Fpt University

FPT POLYTECHNIC

ANDROID NÂNG CAO

BÀI 1: SERVICE

www.poly.edu.vn

Mục tiêu

- Kết thúc bài học này bạn có khả năng
  - Hiểu rõ về Service
  - Quản lý vòng đời Service
  - Sử dụng Service

NỘI DUNG

- Phần I: Service và vòng đời Service
  - Tổng quan về Service
  - Quản lý vòng đời Service
- Phần II: Sử dụng Service
  - Tạo Service
  - Dừng Service

Fpt University

FPT POLYTECHNIC

BÀI 1: SERVICE

PHẦN I: SERVICE VÀ VÒNG ĐỜI SERVICE

www.poly.edu.vn

TỔNG QUAN VỀ SERVICE

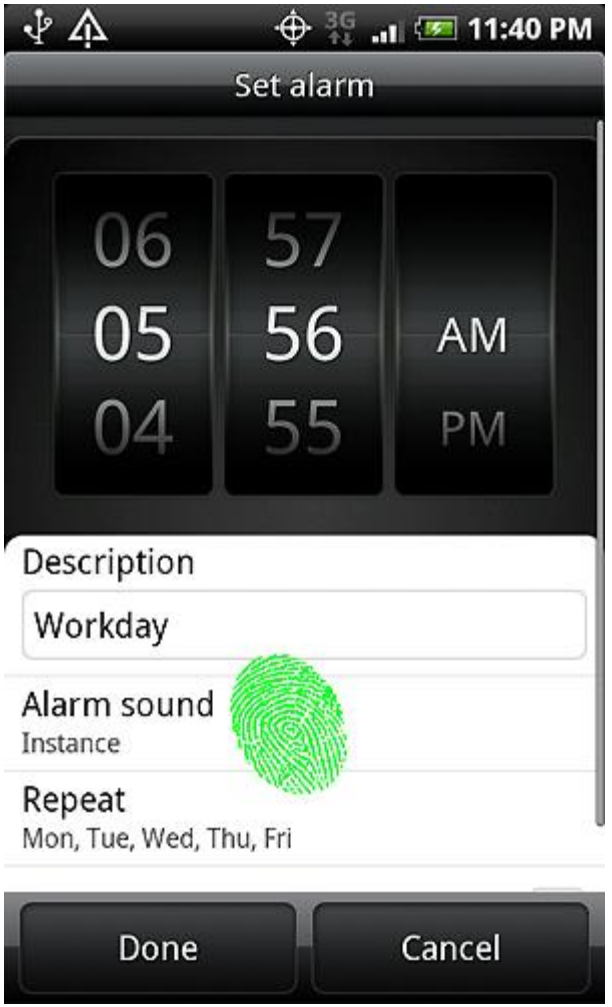
- Service là thành phần ứng dụng có thể thực hiện các thao tác đòi hỏi tốn nhiều thời gian và tài nguyên
- Service không cung cấp giao diện cho người dùng
- Service chạy ngầm để thực hiện các công việc như nghe nhạc, thực hiện ghi và đọc file, hoặc tương tác với Content Provider

Foreground Service

- Là Service mà người dùng biết là Service đang chạy và hệ thống sẽ không kill Service khi bộ nhớ xuống thấp
- Phải cung cấp một notification trên status bar thể hiện Service đang chạy trừ khi Service bị dừng hoặc bị hủy từ foreground
- Ví dụ: music player sẽ sử dụng service để chơi nhạc nên service nên được để ở chế độ chạy foreground vì người dùng biết được bài nhạc đang nghe. Notification trên status bar nên miêu tả bài nhạc hiện tại đang nghe và cho phép người dùng khởi tạo một Activity để tương tác với player

Background Service

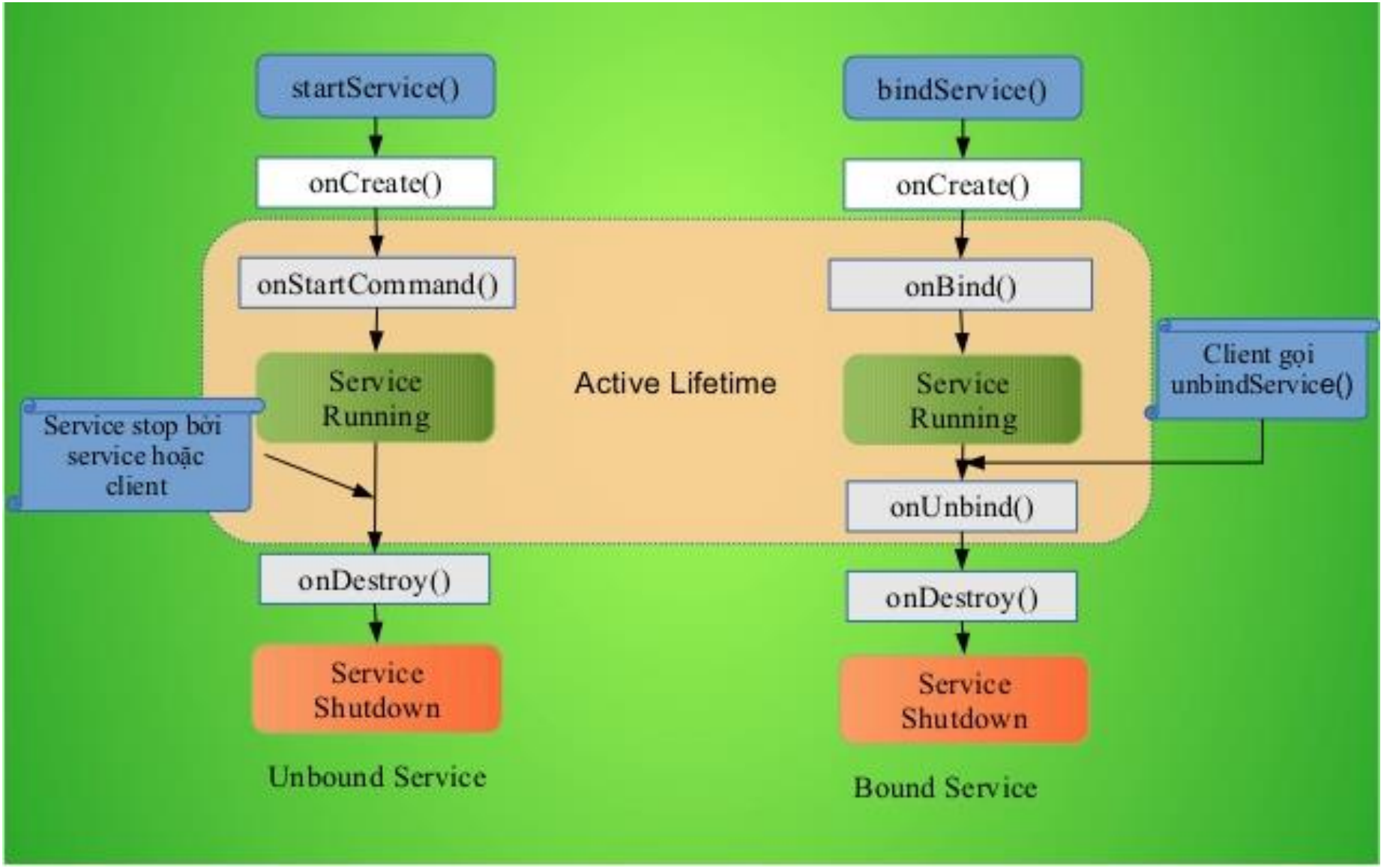
- Là Service chạy mà người dùng không ý thức được là Service đang chạy và không có tương tác với người dùng
- Ví dụ: nhận tin nhắn, nhận cuộc gọi đến, nhận email,...
- Ví dụ: tạo báo thức lúc 5:56am, khi đồng hồ hệ thống điểm 5:56am, thiết bị sẽ thông báo báo thức cho người dùng bằng âm thanh và notification.



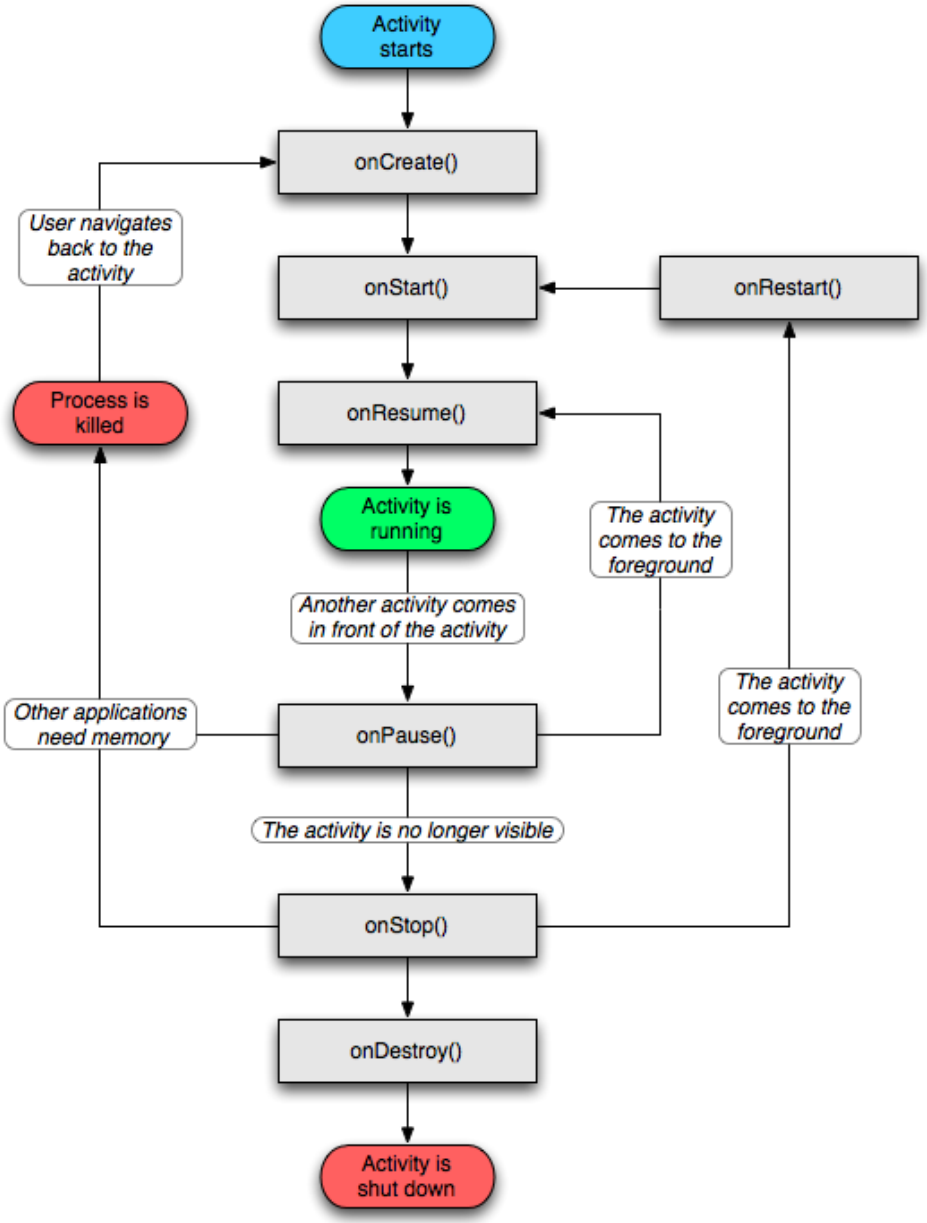
VÒNG ĐỜI CỦA SERVICE

- Vòng đời của Service đơn giản hơn vòng đời của Activity
- Điều quan trọng là bạn phải quan tâm đến Service được tạo và hủy như thế nào, bởi vì Service có thể chạy nền mà người dùng không ý thức được Service đang chạy
- Vòng đời của Service từ khi nó được tạo đến khi nó bị hủy có thể chia làm 2 loại:
  - Unbound Service (Started Service): service không ràng buộc
  - Bound Service: service ràng buộc

VÒNG ĐỜI CỦA SERVICE



VÒNG ĐỜI CỦA SERVICE



So sánh Service và Activity

- Service không có giao diện
- Service không bị kill khi thiết bị bị xoay
- Service: phân cấp độ quan trọng khi bị kill (kill priority)

Fpt University

FPT POLYTECHNIC

BÀI 1: SERVICE

PHẦN II: SỬ DỤNG SERVICE

www.poly.edu.vn

Started Service

- Service đã khởi tạo (Started Service) là service được khởi tạo khi một thành phần ứng dụng (ví dụ như Activity) khởi tạo Service bằng lời gọi phương thức **startService()**
- Ngay khi khởi tạo xong, Service có thể chạy nền vô hạn kể cả khi thành phần khởi tạo Service (ví dụ như Activity khởi tạo Service này) đã bị hủy
- Thông thường, Service đã khởi tạo thực hiện một thao tác đơn giản và không trả lại một kết quả cụ thể. Ví dụ: download và upload một file trên mạng. Khi Service hoàn thành nhiệm vụ việc download và upload, Service sẽ tự động dừng lại

Bound Service

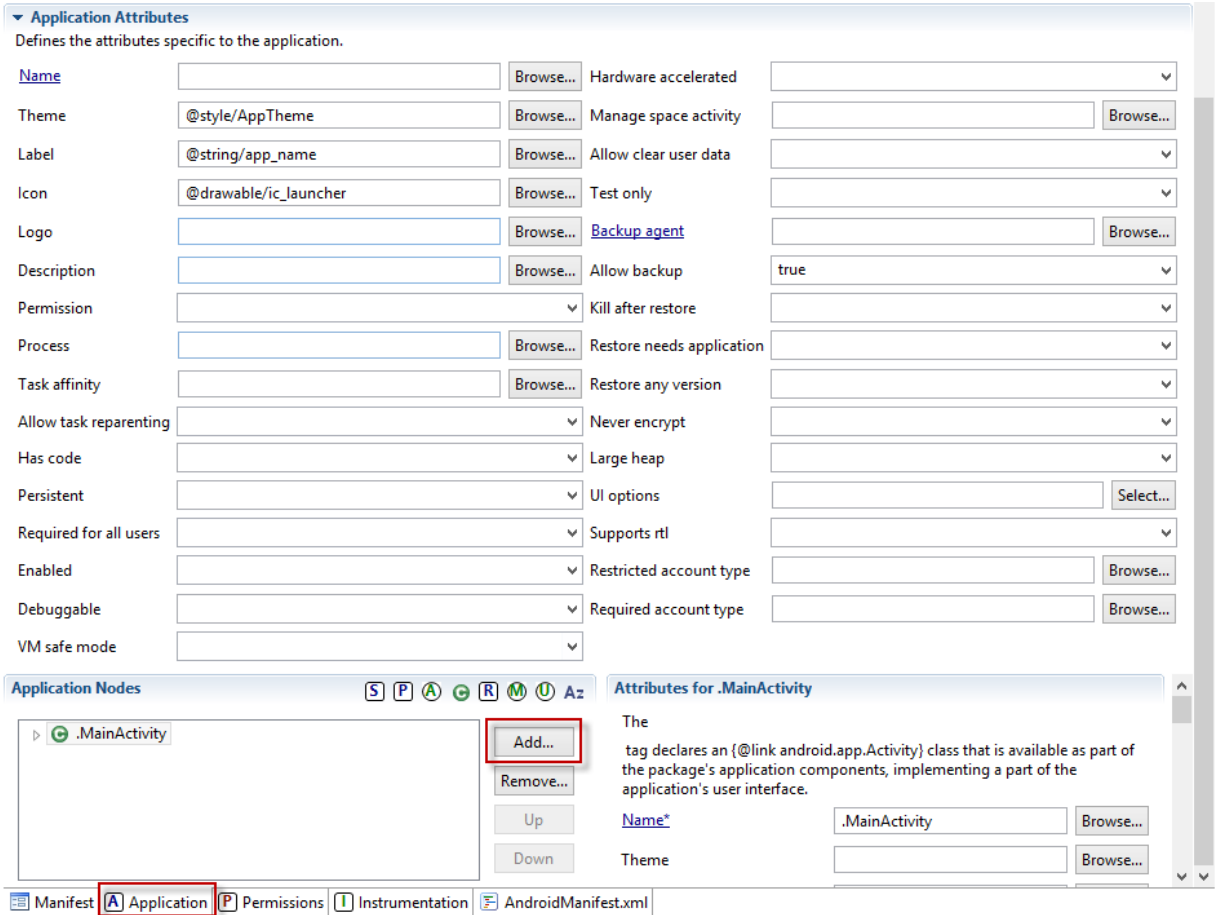
- Service ràng buộc (bound service) khi một thành phần ứng dụng ràng buộc Service bằng lời gọi **bindService()**
- Service ràng buộc có giao diện client-server cho phép các thành phần ứng dụng tương tác với Service, gửi yêu cầu, nhận kết quả
- Một Service ràng buộc sẽ chạy cho đến khi vẫn còn thành phần ứng dụng ràng buộc với Service
- Nhiều thành phần có thể cùng ràng buộc Service, khi tất cả thành phần không ràng buộc Service nữa, Service sẽ bị hủy

Bound Service

- Một Service có thể hoạt động theo cả 2 cách: vừa được khởi tạo (chạy vô hạn) và vừa cho phép ràng buộc (binding). Khi đó, bạn chỉ cần miêu tả 2 phương thức **onStartCommand()** cho phép thành phần khởi tạo nó và **onBind()** cho phép ràng buộc nó
- Dù ứng dụng của bạn có được khởi tạo, ràng buộc hoặc vừa khởi tạo và ràng buộc, bất kỳ thành phần ứng dụng nào cũng có thể sử dụng Service (thậm chí từ một ứng dụng khác) giống như cách mà bất kỳ thành phần nào cũng có thể sử dụng một acvitity bằng cách khởi tạo nó sử dụng Intent
- Tuy nhiên bạn có thể khai báo Service là private trong file AndroidManifest.xml (khai báo **android:exported="false**"

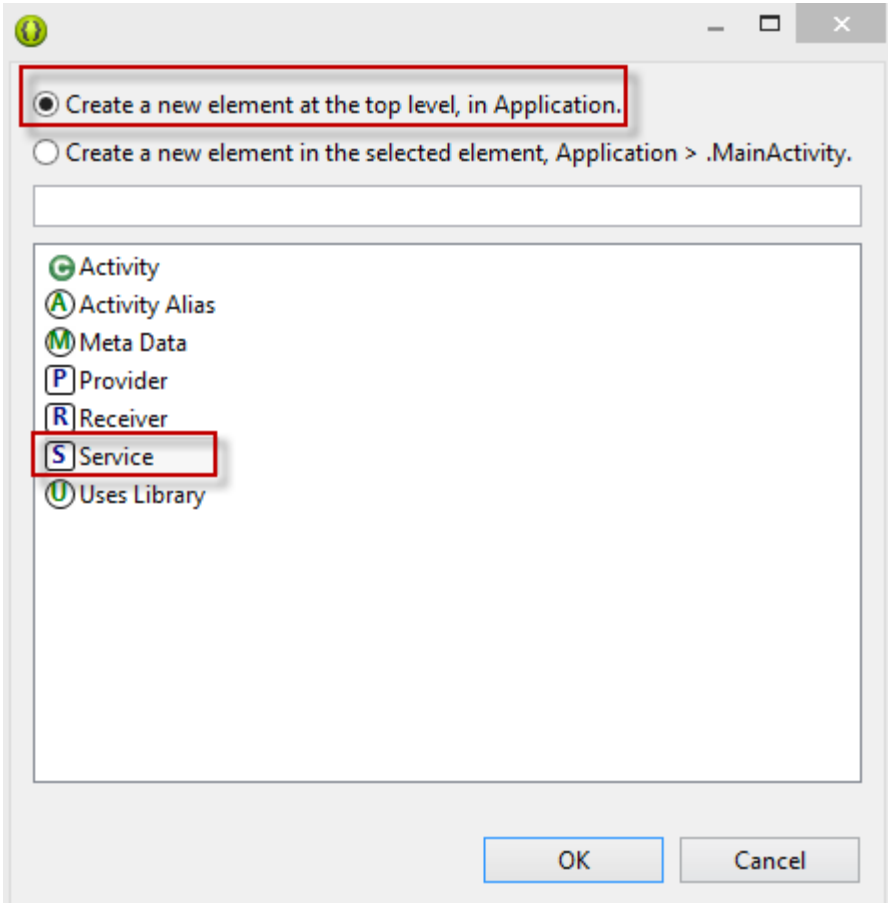
Đăng ký Service trong AndroidManifest.xml

- Bước 1: Mở file AndroidManifest.xml chọn "Application" tab
- Bước 2: Click nút Add ở mục Application Node



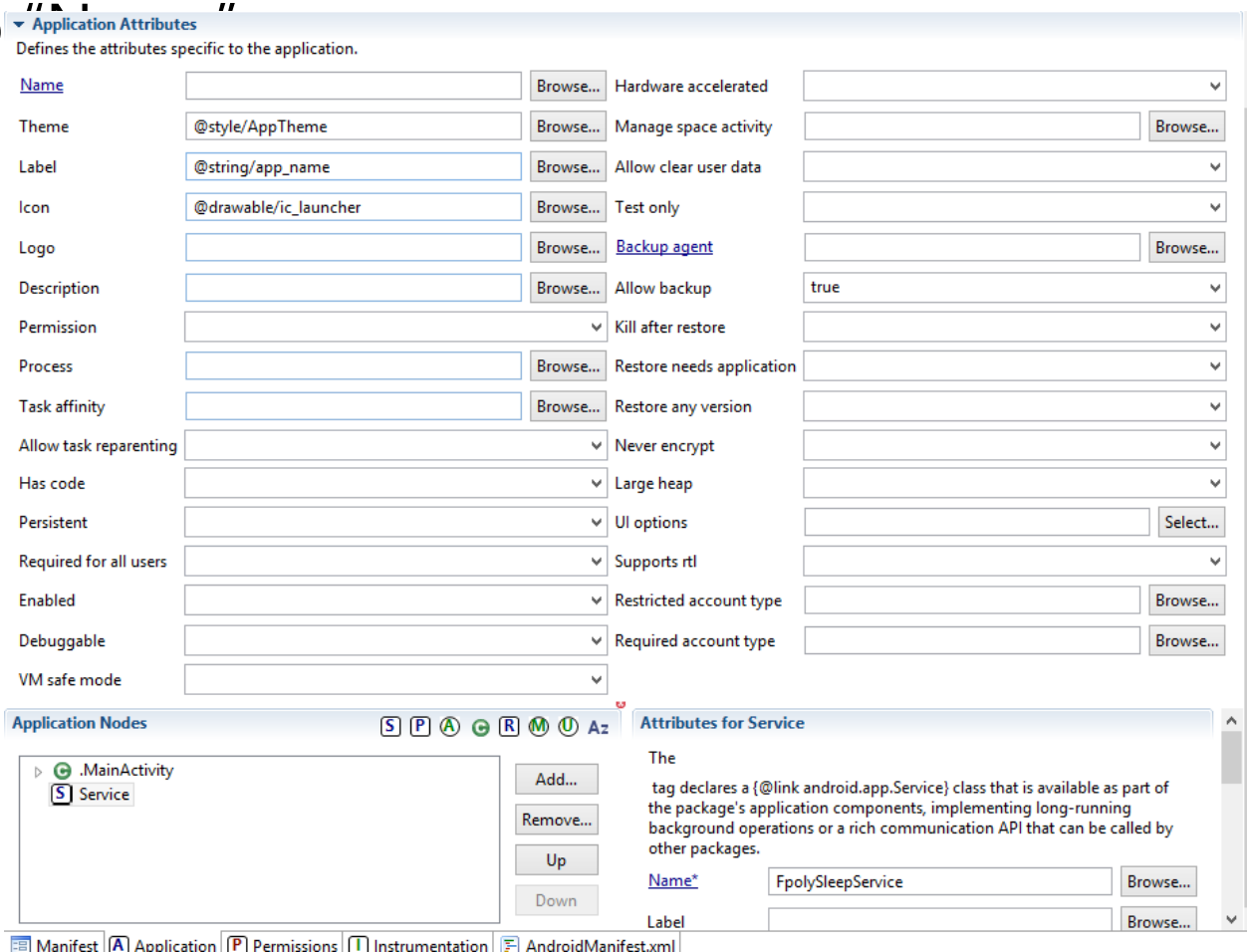
Đăng ký Service trong AndroidManifest.xml

- Bước 3: Chọn "Service" rồi nhấn nút "OK"



Đăng ký Service trong AndroidManifest.xml

- Bước 4: Kiểm tra lại bạn đang chọn "Services" trong "Application Nodes". Nhập tên của service bạn muốn đăng ký vào "



Đăng ký Service trong AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.fpolyintentservicesample"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name="com.example.fpolyintentservicesample.FpolySleepService">
        </service>
    </application>
</manifest>
```

Khởi tạo Service

- Tạo đối tượng Intent
- Sử dụng phương thức **startService** để khởi tạo Service
- Hủy Service bằng cách gọi phương thức **stopService()** hoặc **stopSelf()**

```
Intent intent = new Intent(this, TestService.class);
startService(intent);
```

DEMO

Màn hình có một nút Khởi tạo Service,
một nút Dừng Service

Bind Service

- Ràng buộc Service khi thành phần ứng dụng gọi **bindService()**
- UnBind khi gọi **unbindService()**
- Cung cấp **IBinder** để tương tác với thành phần khác
- Khi không còn ràng buộc, service sẽ bị hủy

Các phương thức callback

- onStartCommand()
- onBind()
- onCreate()
- onDestroy()

onStartCommand()

- Khi Service được tạo bằng cách gọi phương thức **startService**, **onStartCommand** sẽ được gọi một cách tự động
- Service chạy liên tục không phụ thuộc vào thành phần gọi nó
- Service khi hoàn thành nhiệm vụ sẽ dừng bằng lời gọi phương thức **stopSelf()** hoặc thành phần khác có thể dừng Service bằng lời gọi **stopService()**

onStartCommand()

- Phương thức này trả lại Integer miêu tả rằng chương trình sẽ tiếp tục như thế nào khi Service bị kill. Có thể có các giá trị như sau:
  - **START_STICKY**: Nếu hệ thống kill Service sau khi lời gọi phương thức onStartCommand trả lại giá trị, không tạo lại Service, trừ khi có pending intent cần gửi
  - **START_NOT_STICKY**: Nếu hệ thống kill Service sau khi lời gọi phương thức onStartCommand trả lại giá trị, tạo lại Service và gọi onStartCommand, nhưng không gửi lại Intent. Hệ thống sẽ gọi onStartCommand với null intent, trừ khi có pending intent khởi tạo Service

onStartCommand()

- Phương thức này trả lại Integer miêu tả sẽ tiếp tục như thế nào nếu Service bị kill. Có thể có các giá trị như sau:
  - **START_REDELIVER_INTENT:** Nếu hệ thống kill Service sau khi lời gọi phương thức onStartCommand trả lại giá trị, tạo lại Service và gọi onStartCommand() sử dụng intent gần nhất được gửi tới Service. Tất cả pending intent lần lượt được gửi. Thích hợp cho các Service thực hiện các công việc đòi hỏi khôi phục ngay lập tức, như download một file

onBind()

- Để cung cấp ràng buộc (bind) cho một Service, bạn phải miêu tả phương thức callback onBind()
- Phương thức này giúp ràng buộc Service với các thành phần ứng dụng khác bằng **ServiceConnection**
- Phương thức này trả lại đối tượng IBinder định nghĩa programming interface giúp cho client có thể tương tác với Service.
- Trả lại null nếu không cho phép binding

ServiceConnection

- Interface cho phép giám sát trạng thái của một service ứng dụng.
- Giống như các callback của hệ thống, các phương thức của lớp này được gọi từ main thread của tiến trình (process) của bạn
- Bao gồm 2 phương thức:
  - **onServiceConnected() (ComponentName name, IBinder service)**: gọi khi một kết nối tới Serevice được thiết lập
  - **onServiceDisConnected (ComponentName name)**: gọi khi một kết nối tới Service bị mất liên lạc

onCreate

- Phương thức này được gọi bởi hệ thống khi Service được tạo lần đầu tiên
- Không được gọi nếu Service đang chạy
- Gọi trước **onStartCommand()** và **onBind()**
- Chú ý không được gọi phương thức này trực tiếp

onDestroy

- Phương thức được gọi bởi hệ thống khi Service không còn được sử dụng nữa và Service bị hủy
- Bạn nên giải phóng tài nguyên như thread, listener, receiver trong phương thức này
- Sau lời gọi phương thức này, sẽ không có thêm lời gọi nào tới Service nữa và Service thực sự đã không còn tồn tại
- Chú ý không được gọi phương thức này trực tiếp

Dừng Service

- Có các phương thức để dừng một Service như sau:
  - **stopService(Intent service):** yêu cầu dừng một Service ứng dụng cụ thể. Lời gọi này sẽ dừng ngay Service cho dù service được khởi tạo bao nhiêu lần đi nữa. Tham số truyền vào miêu tả Service muốn dừng
  - **stopSelf():** dừng Service nếu nó đã được khởi tạo trước đó. Tương tự như phương thức stopService() cho một Service cụ thể
  - **stopSelf(int startId)**: version cũ của **stopSelfResult (int startId)**, startId là start identifier gần nhất nhận được bởi lời gọi onStart(Intent, int)

Code ví dụ

```
public class MyService extends Service {
    public MyService() {
    }
    @Override
    public IBinder onBind(Intent intent) {
        throw new UnsupportedOperationException("Not yet implemented");
    }
    @Override
    public void onCreate() {
        Toast.makeText(this, "The new Service was Created", Toast.LENGTH_LONG).show();

    }
     @Override
    public void onStart(Intent intent, int startId) {
        Toast.makeText(this, " Service Started", Toast.LENGTH_LONG).show();
    }

    @Override
    public void onDestroy() {
        Toast.makeText(this, "Service Destroyed", Toast.LENGTH_LONG).show();

    }
}
```

INTENT SERVICE

- Kế thừa Service
- Tự động sinh worker thread
- Sinh work queue để tránh xử lý quá nhiều công việc ở main thread của ứng dụng
- onBind() mặc định trả lại null
- onStartCommand() mặc định gọi **onHandlerIntent()**
- Để sử dụng Intent Service, kế thừa Intent Service và miêu tả **onHandlerIntent(intent)**. Intent Service sẽ nhận Intent, khởi tạo worker thread và dừng service khi yêu cầu của Service đã được thực hiện xong

Code ví dụ

```
import android.app.IntentService;

public class MyIntentService extends IntentService {
    private static final String TAG = "com.example.fpolytestservice";

    @Override
    protected void onHandleIntent(Intent arg0) {
        // TODO Auto-generated method stub
        Log.i(TAG, "Intent Service started");

    }
    public MyIntentService() {
            super("MyIntentService");
    }
}
```

DEMO

Sử dụng Intent Service

Gửi thông báo đến người dùng

- Thông thường chúng ta sẽ gửi thông báo đến người dùng khi Service thực hiện xong công việc
- Có hai loại thông báo chính:
  - **Toast notification**: một message xuất hiện trên cửa sổ hiện tại của ứng dụng
  - **Status Bar notification**: cung cấp một icon trên thanh trạng thái (status bar) cùng với một thông điệp, khi đó người dùng có thể đưa ra một hành động (ví dụ như khởi tạo một Acvitity)
- Status bar notification là kỹ thuật tốt nhất khi một công việc chạy nền kết thúc (như kết thúc tải một file và người dùng có thể thao tác trên file đó). Khi người dùng chọn notification từ view mở rộng, notification sẽ khởi tạo một

Gửi thông báo đến người dùng

Toast notification

Status Bar notification

TỔNG KẾT NỘI DUNG BÀI HỌC

- Phần I: Service và vòng đời Service
  - Tổng quan về Service
  - Quản lý vòng đời Service
- Phần II: Sử dụng Service
  - Tạo Service
  - Dừng Service


TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
LUẬT KINH TẾ
LUẬT KINH TẾ
KỸ THUẬT SOẠN THẢO VÀ TRÌNH BÀY VĂN BẢN
TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
Cảm ơn