LẬP TRÌNH ANDROID

Giảng Viên: Phùng Mạnh Dương

Trường ĐH Công nghệ, ĐHQGHN

CHƯƠNG 7
Alarm và Service

- Alarm
  - Khái niệm
  - AlarmManager APIs
  - Các loại Alarm
  - Ứng dụng ví dụ
- Service
  - Tổng quan về dịch vụ
  - Cài đặt started service
  - Cài đặt bound service

Alarm

- Alarm là cơ chế để gửi Intent tại một thời điểm trong tương lai.
- Alarm cho phép 1 ứng dụng thực thi code ngay cả khi ứng dụng đó đang không chạy.
- Một số đặc điểm:
  - Khi được đăng ký, Alarm vẫn active nếu thiết bị ở trạng thái ngủ (tắt màn hình).
  - Alarm bị hủy khi thiết bị tắt hoặc khởi động lại.
  - Có thể sử dụng kết hợp với broadcast receiver để khởi phát service hay các tác vụ khác.
  - Giúp giảm tài nguyên cần thiết cho ứng dụng.

VD về Alarm:

- Ứng dụng báo thức.
- Ứng dụng định kỳ cập nhật thông tin thời tiết qua mạng.
- Ứng dụng MMS sẽ thử gửi lại tin nhắn sau 1 khoảng thời gian nhất định.
- Ứng dụng BlueTooth sẽ ngừng tìm kiếm thiết bị sau 1 khoảng thời gian nhất định.
- Ứng dụng gọi điện sẽ lưu thông tin người dùng (cache) và định kỳ cập nhật thông tin.

AlarmManager

- Để tạo và quản lý alarm, sử dụng lớp AlarmManager.
- Để lấy tham chiếu tới AlarmManager, sử dụng hàm sau của lớp Context:

getSystemService(Context.ALARM_SERVICE)

Tạo Alarm

```
// Alarm 1 lần
void set(int type, long triggerAtTime,
        PendingIntent operation)

// Alarm lặp lại
void setRepeating(int type, long triggerAtTime,
        long interval, PendingIntent operation)
```

Tạo Alarm

```
// Tạo Alarm với thời gian kích hoạt không chính
// xác (khuyên dùng nếu có thể)
void setInexactRepeating(int type,
                         long triggerAtTime,
                         long interval,
                         PendingIntent operation)
```

Các lựa chọn khoảng thời gian:

- INTERVAL_FIFTEEN_MINUTES
- INTERVAL_HALF_HOUR
- INTERVAL_HOUR
- INTERVAL_HALF_DAY
- INTERVAL_DAY

Kiểu Alarm

- 2 mức độ cấu hình Alarm:
  - Thiết lập thời gian như thế nào
  - Làm gì khi xảy ra alarm lúc thiết bị đang ngủ (tắt màn hình).
- Thiết lập thời gian:
  - Thời gian thực: thời gian theo đồng hồ hệ thống (VD: 12:00)
  - Thời gian trôi qua: VD 30 phút từ thời điểm đặt alarm
- Thiết đặt khi thiết bị ngủ:
  - Đánh thức thiết bị và gửi Intent
  - Đợi tới khi thiết bị thức để gửi Intent.

Định nghĩa kiểu Alarm

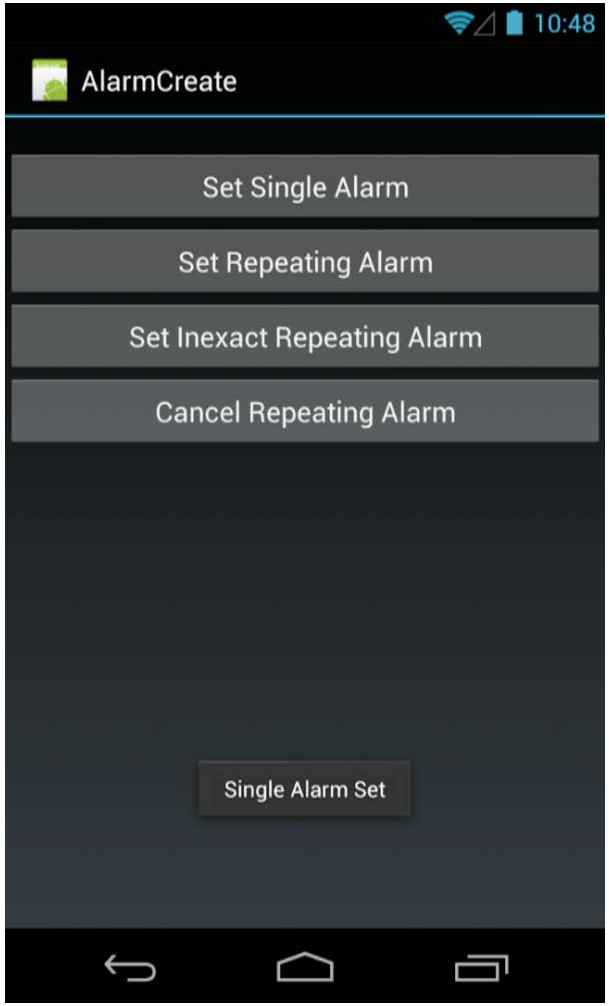
- RTC_WAKEUP: Thời gian thực & đánh thức thiết bị
- RTC: Thời gian thực & không đánh thức thiết bị
- ELAPSED_REALTIME: Thời gian trôi qua & không đánh thức thiết bị
- ELAPSED_REALTIME_WAKEUP: Thời gian trôi qua & đánh thức thiết bị
- Khuyên dùng: ELAPSED_REALTIME & ELAPSED_REALTIME_WAKEUP

Đối số PendingIntent

- PendingIntent là gì?
- 3 hàm để tạo PendingIntent:
  - getActivity(Context context, int requestCode, Intent intent, int flags, Bundle options): Tạo PendingIntent để khởi phát Activity
  - getBroadcast(Context context, int requestCode, Intent intent, int flags): Tạo PendingIntent để Broadcast Intent
  - getService(Context context, int requestCode, Intent intent, int flags): Tạo PendingIntent để khởi phát service
- Đọc thêm về Alarm:

  http://developer.android.com/intl/vi/training/scheduling/alarms.html

Ứng dụng ví dụ: AlarmCreate



Sinh viên tự tìm hiểu

- Thiết đặt lại Alarm ngay khi thiết bị khởi động

http://developer.android.com/intl/vi/training/scheduling/alarms.html#boot

Tổng quan về dịch vụ

- Service là một ứng dụng trong Android mà chạy trong background và không cần tương tác với người dùng.
- Mục đích:
  - Thực hiện các tác vụ cần xử lý lâu trong background. VD: Download file, chơi nhạc.
  - Cho phép các tiến trình (hoặc ứng dụng) tương tác với nhau. VD: đồng bộ dữ liệu giữa các ứng dụng.
- Tương ứng có 2 kiểu dịch vụ:
  - Dịch vụ Khởi phát (Started Service).
  - Dịch vụ Ràng buộc (Bound Service).

http://developer.android.com/guide/components/services.html#Basics

Dịch vụ khởi phát (Started Service)

- Một dịch vụ là "started" khi một thành phần ứng dụng (như Activity) khởi phát nó bằng startService().
- Khi đã khởi phát, dịch vụ có thể chạy mãi mãi trong background, thậm chí cả khi ứng dụng khởi phát nó đã kết thúc.
- Thông thường, kiểu dịch vụ "khởi phát" thực hiện 1 tác vụ đơn lẻ rồi kết thúc mà không trả kết quả về cho bên gọi nó.
- Mặc định, dịch vụ sẽ chạy trong luồng chính của ứng dụng chứa nó. Vì vậy, có thể sẽ cần tạo luồng riêng cho service.

Dịch vụ ràng buộc (Bound Service)

- Một dịch vụ là "ràng buộc" khi một thành phần ứng dụng gắn (bind) tới nó bằng bindService().
- Dịch vụ này cung cấp 1 interface kiểu client-server để cho phép ứng dụng tương tác với dịch vụ như gửi yêu cầu, nhận kết quả và thậm chí giao tiếp liên tiến trình (IPC).
- Tại thời điểm bind, dịch vụ sẽ được khởi phát nếu nó chưa khởi phát.
- Dịch vụ sẽ tiếp tục chạy chừng nào còn ít nhất 1 client gắn tới nó.

Dịch vụ khởi phát (Started Service)

- Tạo service:
  - Kế thừa lớp Service
  - Override các phương thức:

  onStartCommand(Intent intent, int flags, int startId)

  onDestroy()

  onBind(Intent intent)
- Gọi service: Sử dụng các phương thức

  startService(Intent service)

  stopService(Intent name)
- VD:
  - SimpleStartedService
  - MusicPlayingServiceExample

IntentService

- Một số vấn đề cần lưu ý khi sử dụng lớp Service:
  - Mặc định, dịch vụ sẽ chạy trong luồng chính của ứng dụng chứa nó. Vì vậy, service có thể khiến ứng dụng bị treo nếu thực hiện các tác vụ tốn thời gian.
  - Sử dụng method stopSelf() thường hay bị quên.

-> Vì thế để đơn giản sử dụng công việc bất đồng bộ, ta có thể sử dụng lớp IntentService.

- Lớp IntentService được sử dụng để xử lý các yêu cầu asynchronous. Nó được khởi chạy như một service bình thường, và nó thực hiện công việc bên trong một worker class và tự hủy khi công việc được hoàn thành.
- VD: LoggingServiceClient

Dịch vụ ràng buộc (Bound Service)

- Làm sao để giao tiếp giữa activity và service:
  - BroadcastReceiver
  - Intent.putExtra
- Cách tốt hơn: Gắn trực tiếp Activity vào Service (thông qua kết nối kiểu Client – Server).
- Có 2 kiểu bind:
  - Trong nội bộ ứng dụng: Sử dụng Binder Class
  - Giữa các ứng dụng: Sử dụng Messenger

http://developer.android.com/guide/components/bound-services.html

Sử dụng Binder Class

- Lớp Binder cho phép ứng dụng Client có thể truy cập vào các biến và phương thức public của Service.
- Các bước cài đặt:
  - Trong lớp dịch vụ, tạo một thể hiện của lớp Binder bao gồm:
    - Các phương thức public để client có thể gọi.
    - Trả về tham chiếu tới dịch vụ.
  - Trong hàm onBind() của dịch vụ, trả về thể hiện của Binder vừa tạo.
  - Ở phía client, lấy tham chiếu tới Binder của dịch vụ thông qua phương thức onServiceConnected() rồi dùng tham chiếu đó để tương tác với dịch vụ.
- VD: Chương 11 giáo trình, mục "BINDING ACTIVITIES TO SERVICES"

Sử dụng lớp Messenger

- Phía dịch vụ cài đặt Handler để xử lý thông điệp nhận được từ client.
- Handler đó được sử dụng để tạo 1 đối tượng Messenger.
- Messenger này tạo ra 1 IBinder mà phía dịch vụ sẽ gửi cho client khi kết nối.
- Client sử dụng IBinder này để khởi tạo 1 Messenger mà thông qua đó truyền thông điệp tới phía dịch vụ.

- VD: LoggingServiceWithMessenger

Câu hỏi & Bài tập

- Bài tập: Tự học chương 11 giáo trình