3.6. QUY TRÌNH THIẾT KẾ ỨNG DỤNG TRONG ANDROID

Trong bài học trước. Chúng ta đã được:

- Giới thiệu về học phần, các keyword để định hướng việc học tập và nghiên cứu, các qui tắc coding
- Tổng quan về lập trình android: Cài đặt môi trường lập trình, mà hình thiết kế giao diện, cấu trúc thư mục trong dự án Android, qui trình thiết kế ứng dụng, khai thác sử dụng các tài nguyên đã định nghĩa, thiết kế giao diện (ViewGroup, View)
- Các nội dung trên đã được thể hiện Demo qua ứng dụng cụ thể. Qua đó ta thấy việc qui trình thiết kế ứng dụng được chúng ta thực hiện theo 3 bước:
    1. Thiết kế giao diện
    2. Khai báo và ánh xạ view
    3. Sử lý logic, sự kiện, nghiệp vụ

3.8. LISTVIEW, ALERTDIALOG

NỘI DUNG BÀI HỌC

- List View
- DatePickerDialog/TimePickerDialog
- AlertDialog
- Toast
- Demo

3.8.1. LIST VIEW

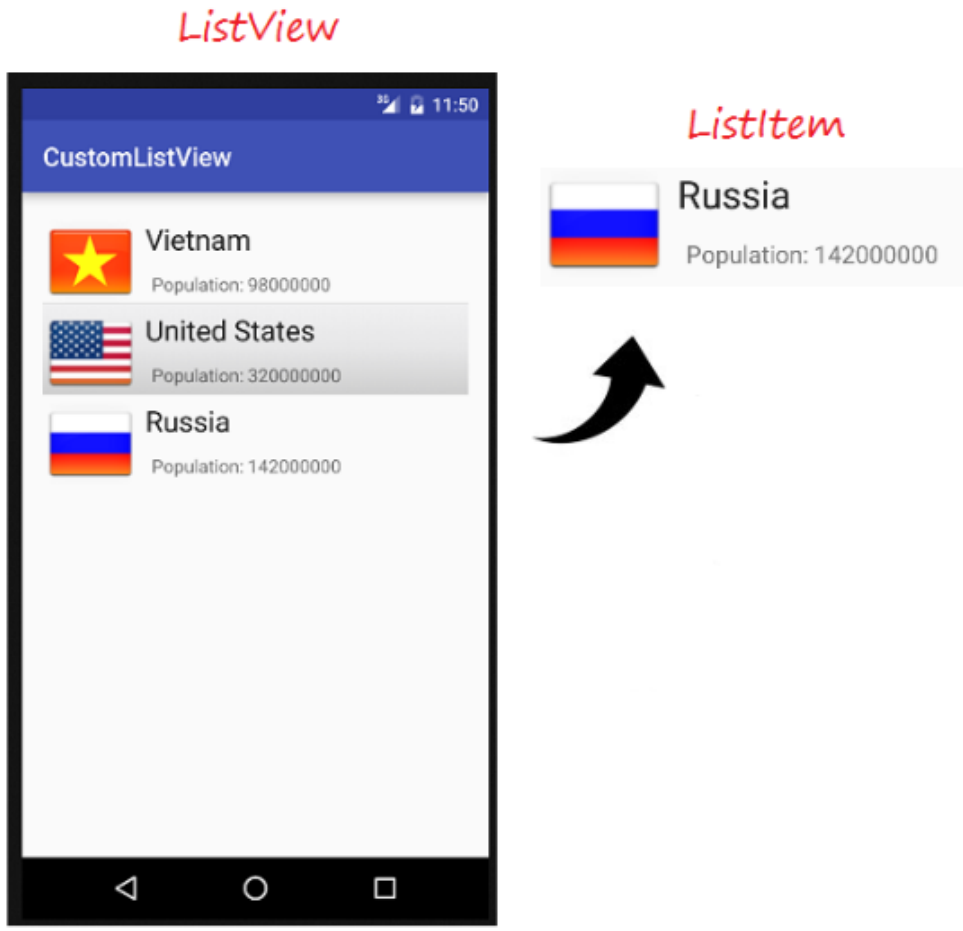
ListView là một thành phần giao diện dùng để hiển thị danh sách các mục có thể cuộn theo chiều dọc. Nó rất phổ biến trong các ứng dụng Android để trình bày dữ liệu như danh bạ, danh sách sản phẩm, tin nhắn, v.v.

3.8.1.1. Cấu trúc cơ bản của ListView

•ListView: View chính hiển thị danh sách.

•ListItem: Mỗi dòng trong ListView, thường là một layout riêng.

•Adapter: Cầu nối giữa dữ liệu và giao diện hiển thị.



3.8.1. LIST VIEW

3.8.1.2. Các bước sử dụng ListView

1.Khai báo ListView trong XML

```
<ListView
    android:id="@+id/list_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

2.Chuẩn bị dữ liệu

```
String[ ] items = {"Item 1", "Item 2", "Item 3"};
```

3.Tạo Adapter

•Với dữ liệu đơn giản, dùng ArrayAdapter:

```
ArrayAdapter<String> adapter =
 new ArrayAdapter<>(this,   android.R.layout.simple_list_item_1, items);
listView.setAdapter(adapter);
```

3.8.1. LIST VIEW

3.Tạo Adapter. Với dữ liệu đơn giản, dùng ArrayAdapter:

```
ArrayAdapter<String> adapter =
    new ArrayAdapter<>(this,   android.R.layout.simple_list_item_1, items);
listView.setAdapter(adapter);
```

Khi thay đổi dữ liệu (thêm, xóa, sửa một mục trong ArrayList), ListView không tự động biết điều đó. Bạn phải "thông báo" cho Adapter rằng dữ liệu đã thay đổi.

Khi gọi adapter.notifyDataSetChanged(), Adapter sẽ:Nhận biết rằng dữ liệu nguồn đã cũ.Báo cho ListView phải làm mới lại toàn bộ danh sách.ListView sẽ yêu cầu lại các View từ Adapter để hiển thị dữ liệu mới nhất.

3.8.1. LIST VIEW

3.8.1.4. Ví dụ hiển thị ds sinh viên:

activity_main.xml

```
<ListView
        android:id="@+id/lv_students"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

Student.java

```
public class Student {
    private String name;
    Boolean sex;
    private String dateBirth;
    private String hobbie;
```

MainActivity.java

```
lstStudents = new ArrayList<Student>();
        lstStudents.add(new Student("aa", true, "01/01/2020",
"Codding"));


        arrayAdapter = new
ArrayAdapter<Student>(MainActivity.this,
android.R.layout.simple_list_item_1, lstStudents);
        lvStudents.setAdapter(arrayAdapter);

        //Xu ly su kien tren ListView
        lvStudents.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?>
adapterView, View view, int i, long l) {
                selectedStuddent(lstStudents.get(i));
                position = i;
            }
        });
```

3.8.2. DATEPICKERDIALOG/TIMEPICKERDIALOG

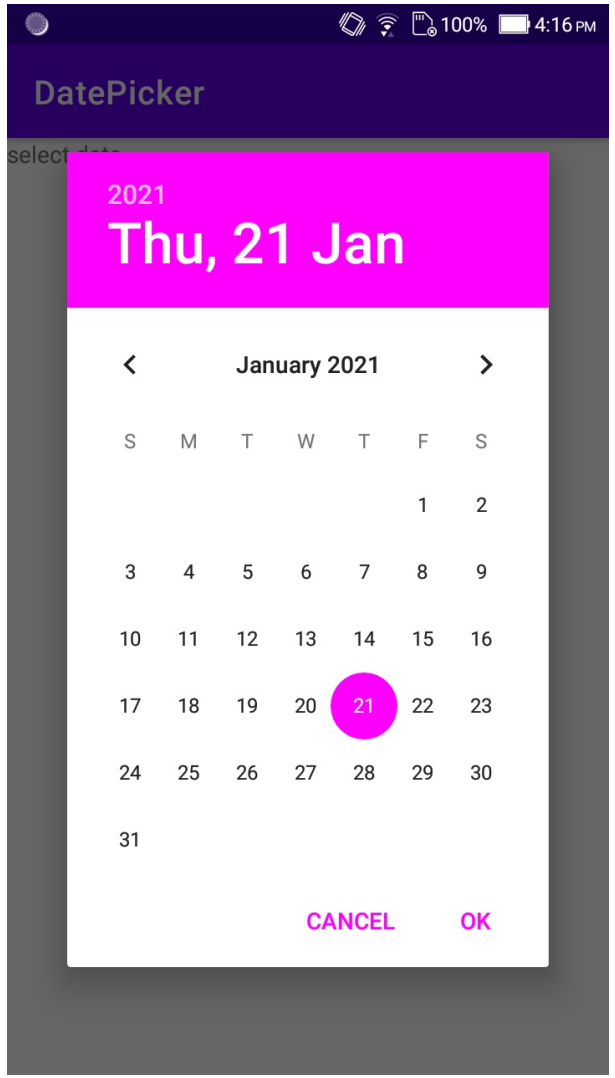
DatePickerDialog là một hộp thoại cho phép người dùng chọn ngày (ngày, tháng, năm) một cách trực quan. Nó thường được sử dụng khi bạn muốn người dùng nhập ngày mà không cần gõ tay.

3.8.2.1. Cách hoạt động

•DatePickerDialog là một lớp mở rộng từ AlertDialog, chứa một DatePicker.

•Khi được gọi, nó hiển thị một giao diện chọn ngày.

•Sau khi người dùng chọn xong, kết quả được trả về qua OnDateSetListener.



3.8.2. DATEPICKERDIALOG/TIMEPICKERDIALOG

3.8.2.2. Ví dụ.

1. *Trong onCreate của MainActivity.java:*

```
btnSelectedDate.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        showDatePickerDialog();
    }
});
```

2. *Viết phương thức showDatePickerDialog trong MainActivity.java*

3.8.2. DATEPICKERDIALOG/TIMEPICKERDIALOG

```
public void selectDay(EditText editText){
        //Lay ngay thang nam hien tai
        final Calendar calendar=Calendar.getInstance();
        int year=calendar.get(Calendar.YEAR);
        int month=calendar.get(Calendar.MONTH);
        int day=calendar.get(Calendar.DAY_OF_MONTH);
        //Tao DatePickerDialog
        DatePickerDialog datePickerDialog= new DatePickerDialog(MainActivity.this, new
DatePickerDialog.OnDateSetListener() {
            @Override
            public void onDateSet(DatePicker datePicker, int y, int m, int d) {
                //Cap nhat textView sau khi nguoi dung da hon
                String selectedDay=d+"/"+(m+1)+"/"+y;
                editText.setText(selectedDay);
            }
        },year,month,day);
        // Hien thi DatePickerDiaglog
        datePickerDialog.show();
    }
```

3.8.3. ALERTDIALOG

AlertDialog là một thành phần giao diện trong Android dùng để hiển thị hộp thoại cảnh báo hoặc xác nhận với người dùng. Nó thường được sử dụng để hỏi người dùng một câu hỏi, hiển thị thông báo quan trọng, hoặc yêu cầu xác nhận hành động.

3.8.3.1. Cấu trúc cơ bản của AlertDialog

AlertDialog gồm 3 phần chính:

- Tiêu đề (Title): Mô tả ngắn gọn nội dung cảnh báo.
- Nội dung (Message): Thông điệp chi tiết gửi đến người dùng.
- Các nút hành động (Buttons): Có thể là nút Positive, Negative, và Neutral.

3.8.3. ALERTDIALOG

3.8.3.2. Cách tạo AlertDialog đơn giản

```
AlertDialog.Builder builder = new AlertDialog.Builder(context);
builder.setTitle("Xác nhận");
builder.setMessage("Bạn có chắc muốn xóa mục này?");
builder.setPositiveButton("Xóa", new DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {
        // Xử lý khi người dùng chọn "Xóa"
    }
});
builder.setNegativeButton("Hủy", null);
builder.show();
```

3.8.4. TOAST

Toast là một cách đơn giản để hiển thị thông báo ngắn gọn cho người dùng. Nó xuất hiện như một popup nhỏ ở cuối màn hình và tự động biến mất sau một khoảng thời gian ngắn.

3.8.4.1. Đặc điểm nổi bật

•Không yêu cầu người dùng tương tác.

•Không làm gián đoạn hoạt động của ứng dụng.

•Thường dùng để hiển thị thông báo trạng thái, xác nhận hành động, hoặc lỗi nhẹ.

3.8.4.2. Cách sử dụng Toast

```
Toast.makeText(context, "Đây là thông báo Toast", Toast.LENGTH_SHORT).show();
```

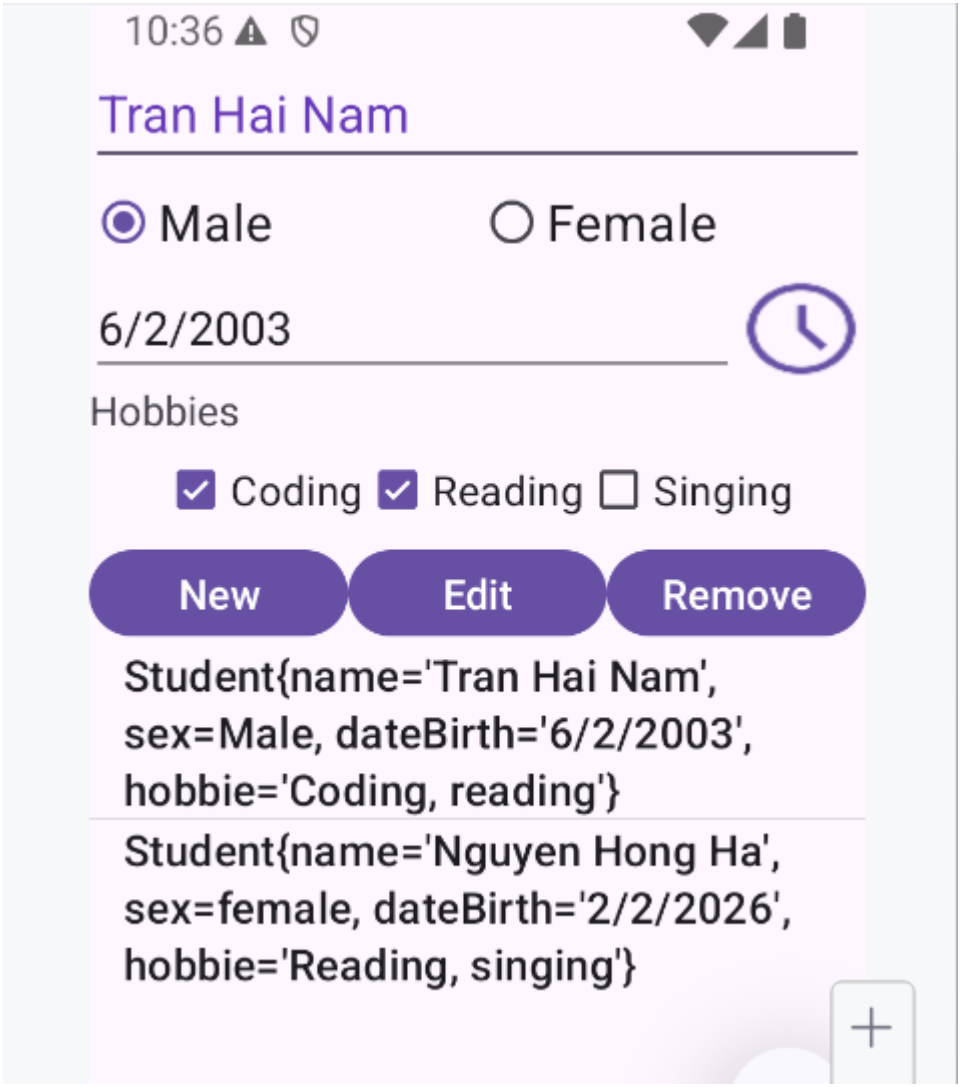
•context: ngữ cảnh hiện tại (thường là this hoặc getApplicationContext()).

•"Đây là thông báo Toast": nội dung hiển thị.

•Toast.LENGTH_SHORT hoặc Toast.LENGTH_LONG: thời gian hiển thị (2 hoặc 3.5 giây).

3.8.5. DEMO:

Sử dụng
DatePickerDialg,
AlertDialog,
Toast,
ListView…
Để xây dựng ứng dung có mô tả như sau:



BÀI 03. MENU VÀ INTENT

Trong bài học trước chúng ta đã tìm hiểu về:

- ListView
- AlertDialog:
  - ✓DatePickerDialog/TimePickerDialog
  - ✓AlertDialog.Buider
  - ✓Toast

Thực hành thông qua bài kiểm tra thường xuyên 1.

Mốt số các vấn đề cần quan tâm nhiều hơn:

- Tính trải nghiệm người dung: lập trình nút điều khiển, Kiểm tra dữ liệu…
- Giao diện người dùng: Bố cuch, hình ảnh, màu sắc
- Máy tính: Cấu hình, pin, phần mềm…

NỘI DUNG

Menu

- OptionMenu
- ContextMenu

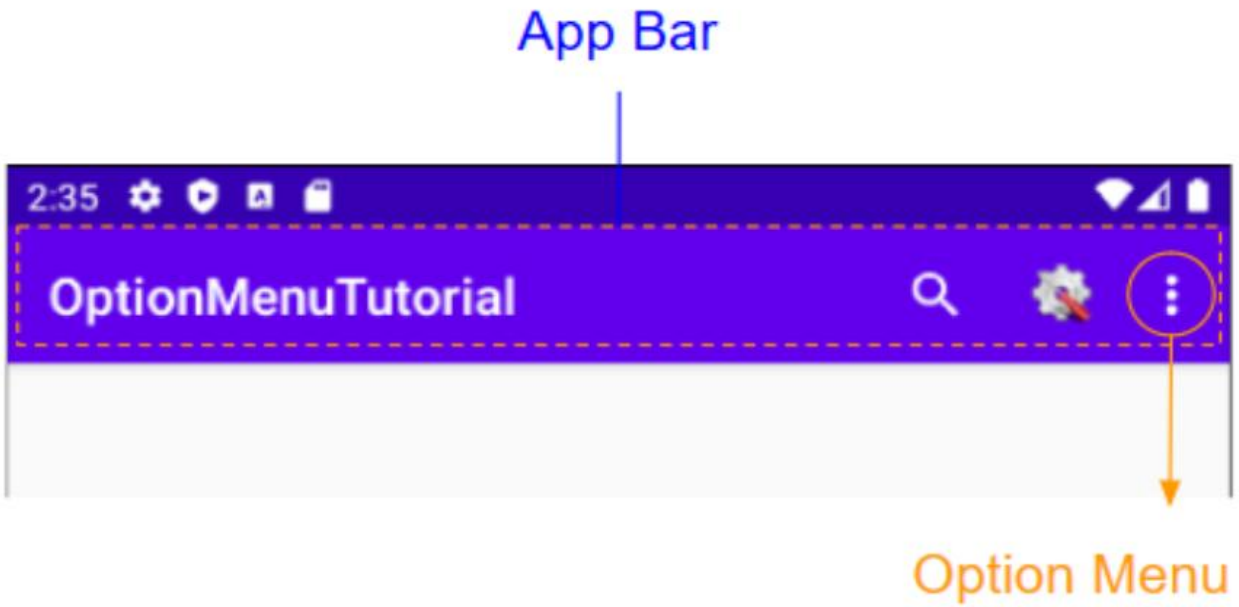
Intent

- Explicit intent (Primary data, Object, Bundle, ActivityResultLaunch…)
- Implicit intent (ACTION_VIEW, ACTION_SENDTO, ACTION_CALL,…)

3.7. MENU

3.7.1. Option Menu

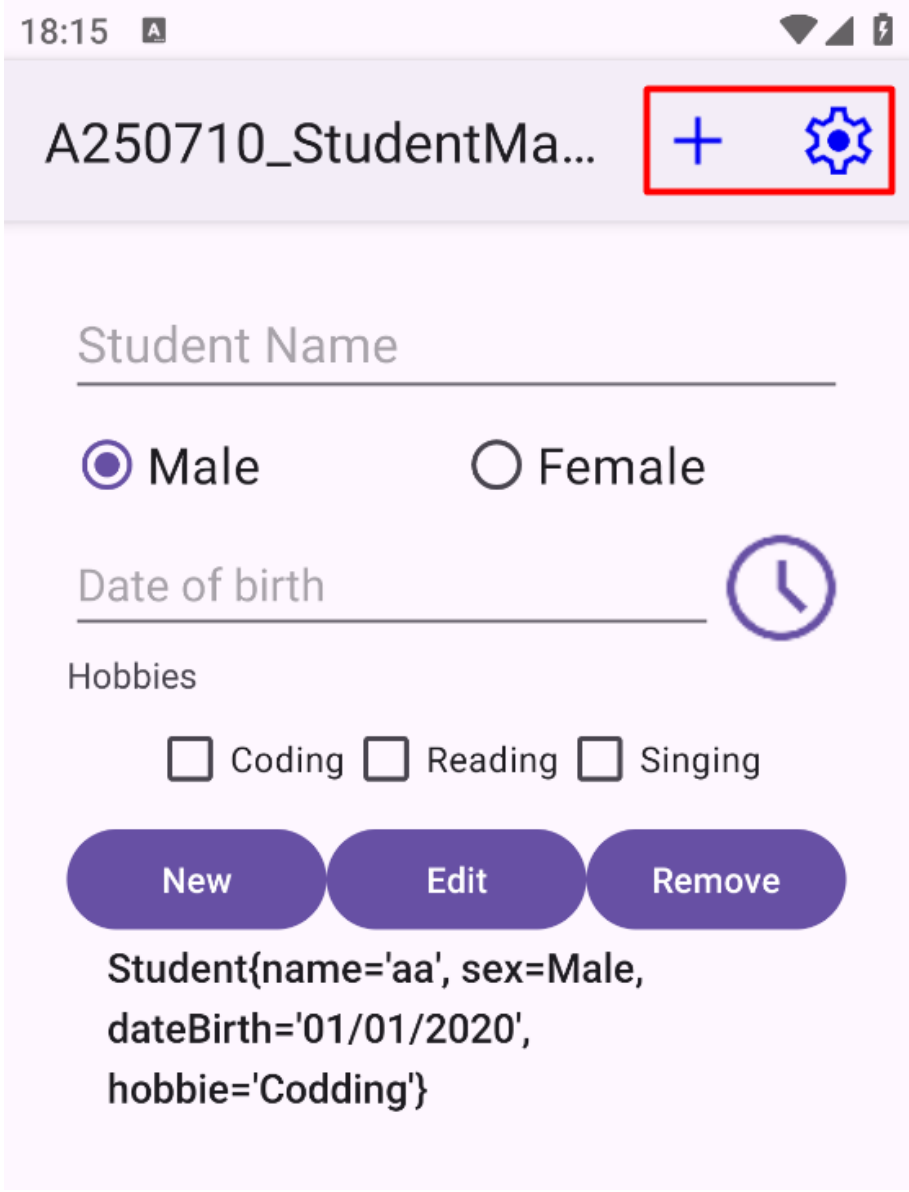
Trong Android một Option Menu là một tập hợp của các tùy chọn (option) chính cho một ứng dụng, người dùng có thể lựa chọn một trong các tùy chọn để thực hiện một hành động. Option Menu xuất hiện trên App Bar (Thanh ứng dụng), phía bên phải.



3.7.1. OPTION MENU

Ví dụ về Option Menu

- Hiện action bar: Vào thư mục res/values/themes để cấu hình lại Actionbar (Bỏ NoActionBar)
- Thiết kế Option menu
- Gọi Option menu trong MainActivity.java
- Bắt sự kiện cho các item menu

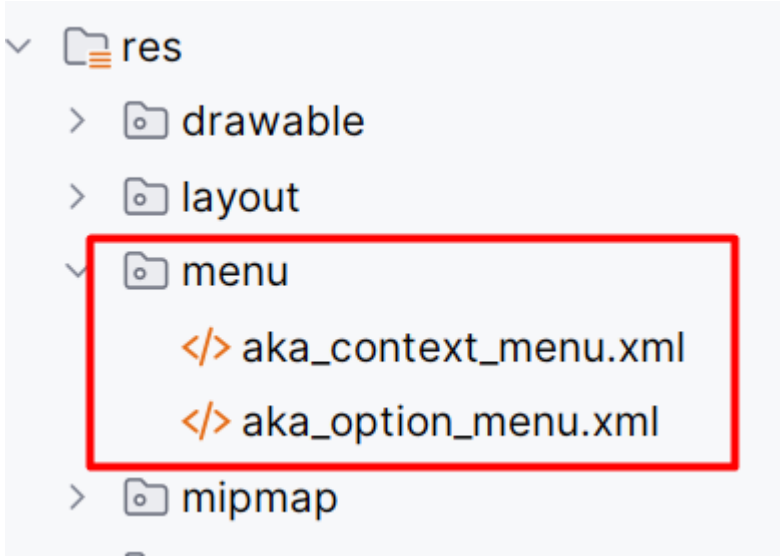


3.7.1. OPTION MENU

Ví dụ về cách sử dụng OptionMenu trong Android để hiển thị các tùy chọn trên thanh công cụ (App Bar):

Bước 1. Tạo file menu XML: Trong thư mục res/menu, tạo file main_menu.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
<item
    android:id="@+id/menu_add"
    android:title="@string/menu_add"
    android:icon="@drawable/add_50dp"
    app:showAsAction="ifRoom"  />
    <item
        android:id="@+id/menu_settings"
        android:title="@string/menu_settings"
        android:icon="@drawable/settings_50dp"
        app:showAsAction="ifRoom"
        />
</menu>
```



3.7.1. OPTION MENU

Bước 2: Gọi menu trong MainActivity.java

```
//Gọi OptionMenu
 @Override
 public boolean onCreateOptionsMenu(Menu menu) {
     getMenuInflater().inflate(R.menu.aka_option_menu, menu);
     return super.onCreateOptionsMenu(menu);
 }
 @Override
 public boolean onOptionsItemSelected(@NonNull MenuItem item) {
     if (item.getItemId() == R.id.menu_add) {
         Toast.makeText(this, "Ban da chon them", Toast.LENGTH_SHORT).show();
     }
     if (item.getItemId() == R.id.menu_settings) {
         Toast.makeText(this, "Ban da chon nut cai dat", Toast.LENGTH_SHORT).show();
     }
     return super.onOptionsItemSelected(item);
}
```

Ghi chú

- showAsAction="ifRoom": Hiển thị icon trên App Bar nếu còn chỗ trống.
- Cần có các icon như ic_add.png, ic_settings.png trong thư mục res/drawable.

3.7.2. Context Menu

- ContextMenu là menu ngữ cảnh xuất hiện khi người dùng nhấn giữ (long press - mặc định là 500 mili giây) vào một View, thường dùng để cung cấp các hành động liên quan đến đối tượng đó.
- Context Menu có thể chứa nhiều Menu Item và Sub Menu.
- Khi một View được đăng ký với sự kiện Long-Press, hệ thống sẽ gọi phương thức onCreateContextMenu() để tạo và hiển thị Context Menu.

3.7.2. CONTEXT MENU

1. Tạo file menu XML: Tạo file aka_context_menu.xml trong thư mục res/menu:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
<item
    android:id="@+id/menu_remove"
    android:title="@string/btn_remove"/>
    <item
      android:id="@+id/menu_share"
       android:title="@string/menu_share">
       <menu>
          <item
             android:id="@+id/menu_facebook"
             android:title="facebook"/>
          <item
             android:id="@+id/menu_zalo"
             android:title="zalo"/>
       </menu>
    </item>
</menu>
```

3.7.2. CONTEXT MENU

2. Xử lý trong MainActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    …
    mapping();
    //Dang ky ContextMenu cho ListView
    registerForContextMenu(lvStudents);
…}
```

3.7.2. CONTEXT MENU

2. Xử lý trong MainActivity.java

```
//Sau khi dang ky ConTextMenu cho ListView, viet cac phuong thuc conCreateContextMenu va OnContextItemSelected
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
        getMenuInflater().inflate(R.menu.aka_context_menu, menu);
        menu.setHeaderTitle("Select action");
}
@Override
public boolean onContextItemSelected(@NonNull MenuItem item) {
    if(item.getItemId()==R.id.menu_remove){
        Toast.makeText(this, "Xoa ban ghi nay", Toast.LENGTH_SHORT).show();
    }
    if(item.getItemId()==R.id.menu_facebook){
        Toast.makeText(this, "Chia se tren Facbook", Toast.LENGTH_SHORT).show();
    }
    if(item.getItemId()==R.id.menu_zalo){
        Toast.makeText(this, "Chia se tren zalo", Toast.LENGTH_SHORT).show();
    }
    return super.onContextItemSelected(item);
}
```

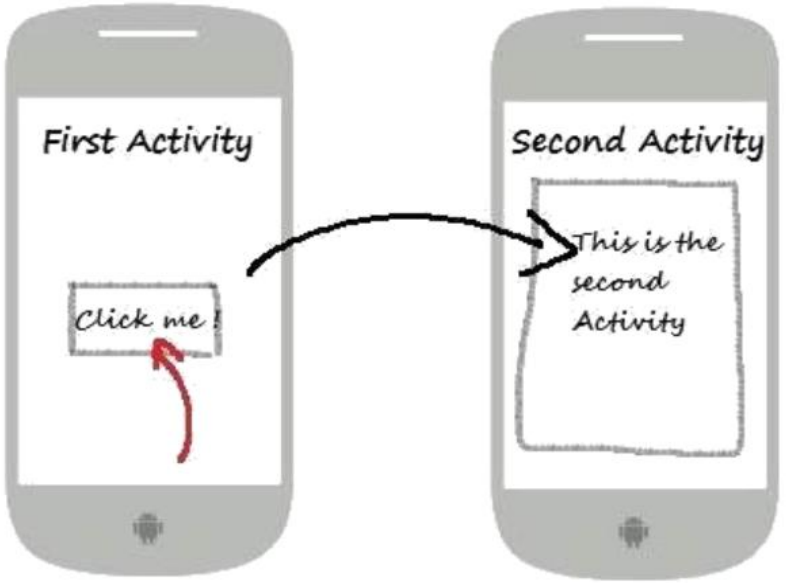
3.7.2. CONTEXT MENU

📌 Ghi chú

- registerForContextMenu(view) là bắt buộc để kích hoạt ContextMenu cho View đó.
- ContextMenu thường dùng cho ListView, Button, hoặc TextView.
- Từ Android 3.0 trở lên, icon trong ContextMenu không được hỗ trợ.

PHẦN 4. INTENT

Intent là một đối tượng dùng để yêu cầu thực hiện một hành động từ một thành phần khác trong hệ thống, như mở một Activity, khởi động Service, hoặc gửi Broadcast. Nó là cách để các thành phần trong ứng dụng giao tiếp với nhau hoặc với các ứng dụng khác.



PHẦN 4. INTENT

4.1. Vai trò của Intent

- Kết nối các thành phần: Giúp Activity, Service, BroadcastReceiver tương tác với nhau.
- Truyền dữ liệu: Gửi thông tin từ nơi này sang nơi khác.
- Kích hoạt hành động: Ví dụ mở trình duyệt, gọi điện, chụp ảnh...

4.2. Các loại Intent

Loại Intent	Mô tả
Explicit Intent	Chỉ rõ thành phần đích (ví dụ: mở một Activity cụ thể trong app)
Implicit Intent	Không chỉ rõ đích, hệ thống sẽ tìm ứng dụng phù hợp để xử lý yêu cầu

PHẦN 4. INTENT

4.2.3. Ví dụ sử dụng Intent

Explicit Intent – mở Activity khác:

```
Intent intent = new Intent(this, SecondActivity.class);
intent.putExtra("username", “CongDH");
startActivity(intent);
```

Implicit Intent – mở trình duyệt:

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("https://www.microsoft.com"));
```

PHẦN 4. INTENT

4.2.3.1. Truyền dữ liệu qua Intent

- Dùng putExtra() để gửi dữ liệu: intent.putExtra("key", "value");
- Dùng getIntent().getStringExtra("key") để nhận dữ liệu ở Activity đích.

4.2.3.2. Nhận kết quả từ Activity khác

- Gọi startActivityForResult(intent, REQUEST_CODE)
- Nhận kết quả trong onActivityResult()

4.1. INTENT TƯỜNG MINH

4.1.4. Intent với dữ liệu là Object

Lưu ý đối tượng Product phải cài đặt **Serializable**

```
public class Product implements Serializable {
    private int id;
    private String name;
    private
```

MainActivity.java

```
myintent= new Intent(MainActivity.this,ObjIntentSub.class);
Product obj =new Product();
obj.setId(Integer.parseInt(edtID.getText().toString()));
obj.setName(edtName.getText().toString());
obj.setPrice(Float.parseFloat(edtPrice.getText().toString()));
myintent.putExtra("productObj",obj);
startActivity(myintent);
```

SubActivity.java

```
intentObj = getIntent();
Product obj = new Product();
obj = (Product) intentObj.getSerializableExtra("productObj");
txtIdObj.setText("" + obj.getId());
txtNameObj.setText(obj.getName());
txtPriceObj.setText("" + obj.getPrice());
```

4.1. INTENT TƯỜNG MINH

4.1.5. Sử dụng bundle để chuyển dữ liệu

MainActivity.java

```
      //demo voi intent su dung Bundle
btnBundle.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      bundle=new Bundle();
      bundle.putInt("id",Integer.parseInt(edtID.getText().toString()));
      bundle.putString("name",edtName.getText().toString());
      bundle.putFloat("price",Float.parseFloat(edtPrice.getText().toString()));

      myintent=new Intent(MainActivity.this, BundleIntentSubActivity.class);
      myintent.putExtra("myBundle",bundle);
      startActivity(myintent);
   }
});
```

BundleIntentSubActivity.java

```
…
intentBundle=getIntent();
myBundle=intentBundle.getBundleExtra("myBundle");
txtIdBundle.setText(""+myBundle.getInt("id"));
txtNameBundle.setText(myBundle.getString("name"));
txtPriceBundle.setText(""+myBundle.getFloat("price"));

btnOkBundle.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      finish();
   }
});
```

4.2.4. VÍ DỤ VỀ EXPLICIT INTENT

Sử dụng Explicit Intent để chuyển từ MainActivity sang SecondActivity và truyền dữ liệu giữa hai màn hình.

4.2.4.1. Tạo giao diện activity_main.xml

```
<Button
    android:id="@+id/btn_go"
    android:text="Đi đến màn hình kế tiếp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

4.2.4.2. MainActivity.java

4.2.4. VÍ DỤ VỀ EXPLICIT INTENT

4.2.4.2. MainActivity.java

```
btnGo.setOnClickListener(new View.OnClickListener() {
        public void onClick(View view) {
                sendData();
        } });
// Chuyen du lieu di
    private void sendData(){
        intent=new Intent(MainActivity.this,SecondActivity.class);
        intent.putExtra("msg","Loi chao tu MainActivity");
        startActivity(intent);
    }
```

4.2.4. VÍ DỤ VỀ EXPLICIT INTENT

4.2.4.3. Giao diện activity_second.xml

```
<TextView
        android:id="@+id/tv_msg"
        android:text="Chua co du lieu"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
```

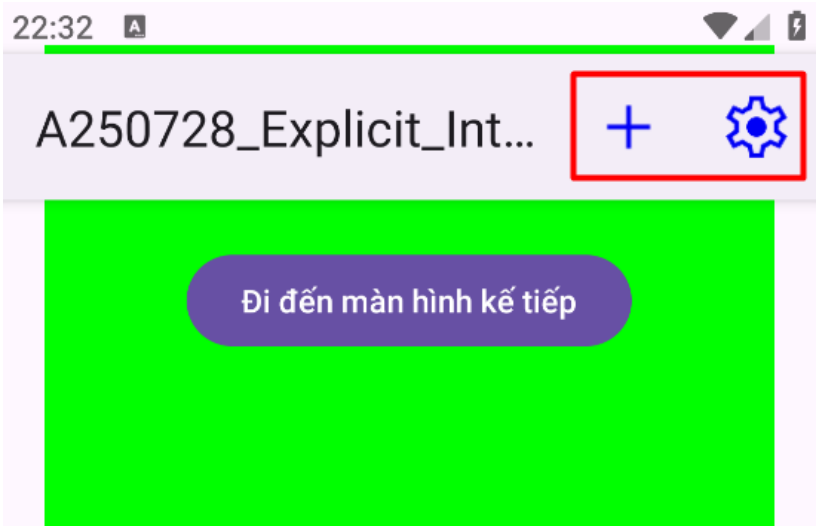
4.2.4.4. SecondActivity.java

```
 mapping();
        String msg=getIntent().getStringExtra("msg");
        tvMsg.setText(msg);
```

4.2.5. ACTIVITYRESULTLAUNCHER

Ta đã có một ví dụ hoàn chỉnh về Explicit Intent để chuyển màn hình và truyền dữ liệu.

Để nhận kết quả từ SecondActivity, có thể sử dụng ActivityResultLauncher để thay thế cho startActivityForResult() đã bị deprecated.



Ví dụ: Khi nhấn vào menu Settings, hiện activity_settings cho phép chọn màu nền cho activity_main

4.2.5.1. MAINACTIVITY.JAVA – ĐĂNG KÝ VÀ SỬ DỤNG ACTIVITYRESULTLAUNCHER

Trong MainActivity.java – Đăng ký và sử dụng ActivityResultLauncher

//Khai báo đối tượng ActivityResultLauncher;

```
private ActivityResultLauncher<Intent> launcher;
```

Trong onCreate() method định nghĩa đối tượng laucher;

```
launcher = registerForActivityResult(
        new ActivityResultContracts.StartActivityForResult(), activityResult -> {
            if (activityResult.getResultCode() == RESULT_OK && activityResult.getData() != null) {
                Toast.makeText(this, "Nhan ket qua tra ve", Toast.LENGTH_SHORT).show();
                int selectedColor = activityResult.getData().getIntExtra("selectedColor", Color.GRAY);
                linearLayout.setBackgroundColor(selectedColor);

            }
        }
);
```

4.2.5.1. MAINACTIVITY.JAVA – ĐĂNG KÝ VÀ SỬ DỤNG ACTIVITYRESULTLAUNCHER

- Trong menu Settings - onOptionsItemSelected(), gọi activity_settings

```
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.menu_add) {
        Toast.makeText(this, "Bạn đã chọn Add", Toast.LENGTH_SHORT).show();
    }
    if (item.getItemId() == R.id.menu_settings) {
        Intent intent2 = new Intent(MainActivity.this, SettingsActivity.class);
        intent2.putExtra("msgSettings", "Chon background");
        launcher.launch(intent2);
}
    return super.onOptionsItemSelected(item);
}
```

4.2.5.2. SETTINGSACTIVITY.JAVA – TRẢ KẾT QUẢ VỀ

Trong SettingActivity.java

- Trong onCreate() method của SettingActivity.java, nhận giá trị gửi đến từ MainActivity.java

```
mapping();
String msgSettings = getIntent().getStringExtra("msgSettings");
Toast.makeText(this, msgSettings, Toast.LENGTH_SHORT).show();
```

- Trong định nghĩa sự kiện của nút btnOk

4.2.5.2. SETTINGSACTIVITY.JAVA – TRẢ KẾT QUẢ VỀ

```
btnOk.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
        int selectedColor = Color.YELLOW;
        if (rgColor.getCheckedRadioButtonId() == R.id.rd_blue) {
            selectedColor = Color.BLUE;
        }
        if (rgColor.getCheckedRadioButtonId() == R.id.rd_green) {
            selectedColor = Color.GREEN;
        }
        if (rgColor.getCheckedRadioButtonId() == R.id.rd_red) {
            selectedColor = Color.RED;
        }
        Intent resultIntent = new Intent();
        resultIntent.putExtra("selectedColor", selectedColor);
        setResult(RESULT_OK, resultIntent);
        finish();//Tra ket qua va ket thuc activity_settings
    }
});
```

4.2.7. 📌GHI CHÚ

•ActivityResultLauncher giúp mã ngắn gọn, dễ bảo trì hơn so với startActivityForResult.

•Bạn có thể dùng nhiều launcher khác nhau cho các mục đích như chọn ảnh, xin quyền truy cập, v.v.