefences = get SharedPrefences(“Tên file”, MODE\_PRIVATE);

->Đối số tên file là một String, chỉ cần nhập tên file, không cần nhập đuôi .xml. Vì mặc định sẽ có đuôi là .xml

->File .xml được lưu trong bộ nhớ có đường dẫn như sau

DATA/data/app name/shared\_prefs/[tên file].xml

B2: Muốn lưu trữ ta phải gọi đến đối tượng chỉnh sửa Editor từ đối tượng SharedPrefences ở trên:

SharedPrefences.Editor editor = prefences.edit();

B3: Thêm dữ liệu: editor.putxxx(String key, value);

B4: Xác nhận lưu trữ (Bắt buộc phải có): editor.commit();

+ Những phương thức có sẵn trong lớp editor, cho phép thao tác dữ liệu trong Shared Prefences:

+ editor.clear() ->Xóa tất cả dữ liệu

+ editor.remove(String key) ->Xóa dữ liệu tương ứng với key

+ editor.putString(“key”, “value”);

+ editor.putLong(String key, long value);

+ editor.putInt(String key, int value);

+ editor.putFloat(String key, float value);

+ editor.putBoolean(String key true);

* Nhận data từ SharedPrefences trong onCreate()
* Để nhận dữ liệu từ SharedPreferences chúng ta không cần tạo đối tượng Editor mà có thể sử dụng ngay chính đối tượng sharedPrefernces với các phương thức là getxxx(String key, defaultValue)
* Với xxx là Boolean, Int, Float, Long, String. Phương thức trả về đúng giá trị ứng với key trong sharedPreferences. Nếu không có key sẽ trả về giá trị defaultValue

SharedPrefences sharedPreferences = getSharedPreferences(“tên file”, MORE\_PRIVATE);

Ví dụ:

Int score = sharedPreferences.getInt(“high score”, 0);