BÀI 6: SỬ DỤNG INTENT VỚI ACTIVITY, SERVICE VÀ BROADCAST RECEIVER

I. Tóm tắt nội dung thực hành

1. Yêu cầu lý thuyết

1. Intent và Activity:

Intent là một đối tượng dùng để giao tiếp giữa các thành phần trong ứng dụng Android, bao gồm cả các Activity. Khi một Activity muốn khởi động một Activity khác, nó sẽ tạo một Intent và gửi cho hệ thống Android.

Ví dụ: Giả sử ứng dụng của có hai Activity: MainActivity và WebViewActivity. Khi người dùng nhấn vào nút "Mở trang web" trong MainActivity, ứng dụng sẽ tạo một Intent với hành động VIEW và dữ liệu là URL của trang web, sau đó gửi Intent này cho hệ thống Android để mở WebViewActivity và hiển thị trang web.

```
public void openWebView(View view) {
  Intent intent = new Intent(this, WebViewActivity.class);
  intent.setAction(Intent.ACTION_VIEW);
  intent.setData(Uri.parse("https://www.example.com"));
  startActivity(intent);
}
```

2. Intent và Service:

Khi một Activity muốn thực hiện một tác vụ nền, nó có thể khởi động một Service bằng cách gửi một Intent cho Service. Service sẽ chạy ngầm trong ứng dụng và thực hiện tác vụ đó mà không cần hiển thị giao diện người dùng.

Ví dụ: Giả sử ứng dụng có một tính năng tải xuống tập tin trên nền. Khi người dùng nhấn vào nút "Tải xuống" trong MainActivity, ứng dụng sẽ tạo một Intent với hành

động "DOWNLOAD" và dữ liệu là URL của tệp tin, sau đó gửi Intent này cho DownloadService để thực hiện tác vụ tải xuống.

```
//MainActivity.java
public void downloadFile(View view) {
    Intent intent = new Intent(this, DownloadService.class);
    intent.setAction(Intent.ACTION_DOWNLOAD);
    intent.setData(Uri.parse("https://example.com/file.zip"));
    startService(intent);
}
// DownloadService.java
public class DownloadService extends Service {
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        if (intent.getAction().equals(Intent.ACTION_DOWNLOAD)) {
            Uri fileUrl = intent.getData();
            // Thực hiện tác vụ tải xuống tệp tin
            downloadFile(fileUrl);
        }
        return START_STICKY;
    }
}
```

3. Intent và Broadcast Receiver:

Broadcast Receiver là một thành phần của ứng dụng Android có thể nhận và xử lý các sự kiện (Broadcast) được gửi bởi hệ thống hoặc các thành phần khác. Khi một Broadcast được gửi, các Broadcast Receiver đăng ký để lắng nghe sự kiện đó sẽ nhận được Intent chứa thông tin về sự kiện.

Ví dụ: Giả sử ứng dụng cần hiển thị một thông báo khi pin của thiết bị sắp hết. Khi hệ thống Android phát hiện pin sắp hết, nó sẽ gửi một Broadcast với hành động "BATTERY_LOW". Ứng dụng có thể đăng ký một Broadcast Receiver để lắng nghe sự kiện này, và khi nhận được Intent, nó sẽ hiển thị một thông báo cho người dùng.

```
// BatteryLowReceiver.java
public class BatteryLowReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        if (intent.getAction().equals(Intent.ACTION_BATTERY_LOW)) {
            // Hiển thị thông báo pin sắp hết
            showBatteryLowNotification(context);
        }
    }
}
```

```
// AndroidManifest.xml
<!-- AndroidManifest.xml -->
  <receiver android:name=".BatteryLowReceiver">
```

```
    <intent-filter>
        <action android:name="android.intent.action.BATTERY_LOW" />
    </intent-filter>
</receiver>
```

Trong các ví dụ trên, Intent được sử dụng để liên kết các thành phần của ứng dụng (Activity, Service, Broadcast Receiver) với nhau, giúp chúng có thể giao tiếp và trao đổi thông tin. Intent là một cơ chế quan trọng trong việc tạo ra các ứng dụng Android linh hoạt và có khả năng tái sử dụng.

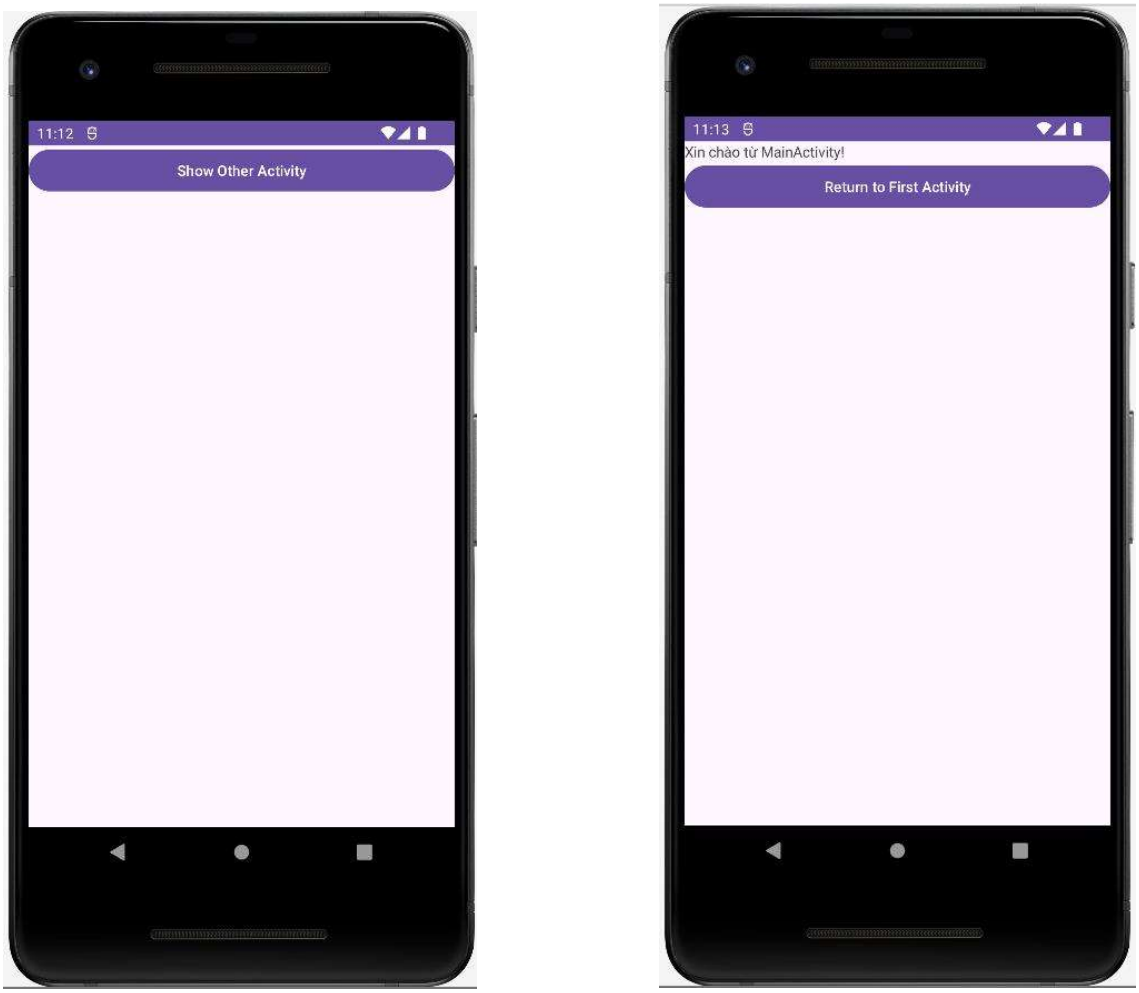
2. Nội dung

Thực hiện các bài tập sử dụng intent với Activity, Service và Broadcast Receiver.

II. Bài tập yêu cầu

Bài 1: (Intent và Activity)

Xây dựng ứng dụng minh họa cách sử dụng Intent để chuyển giữa các Activity. Trong MainActivity, tạo một nút Button để chuyển sang OtherActivity. Khi nút được nhấn, sẽ tạo một Intent và truyền một thông điệp qua Intent để hiển thị trên OtherActivity. Ở OtherActivity, nhận thông điệp từ Intent và hiển thị nó trên màn hình. Cuối cùng, tạo button để quay lại MainActivity.



Hình 50 Giao diện bài tập 6.1

Hướng dẫn:

Tạo giao diện activity_main.xml gồm 1 button

```
<Button
    android:id="@+id/btn_open_other_activity"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Show Other Activity"

    tools:layout_editor_absoluteX="150dp"
    tools:layout_editor_absoluteY="134dp"
    tools:ignore="MissingConstraints" />
```

Tạo giao diện activity_other.xml gồm 1 TextView và 1 Button

```
<TextView
    android:id="@+id/text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    tools:ignore="MissingConstraints" />
```

```
<Button
    android:id="@+id/btn_back_to_main"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Return to First Activity"
    tools:ignore="MissingConstraints"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="328dp" />
```

Tạo class OtherActivity trong file OtherActivity.java, trong đó hiển thị giao diện activity_other và message nhận được từ intent. Viết thêm xử lý cho Button để quay về MainActivity.

```java
public class OtherActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_other);
        // Nhận dữ liệu từ Intent
        String message = getIntent().getStringExtra( name: "message");
        // Hiển thị dữ liệu trên TextView
        TextView textView = findViewById(R.id.text_view);
        textView.setText(message);
        // Tạo nút bấm để quay lại MainActivity
        Button btnBackToMain = findViewById(R.id.btn_back_to_main);
        btnBackToMain.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Kết thúc Activity hiện tại và quay lại MainActivity
                finish();
            }
        });
    }
}
```

Thêm thẻ activity trong application của AndroidManifest.xml,

```xml
<activity
    android:name=".OtherActivity"
    android:exported="false">
</activity>
```

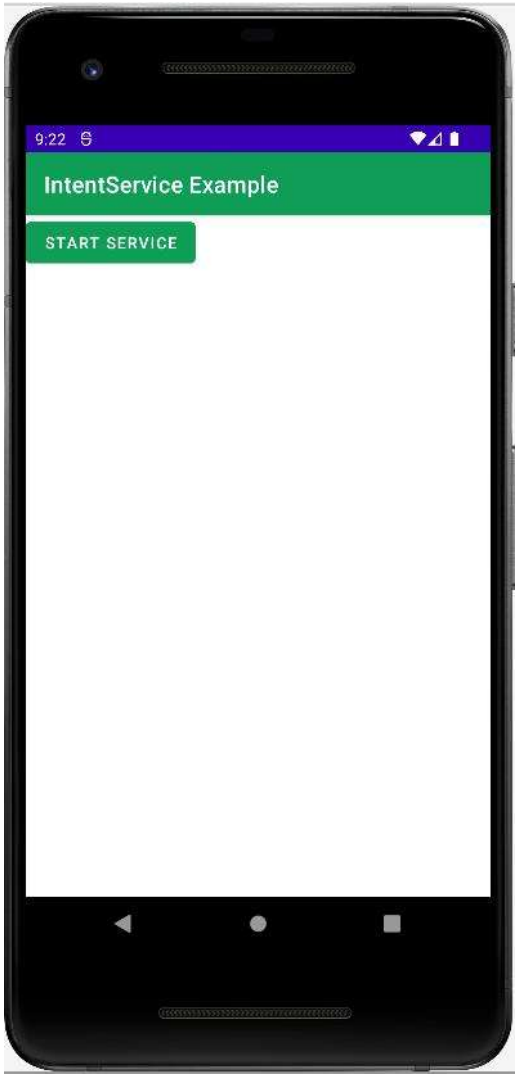
Viết xử lý Button để chuyển màn hình sang OtherActivity

```java
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Tạo nút bấm để chuyển sang Activity khác
        Button btnOpenOtherActivity = findViewById(R.id.btn_open_other_activity);
        btnOpenOtherActivity.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Tạo Intent để chuyển sang Activity khác
                Intent intent = new Intent(getApplicationContext(), OtherActivity.class);
                // Truyền dữ liệu qua Intent
                intent.putExtra( name: "message",  value: "Xin chào từ MainActivity!");
                // Khởi chạy Activity mới
                startActivity(intent);
            }
        });
    }
}
```

Chạy chương trình và xem kết quả.

Bài 2: (Intent và Service)

Xây dựng ứng dụng sử dụng IntentService như hình, gồm 1 button



Hình 51 Giao diện bài tập 6.2

Logcat thông báo các xử lý của IntentService

```
I  Davey! duration=1071ms; Flags=0, FrameTimelineVsy
D  Task completed
D  app_time_stats: avg=2143.23ms min=5.80ms max=8554
D  Task in progress
D  Task completed
```

Hướng dẫn:

Tạo giao diện gồm 1 button trong activity_main.xml

```
<!-- Button to start the service -->
<Button
    android:id="@+id/start_service_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Start Service"
    tools:ignore="MissingConstraints" />
```

Tạo giao diện thứ hai trong file activity_example_intent_service.xml, có 1 TextView

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Second Activity"
    android:textSize="32dp"
    tools:ignore="MissingConstraints" />
```

Trong AndroidManifest tạo 1 thẻ service có tên là class IntentService sẽ định nghĩa bên dưới

```
<service
    android:name=".ExampleIntentService"
    android:exported="false" />
```

Tạo 1 class java mới là ExampleIntentService thừa kế từ IntentService

```
public class ExampleIntentService extends IntentService {
    // Constructor
    no usages
    public ExampleIntentService() {
        // Call superclass constructor with
        // the name of the worker thread
        super( name: "ExampleIntentService");
    }
```

Trong MainActivity.java định nghĩa lớp MainActivity và viết xử lý cho Button

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Get a reference to the start service button in the layout
        Button startServiceButton = findViewById(R.id.start_service_button);

        // Set an OnClickListener for the button
        startServiceButton.setOnClickListener(v -> {
            // Create an Intent to start
            // the ExampleIntentService
            Intent intent = new Intent( packageContext: MainActivity.this, ExampleIntentService.class);
            // Start the service
            // using the intent
            startService(intent);
        });
    }
}
```

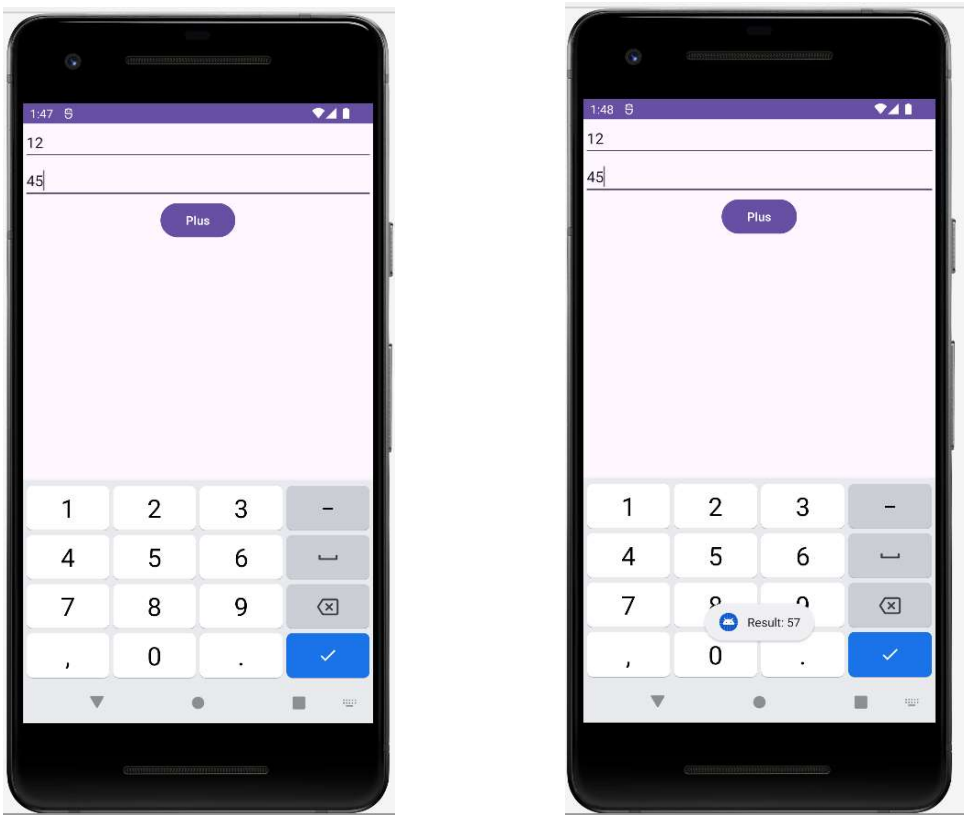
Khi nhấn button, sẽ tạo một intent tường minh chuyển đến IntentService đã định nghĩa. Override hàm onHandleIntent của lớp ExampleIntentService để xuất ra màn hình logcat các hoạt động của service.

```
4 usages
@Override
protected void onHandleIntent(@Nullable Intent intent) {
    // Perform a task in the background
    Log.d( tag: "ExampleIntentService",  msg: "Task in progress");

    // Simulate a long running task by
    // sleeping the thread for 5 seconds
    try {
        Thread.sleep( millis: 5000);
    } catch (InterruptedException e) {
        // Print stack trace if an
        // InterruptedException occurs
        e.printStackTrace();
    }
    Log.d( tag: "ExampleIntentService",  msg: "Task completed");
}
```

Tuy nhiên từ API 30, không còn khuyến khích sử dụng nữa.

Bài 3: (Intent và BroadcastReceiver)

Xây dựng ứng dụng thực hiện phép toán cộng hai số hạng (giao diện như hình). Ứng dụng sẽ tự tạo ra 1 event khi bấm button "Plus" và ứng dụng cũng có riêng 1 bộ bắt sự kiện (BroadcastReceiver) để xử lý sự kiện này. Xuất thông báo kết quả của phép cộng.



Hình 52 Giao diện bài tập 6.3

Hướng dẫn:

Tạo project và thiết kế giao diện gồm:

```
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="number"
    android:hint="Input1"
    android:id="@+id/editText1" />
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="number"
    android:hint="Input2"
    android:id="@+id/editText2" />
<Button
    android:id="@+id/btn"
    android:text="Plus"
    android:layout_gravity="center"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

Sau đó tạo một class CalculatorReceiver thừa kế từ BroadcastReceiver. Trong lớp này thực hiện override hàm onReceive. Khi bắt được sự kiện có tên là Key.ACTION_PLUS_NUMBER (đây là một hằng số tên sự kiện tự tạo trong class MainActivity), thực hiện phép cộng bằng cách cộng giá trị của 2 EditText của 2 số hạng. Sau đó, xuất kết quả của phép cộng.

```
public class CalculatorReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        switch (intent.getAction()) {
            case MainActivity.Key.ACTION_PLUS_NUMBER:
                int a=intent.getIntExtra(MainActivity.Key.NUMBER_A, defaultValue: 0);
                int b=intent.getIntExtra(MainActivity.Key.NUMBER_B, defaultValue: 0);
                Toast.makeText(context, text: "Result: "+(a+b),Toast.LENGTH_SHORT).show();
                break;
            default:
                break;
        }
    }
}
```

Class MainActivity trong MainActivity.java được định nghĩa gồm: lớp Key với các hằng số chuỗi; một số thuộc tính của class là 2 EditText và 1 Button, đối tượng receiver thuộc lớp Calculator để xử lý bắt sự kiện thực hiện phép cộng.

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    7 usages
    public static class Key {
        3 usages
        public static final String ACTION_PLUS_NUMBER="com.example.action_add_number";
        2 usages
        public static final String NUMBER_A="nuber_a";
        2 usages
        public static final String NUMBER_B="nuber_b";
    }
    2 usages
    private EditText edtA, edtB;
    2 usages
    private Button btn;
    3 usages
    private CalculatorReceiver receiver;
```

Trong hàm onCreate của MainActivity, khởi tạo các giá trị cho thuộc tính. Sau đó tạo một inten filter và đăng ký cho bộ lắng nghe sự kiện.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    edtA = (EditText) findViewById(R.id.editText1);
    edtB = (EditText) findViewById(R.id.editText2);
    btn = (Button) findViewById(R.id.btn);
    btn.setOnClickListener(this);

    receiver = new CalculatorReceiver();
    IntentFilter filter = new IntentFilter();
    filter.addAction(Key.ACTION_PLUS_NUMBER);
    registerReceiver(receiver, filter);
}
```

Trong Activity override hàm onDestroy để hủy đăng ký cho receiver khi activity bị hủy

```
@Override
protected void onDestroy() {
    unregisterReceiver(receiver);
    super.onDestroy();
}
```

Viết xử lý cho sự kiện onClick của button “Plus”. Khi nhấn button sẽ tạo ra một intent và broadcast sự kiện có tên là Key.ACTION_PLUS_NUMBER

```
@Override
public void onClick(View v) {
    try {
        int a = Integer.parseInt(edtA.getText().toString());
        int b = Integer.parseInt(edtB.getText().toString());
        Intent intent=new Intent();
        intent.setAction(Key.ACTION_PLUS_NUMBER);
        intent.putExtra(Key.NUMBER_A,a);
        intent.putExtra(Key.NUMBER_B,b);
        sendBroadcast(intent);
    } catch (NumberFormatException e) {
        e.printStackTrace();
    }
}
```

Chạy chương trình và xem kết quả.

III. Câu hỏi ôn tập:

1. Để khởi chạy một Activity mới, chúng ta sử dụng phương thức nào?

   A. startActivity()

   B. launchActivity()

   C. initActivity()

2. Để nhận kết quả trả về từ một Activity, chúng ta sử dụng phương thức nào?

   A. startActivity()

   B. startActivityForResult()

   C. Cả A và B đều đúng

3. Để khởi chạy một Service, chúng ta sử dụng phương thức nào?

   A. startService()

   B. launchService()

C. initService()

4. Để liên kết với một Service, chúng ta sử dụng phương thức nào?

A. bindService()

B. connectService()

C. Cả A và B đều đúng

5. Khi một Service kết thúc, phương thức nào được gọi?

A. onServiceStarted()

B. onServiceFinished()

C. onDestroy()

6. Để đăng ký một Broadcast Receiver, chúng ta sử dụng phương thức nào?

A. registerReceiver()

B. addReceiver()

C. initReceiver()

7. Để gửi một Broadcast, chúng ta sử dụng phương thức nào?

A. sendBroadcast()

B. broadcastIntent()

C. postBroadcast()

8. Câu hỏi: Khi một Broadcast Receiver nhận được một Broadcast, phương thức nào được gọi?

A. onReceive()

B. onBroadcastReceived()

C. onNotify()

9. Để nhận dữ liệu từ một Intent trong Service, chúng ta sử dụng phương thức nào?

A. getIntent()

B. getData()

C. getExtras()

10. Để gửi dữ liệu từ một Service sang một Activity, chúng ta sử dụng phương thức nào?

A. startActivity()

B. sendBroadcast()

C. Cả A và B đều không đúng

11. Để gửi một Broadcast với dữ liệu, chúng ta sử dụng phương thức nào?

A. sendBroadcast()

B. sendBroadcast(Intent)

C. Cả A và B đều đúng

12. Khi nhận được một Broadcast, để lấy dữ liệu từ Intent, chúng ta sử dụng phương thức nào?

A. getIntent()

B. getExtras()

C. getData()