Android skeleton và hoạt động(tt)

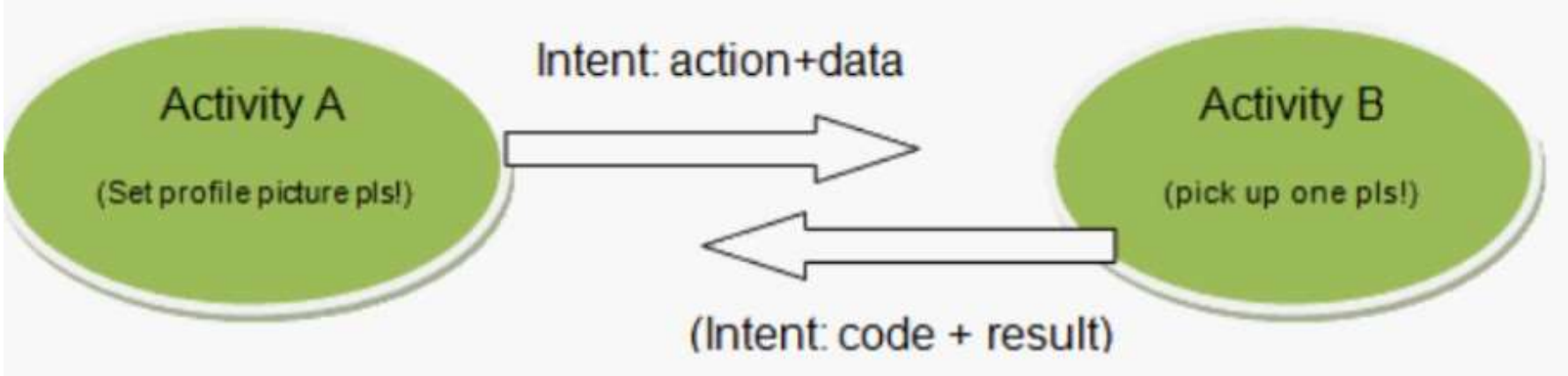
ThS.Trần Hồng Vinh

Nội dung bài học

- Intent là gì?
- Trao đổi dữ liệu giữ các Activity
- Sử dụng Intent-Filter
- Giới thiệu Fragments
- Làm việc với Notifications

Intents là gì?

- **Intent** là một cấu trúc thông điệp mô tả hành động sẽ được thực thi. Nó đại diện cho một hành động đi kèm với dữ liệu theo một ngữ cảnh xác định.
- **Intent** là chuẩn giao tiếp giữa các thành phần trong ứng dụng Android, được sử dụng để chạy Activity, Service hoặc một thành phần khác của ứng dụng hoặc hệ thống.



Sử dụng Intent

- Về cơ bản, Intent là đối tượng của lớp android.content.Intent.
- Thành phần chính của Intent bao gồm:
  - *Action*: xác định hành động sẽ được thực thi, các hành động này có thể là: ACTION_VIEW, ACTION_EDIT, ACTION_MAIN…
  - *Data*: là các dữ liệu đi kèm với Intent, được sử dụng để hành động (Action) thao tác trên nó.

Cách gọi Intent

- Tạo đối tượng Intent với tham số Action/Data

```
Intent  myActivity = new Intent (action, data);

startActivity (myActivity);
```

Built-in
or
user-created
activity

Primary data (as an URI)
tel://
http://
sendto://

Các Action và data thường dùng

ACTION_DIAL ***tel:123***
Display the phone dialer with the given number filled in.

ACTION_VIEW ***http://www.google.com***
Show Google page in a browser view. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.

ACTION_EDIT ***content://contacts/people/2***
Edit information about the person whose identifier is "2".

ACTION_VIEW ***content://contacts/people/2***
Used to start an activity to display 2-*nd* person.

ACTION_VIEW ***content://contacts/*** **people/**
Display a list of people, which the user can browse through. Selecting a particular person to view would result in a new intent

Ví dụ sử dụng Action/Data

```
Intent myActivity2 = new Intent (Intent.ACTION_DIAL,
                                 Uri.parse( "tel:555-1234"));
startActivity(myActivity2);
```

Ví dụ sử dụng Intent

- Một số Intent không cần truyền theo dữ liệu cụ thể, mà chỉ cần kiểu của dữ liệu
- Ví dụ: Mở Activity cho người dùng chọn contact từ danh bạ:

```
Intent it = new Intent(Intent.ACTION_PICK);
it.setType(ContactsContract.Contacts.CONTENT_TYPE);
startActivityForResult(it, 1234);
```

Các tham số của Intent

- Ngoài Action/Data, còn có một số tham số phụ thường dùng với Intent:
  - Category
  - Type
  - Component
  - Extra

Các tham số của Intent

- **Category**: Thông tin chi tiết về hành động được thực thi, ví dụ như CATEGORY_LAUNCHER có nghĩa là nó sẽ xuất hiện trong Launcher
- **Type**: Chỉ định kiểu dữ liệu (kiểu MIME) được mang bởi Intent. Thường thì Type được suy ra từ chính dữ liệu nếu không xác định Type.
- **Component:** Chỉ định rõ tên của lớp thành phần (thường là Activity) để thực thi của Intent.
- ***Extras***: là một đối tượng Bundle dùng để chứa các thông tin kèm theo được dùng để cung cấp thông tin cần thiết cho component.

Các tham số của Intent

ACTION

Standard

ACTION_MAIN
ACTION_VIEW
ACTION_ATTACH_DATA
ACTION_EDIT
ACTION_PICK
ACTION_CHOOSER
ACTION_GET_CONTENT
ACTION_DIAL
ACTION_CALL
ACTION_SEND
ACTION_SENDTO
ACTION_ANSWER
ACTION_INSERT
ACTION_DELETE
ACTION_RUN
ACTION_SYNC
ACTION_PICK_ACTIVITY
ACTION_SEARCH
ACTION_WEB_SEARCH
ACTION_FACTORY_TEST

ACTION_TIME_TICK
ACTION_TIME_CHANGED
ACTION_TIMEZONE_CHANGED
ACTION_BOOT_COMPLETED
ACTION_PACKAGE_ADDED
ACTION_PACKAGE_CHANGED
ACTION_PACKAGE_REMOVED
ACTION_PACKAGE_RESTARTED
ACTION_PACKAGE_DATA_CLEARED
ACTION_UID_REMOVED
ACTION_BATTERY_CHANGED
ACTION_POWER_CONNECTED
ACTION_POWER_DISCONNECTED
ACTION_SHUTDOWN

DATA

URI

CONTENTS *such as:*

content://contacts/
content://contacts/1

SCHEME *such as:*

tel:123
http://aaa.bbb.ccc
mailto://aa@bbb.ccc
ftp://aaa.bbb.ccc
. . .
pop://
smtp://
ssl://

MISC

Category

CATEGORY_DEFAULT
CATEGORY_BROWSABLE
CATEGORY_TAB
CATEGORY_ALTERNATIVE
CATEGORY_SELECTED_ALTERNATIVE
CATEGORY_LAUNCHER
CATEGORY_INFO
CATEGORY_HOME
CATEGORY_PREFERENCE
CATEGORY_TEST

MIME

Explicit type (a MIME type) of the intent data.

Component

Explicit name of a component class to use for the intent.

Extras

putExtra(String, Bundle)

Flags

Các ví dụ sử dụng Intent

- Hiển thị danh bạ liên lạc

```
String myData = "content://contacts/people/";

Intent myActivity2 = new Intent(Intent.ACTION_VIEW,
                                Uri.parse(myData));

startActivity(myActivity2);
```

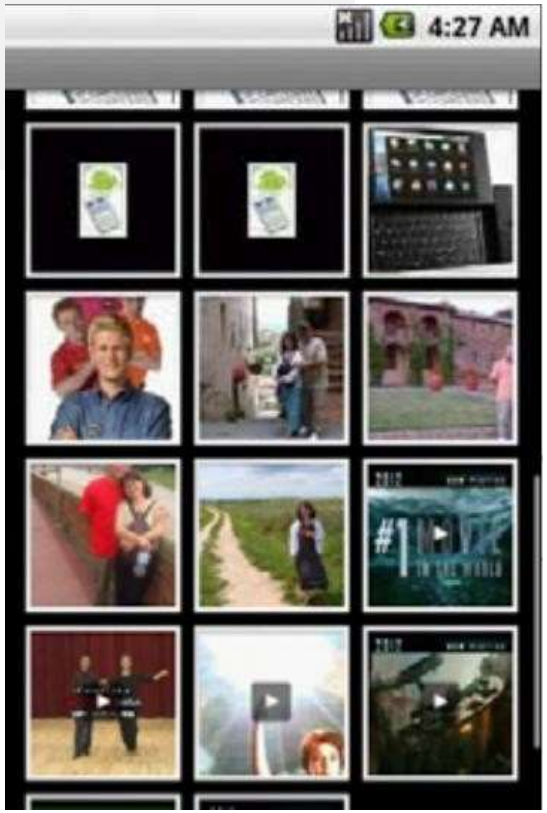
- Xem nội dung trang web:

```
String myData = "http://www.youTube.com";

Intent myActivity2 = new Intent(Intent.ACTION_VIEW,
                                Uri.parse(myData));

startActivity(myActivity2);
```

Các ví dụ sử dụng Intent

- Mở gallery hiển thị các hình ảnh của phone

```
Intent myIntent = new Intent();

myIntent.setType("image/pictures/*");
myIntent.setAction(Intent.ACTION_GET_CONTENT);

startActivity(myIntent);
```

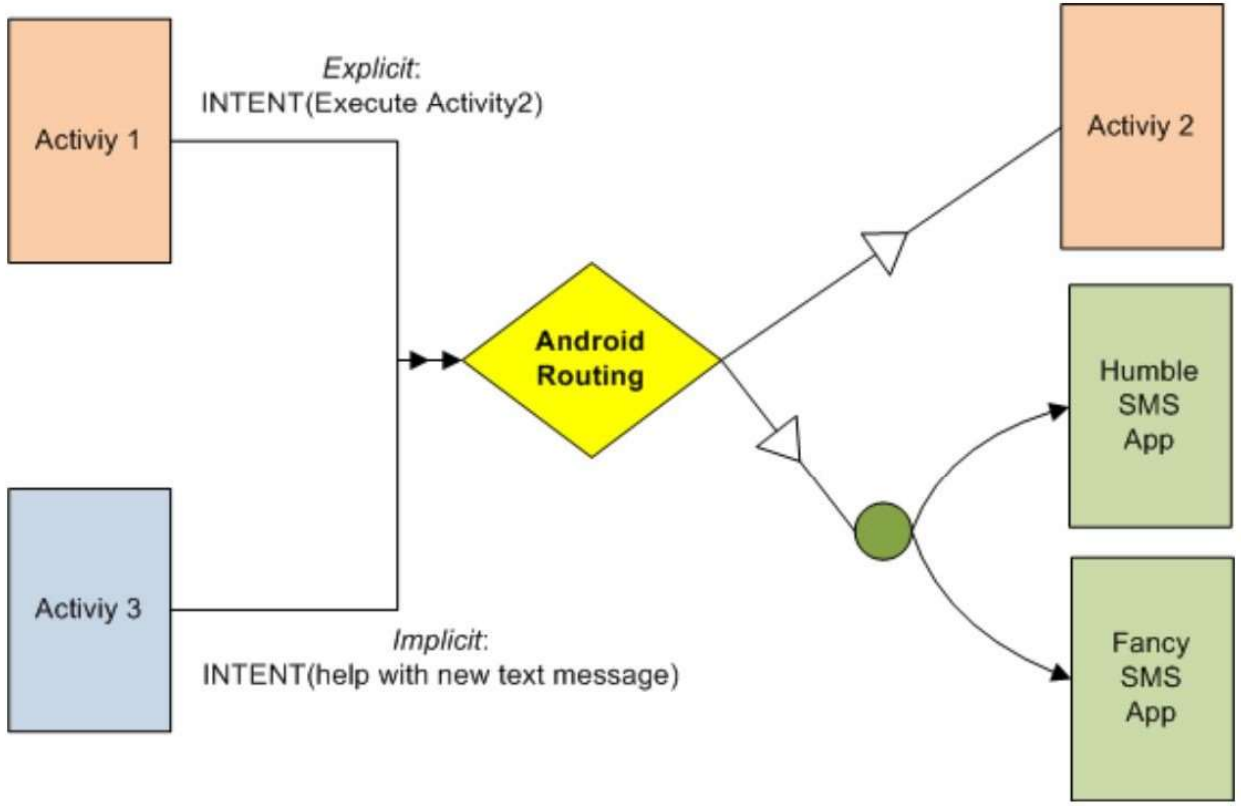


Các ví dụ sử dụng Intent

- Để mở chạy một Activity khác, sử dụng theo dạng
  - Intent it = new Intent(<context>, <component_name>);
- Ví dụ:
  - Intent it = new Intent(this, LoginActivity.class);
  - startActivity(it);

Các loại Intent

- Có 2 loại Intent:
  - Explicit Intent (Intent tường minh)
  - Implicit Intent (Intent không tường minh)



Explicit Intent

- Intent xác định rõ một component để xử lý (chẳng hạn tên Activity)
  - Có thể được gán một giá trị cụ thể sử dụng hàm setComponent() hoặc setClass().
- Intent này thường không chứa bất kỳ thông tin nào khác (như category, type) và thường được dùng để kết nối với các Activity trong cùng một ứng dụng.
  - Ví dụ: khởi chạy một Activity khác.

```
// Explicit Intent by specifying its class name
Intent i = new Intent(this, SecondAcitivity.class);
// Starts TargetActivity
startActivity(i);
```

Implicit Intent

- Intent không chỉ định rõ một component cụ thể. Nhưng nó sẽ chứa các thông tin cần thiết (Action/Data) để hệ thống có thể xác định component thực thi Intentđó.
- Intent này thường được dùng để chạy các Component của ứng dụng khác.
  - Ví dụ: Mở Activity xem nội dung trangweb

```
Intent intent = new Intent(
    android.content.Intent.ACTION_VIEW,
    Uri.parse("http://www.example.com"));
// Starts TargetActivity
startActivity(intent);
```

Dữ liệu phụ (extra) của Intent

- Dữ liệu phụ (extra) là các thông tin kèm theo được dùng để cung cấp thông tin bổ sung cho component như dữ liệu các biến,...
- Có 2 cách để gắn dữ liệu phụ cho Intent:
  - Dùng hàm putExtra() của Intent theo dạng

    intent.putExtra(<key_name>, <value>)
  - Dùng đối tượng Bundle để gói nhiều dữ liệu.

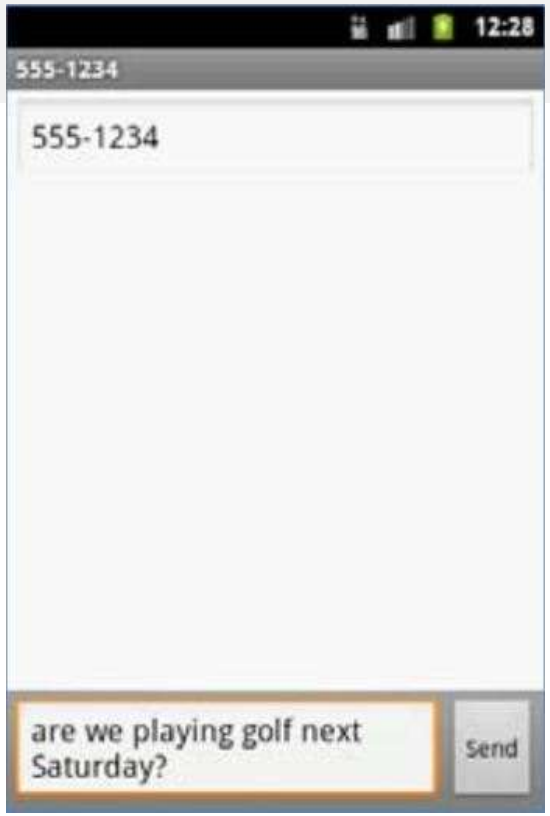
Dữ liệu phụ (extra) của Intent

- Ví dụ sử dụng putExtra()
  - intent.putExtra(“student_name”, “Ngọc Hoa”);
  - Intent.putExtra(“student_age”, 18);
- Ví dụ sử dụng gói bundle để gắn vào Intent
  - Bundle extra = new Bundle();
  - extra.putString("student_country", “Da Nang");
  - extra.putInt("student_yearbirth", 1998);
  - intent.putExtras(extra);

Ví dụ sử dụng Intent - Extra

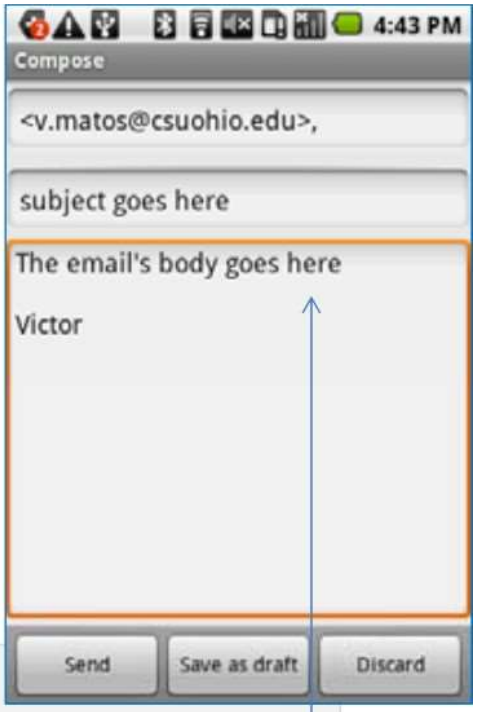
- Gữi tin nhắn SMS

```
Intent intent = new Intent( Intent.ACTION_SENDTO,
                        Uri.parse("sms:5551234"));

intent.putExtra("sms_body", "are we playing golf next Saturday?");

startActivity(intent);
```



Ví dụ sử dụng Intent - Extra

- Gửi email




```
// send email
Uri uri = Uri.parse("mailto:v.matos@csuohio.edu");
Intent myActivity2 = new Intent(Intent.ACTION_SENDTO, uri);

// you may skip the next two pieces [subject/text]
myActivity2.putExtra(Intent.EXTRA_SUBJECT,
                     "subject goes here");
myActivity2.putExtra(Intent.EXTRA_TEXT,
                     "The email's body goes here");

startActivity(myActivity2);
```

34

Truy xuất dữ liệu extra

- Truy xuất trực tiếp từ đối tượng intent, dùng các phương phức có sẵn của Intent theo dạng:
  - Intent.get<Kiểu dữ liêu>Extra(<key_name>, <default_value>);
- Ví dụ:
  - String name = it.getStringExtra("student_name");
  - int age = it.getIntExtra("student_age", 0);

Truy xuất dữ liệu extra

- Truy xuất thông qua đối tượng bundle:
  - Dùng hàm getExtras() để truy xuất đối tượng Bundle trong Intent.
  - Dùng hàm bundle.get<Kiểu dữ liệu>(<key_name>, <default_value>) để truy xuất dữ liệu.
- Ví dụ:
  - Bundle bundle = it.getExtras();
  - String country = bundle.getString("student_country");
  - int year = bundle.getInt("student_yearbirth");

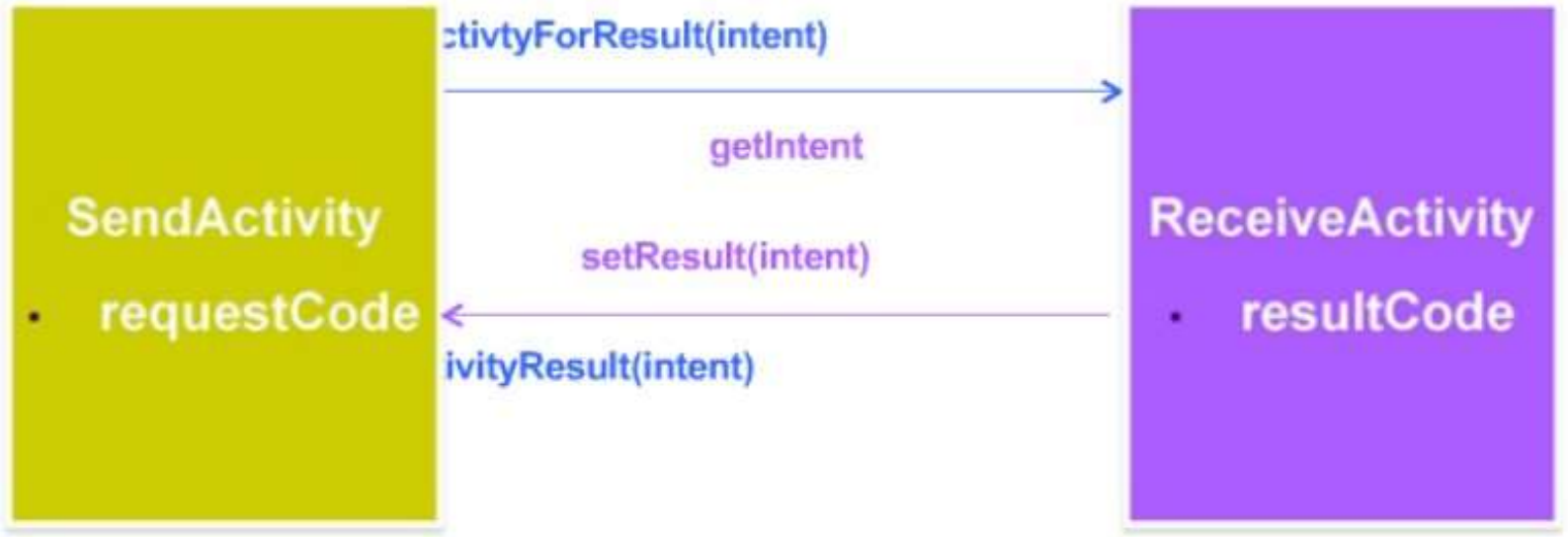
Trao đổi dữ liệu giữa các Activity

- Một thao tác phổ biến trong lập trình Android là truyền dữ liệu cho Activity mới được mở ra.
- Activity mới xử lý dữ liệu từ Activity và trả kết quả cho Activity gửi.
- Activity ban đầu nhận kết quả và hiển thị cho người dùng

Các bước thực hiện

- Bước 1: Mở Activity mới với đối tượng Intent thông qua phương thức
  - startActivityForResult(Intent, requestCode)
- Bước 2: Nhận và xử lý Intent, sau đó xác nhận thông tin phản hồi thông qua phương thức setResult():
  - Intent intent1 = getIntent();
  - setResult(resultCode, Intent2)
- Bước 3: Activity ban đầu ghi đè phương thức sự kiện onActivityResult() để nhận kết quả
  - onActivityResult(requestCode, resultCode, Intent)

Các bước thực hiện – mô hình



Ví dụ: Trao đổi dữ liệu Activity

- Activity A truyền "student_name", "student_dtb",… cho Activity B (dùng phương thức putExtra).
- Activity B nhận Intent và dữ liệu của A. Xử lý theo yêu cầu ứng dụng (Hiển thị ra màn hình logcat, hoặc trên giao diện)
- Activity B trả kết quả cho Activity A. Tùy theo DTB để trả kết quả là: "Xuat Sac", "Binh Thuong"
- Activity A nhận và xử lý kết quả theo yêu cầu (Hiển thị ra màn hình logcat hoặc giao diện).

B1: Mở Activity với dữ liệu

- Để truyền dữ liệu cho Activity mới, gắn dữ liệu cần truyền vào đối tượng Intent
  - Intent it = new Intent(this, StudentActivity.class);
  - it.putExtra("student_name", “Ngoc Hoa");
  - it.putExtra("student_dtb", 8.2);
  - // Yêu cầu start Activity chỉ định trong Intent.
  - this.startActivityForResult(it, 1234);

B2: Nhận dữ liệu từ Activity gửi

- Trong hàm onCreate của Activity mới, ta có thể lấy dữ liệu được truyền sang theo các hàm sau:
  - Intent it = getIntent();
  - String name = it.getStringExtra("student_name");
  - float dtb = it.getIntExtra("student_dtb", 0);

B2: Trả kết quả cho Activity gửi (tt)

- Trong Activity mới, sau khi nhận và xử lý dữ liệu xong, có thể gữi trả lại kết quả đã xử lý cho Activity ban đầu như sau.
  - String xloai = (dtb > 8.0) ? “Xuat Sac” : “Binh Thuong”;
  - Intent itRet = new Intent();
  - itRet.putExtra(“xeploai”, xloai);
  - setResult(RESULT_OK, itRet);

B3: Lấy kết quả trả về

- Để lấy kết quả trả về, Activity gữi có thể lấy thông qua hàm sự kiện onActivityResult()

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data)
{
   if (requestCode == 1234 && resultCode == RESULT_OK) {
      String xloai = data.getStringExtra("xeploai");
      Log.d(TAG, "Xep loai: " + xloai);
   }
}
```

Ví dụ: Xem danh bạ

- Tạo ứng dụng cho phép người dùng chọn một contact từ danh bạ và hiển thị chị tiết của contact đó.
  - Khởi chạy Activity của ứng dụng hệ thống (Contacts – dang bạ) để chọn một liên lạc. Và chờ nhận kết quả (không đồng bộ).
  - Sau khi người dùng chọn một liên lạc từ danh bạ. Ứng dụng danh bạ sẽ trả thông tin cho Activity gọi.
  - Activity gọi xử lý dữ liệu nhận được và hiển thị cho người dùng.

Ví dụ: Xem danh bạ

- Khởi chạy ứng dụng danh bạ để chọn

```
Intent it = new Intent(Intent.ACTION_PICK);
it.setType(ContactsContract.Contacts.CONTENT_TYPE);
startActivityForResult(it, 1234);
```

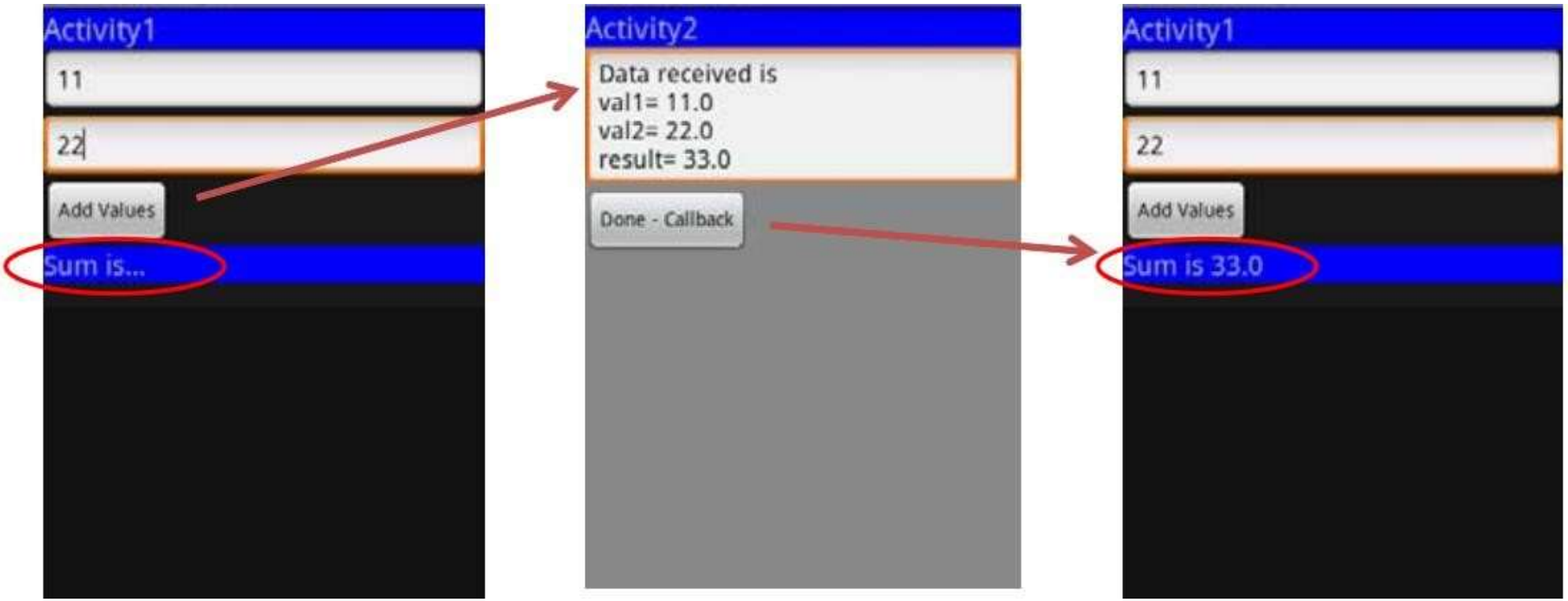
Ví dụ: Xem danh bạ

- Activity gọi chờ nhận kết quả và xử lý:

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
                                @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if( requestCode == 1234 && resultCode == RESULT_OK) {
        // View details of selected contact
        Uri uri = data.getData();
        Intent it = new Intent(Intent.ACTION_VIEW);
        it.setData(uri);
        startActivity(it);
    }
}
```

Thực hành: Trao đổi dữ liệu Activity

- Tạo ứng dụng như sau: Activity 1 truyền 2 số thực từ UI cho Activity 2 (theo dạng Bundle). Activity 2 xử lý và trả kết quả tổng cho Activity 1 như sau.



Activity 1: Tạo Intent với bundle

```
btnAdd.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        // get values from the UI
        Double v1 = Double.parseDouble(txtVal1.getText().toString());
        Double v2 = Double.parseDouble(txtVal2.getText().toString());

        // create intent to call Activity2
        Intent myIntentA1A2 = new Intent (Activity1.this, Activity2.class);
        // create a container to ship data
        Bundle myData = new Bundle();

        // add <key,value> data items to the container
        myData.putDouble("val1", v1);
        myData.putDouble("val2", v2);

        // attach the container to the intent
        myIntentA1A2.putExtras(myData);

        // call Activity2, tell your local listener to wait for response
        startActivityForResult(myIntentA1A2, 101);
    }
});
```

Activity 2: Nhận intent và xử lý dữ liệu

```
dataReceived = (EditText) findViewById(R.id.etDataReceived);
// pick call made to Activity2 via Intent
Intent myLocalIntent = getIntent();

// look into the bundle sent to Activity2 for data items
Bundle myBundle =  myLocalIntent.getExtras();
Double v1 = myBundle.getDouble("val1");
Double v2 = myBundle.getDouble("val2");
// operate on the input data
vResult =  v1 + v2;
// for illustration purposes. show data received & result
dataReceived.setText("Data received is \n"
      + "val1= " + v1 + "\nval2= " + v2
      + "\n\nresult= " + vResult);
```

Activity 1: Nhận kết quả và hiển thị tổng

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
   super.onActivityResult(requestCode, resultCode, data);

   try   {
      if ((requestCode == 101 ) && (resultCode == Activity.RESULT_OK)){

         Bundle myResults = data.getExtras();

         Double vresult = myResults.getDouble("vresult");

         lblResult.setText("Sum is " + vresult);

      }
   }
   catch (Exception e) {
      lblResult.setText("Problems - " + requestCode + " " + resultCode);
   }
}//onActivityResult
```

Activity 2: Trả kết quả cho Activity 1

```
btnDone = (Button) findViewById(R.id.btnDone);
btnDone.setOnClickListener(new OnClickListener() {
   @Override
   public void onClick(View v) {
      // add to the bundle the computed result
      myBundle.putDouble("vresult", vResult);

      // attach updated bumble to invoking intent
      myLocalIntent.putExtras(myBundle);

      // return sending an OK signal to calling activity
      setResult(Activity.RESULT_OK, myLocalIntent);

      // terminate Activity2
      finish();

   }
});
```

Intent Filter

- Bất cứ thành phần nào (Activity, BroadcastReceiver, Service) khi muốn sử dụng trong ứng dụng đều phải được đăng kí trong file AndroidManifest.xml.
- Intent-Fillter là bản đặc tả có cấu trúc của các giá trị của Intent dùng để xác định component phù hợp để thực hiện các hành động được đặc tả trong Intent.
- Android dùng Intent-Filter để xác định các Activity, Service, hay Broadcast Receiver nào phù hợp với Intent thông các thuộc tính Action, Category, Data scheme tương ứng với Intent đó.

Intent Filter

- Intent Fillter được khai báo trong AndroidManifest.xml và sử dụng thẻ <intent-fillter>
- Một Intent-filter có các thành phần chính sau:
  - *Action*: Hành động mà component có thể thực thi.
  - *Category*: Phân nhóm các hành động.
  - Data: chỉ ra URI của kiểu dữ liệu MIME. Có các thuộc tính như **scheme, host, port**, và **path**.

Intent Filter

```
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

```
<activity android:name=".LayoutActivity">
    <intent-filter>
        <action android:name="android.intent.action.SENDTO" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:scheme="sms" />
    </intent-filter>
</activity>
```

Intent Filter

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.o7planning.explicitintentexample" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >

        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name=".GreetingActivity" >
        </activity>

    </application>

</manifest>
```

Luật xác định Intent-Filter

- Hệ thống sẽ tiến hành xem xét từ trên xuống
- Khi một Intent được gọi, Android sẽ tìm kiếm những component có action-name phù hợp với Intent
  - Nếu có sẽ mở component đó lên để thực thi
  - Nếu có nhiều hơn một component có action-name phù hợp thì Android sẽ yêu cầu người dùng chọn component phù hợp.
  - Ngược lại, nếu không có component nào phù hợp Android sẽ tiến hành xem xét kiểu dữ liệu của Intent, sau đó là Category.

Giới thiệu Fragments

- Fragments là một phần quan trọng trong phát triển ứng dụng Android, cho phép xây dựng giao diện người dùng linh hoạt và dễ quản lý.
- Là các đoạn giao diện tái sử dụng được, có thể kết hợp lại để tạo nên một UI thích ứng với nhiều kích thước màn hình và cấu hình khác nhau.
- Fragment có vòng đời riêng, tương tự như activity, với các callback phương thức như onCreate(), onCreateView(), onStart(), onResume(), onPause(), onStop(), và onDestroy()
- Fragments có thể giao tiếp với activity chứa nó và các fragment khác thông qua interfaces hoặc view model, tạo nên sự linh hoạt trong cách xử lý dữ liệu và sự kiện.
- Fragment Transactions: FragmentManager cho phép thực hiện các giao dịch fragment như thêm, thay thế, hoặc xóa fragment động, giúp tạo nên giao diện động và tương tác.

Giới thiệu Fragments

- Ví dụ về Fragments:

```
public class DemoFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_demo,
container, false);
    }
}
```

Giới thiệu Fragments

- Ví dụ về Fragments:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <!-- Your fragment's UI elements go here -->
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello from Fragment!" />
</LinearLayout>
```

Giới thiệu Fragments

- Ví dụ về Fragments:

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        if (savedInstanceState == null) {
            getSupportFragmentManager().beginTransaction()
                .replace(R.id.fragment_container, new DemoFragment())
                .commit();
        }
    }
}
```

Giới thiệu Fragments

- Ví dụ về Fragments: activity_main.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/fragment_container">
</FrameLayout>
```

Làm việc với Notificationns

- Notifications trong Android cho phép bạn hiển thị thông báo cho người dùng ngay cả khi họ không sử dụng ứng dụng của bạn.
- Các bước cơ bản để tạo và quản lý notifications:
  - Thêm quyền thông báo vào tệp Manifest.
  - Tạo kênh thông báo : Từ Android 8.0 (API cấp 26) trở lên, bạn cần tạo một kênh thông báo cho ứng dụng.
  - Tạo và hiển thị thông báo: Sử dụng NotificationCompat.Builder để tạo và hiển thị thông báo
  - Xử lý hành động của thông báo : có thể thêm các hành động vào thông báo, chẳng hạn như nút "Reply" hoặc "Mark as Done“.
  - Cập nhật hoặc huỷ thông báo.

Làm việc với Notificationns

- Ví dụ Notifications trong Android :

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Tạo và hiển thị thông báo khi Activity được tạo
        showNotification();
    }
}
```

```
private void showNotification() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        CharSequence name = "MyChannel";
        String description = "Channel Description";
        int importance =
NotificationManager.IMPORTANCE_DEFAULT;
        NotificationChannel channel = new
NotificationChannel("my_channel_id", name, importance);
        channel.setDescription(description);
        NotificationManager notificationManager =
getSystemService(NotificationManager.class);
        notificationManager.createNotificationChannel(channel);
    }
```

```
    NotificationCompat.Builder builder = new NotificationCompat.Builder(this,
    "my_channel_id")
            .setSmallIcon(R.drawable.notification_icon)
            .setContentTitle("Thông báo của tôi")
            .setContentText("Nội dung thông báo")
            .setPriority(NotificationCompat.PRIORITY_DEFAULT);

        Intent intent = new Intent(this, MainActivity.class);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent,
    PendingIntent.FLAG_UPDATE_CURRENT);
        builder.setContentIntent(pendingIntent);

        NotificationManagerCompat notificationManager =
    NotificationManagerCompat.from(this);
        notificationManager.notify(1, builder.build());
}
```