Android Developer Fundamentals V2

Alarms and Schedulers

Lesson 8

8.2 Alarms

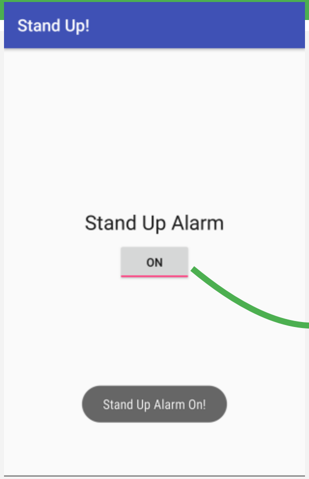
Contents

What Are Alarms

What is an alarm in Android?

- Not an actual alarm clock.
- Schedules something to happen at a set time.
- Fire intents at set times or intervals.
- Goes off once or recurring.
- Can be based on a real-time clock or elapsed time.
- App does not need to run for alarm to be active.

How alarms work with components

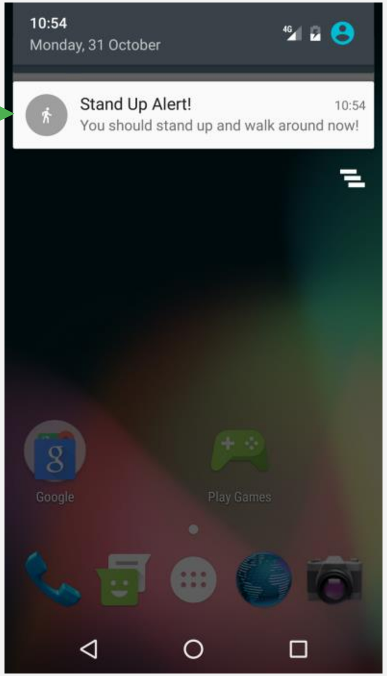


Activity creates a notification and sets an alarm.

Alarm triggers and sends out Intent.

App may be destroyed so….

BroadcastReceiver wakes up the app and delivers the notification.



Benefits of alarms

- App does not need to run for alarm to be active.
- Device does not have to be awake.
- Does not use resources until it goes off.
- Use with `BroadcastReceiver` to start services and other operations.

Measuring time

- Elapsed Real Time—time since system boot.
  - Independent of time zone and locale.
  - Use for intervals and relative time.
  - Use whenever possible.
  - Elapsed time includes time device was asleep.

- Real Time Clock (RTC)—UTC (wall clock) time.
  - When time of day at locale matter.

Wakeup behavior

- Wakes up device CPU if screen is off.
    - Use only for time critical operations.
    - Can drain battery.

- Does not wake up device.
    - Fires next time device is awake.
    - Is polite.

Types of alarms

	Elapsed Real Time (ERT)—since system boot	**Real Time Clock (RTC)**—time of day matters
Do not wake up device	`ELAPSED_REALTIME`	`RTC`
Wake up	`ELAPSED_REALTIME_WAKEUP`	`RTC_WAKEUP`

Alarms Best Practices

If everybody syncs at the same time...

Imagine an app with millions of users:

- Server sync operation based on clock time.
- Every instance of app syncs at 11:00 p.m.

➡ Load on the server could result in high latency or even "denial of service"

Alarm Best Practices

- Add randomness to network requests on alarms.
- Minimize alarm frequency.
- Use `ELAPSED_REALTIME`, not clock time, if you can.

Battery

- Minimize waking up the device.
- Use inexact alarms.
    - Android synchronizes multiple inexact repeating alarms and fires them at the same time.
    - Reduces the drain on the battery.
    - Use `setInexactRepeating()` instead of `setRepeating()`.

When not to use an alarm

- Ticks, timeouts, and while app is running—`Handler`.
- Server sync—`SyncAdapter` with Cloud Messaging Service.
- Inexact time and resource efficiency—`JobScheduler`.

AlarmManager

What is AlarmManager

- `AlarmManager` provides access to system alarm services.
- Schedules future operation.
- When alarm goes off, registered `Intent` is broadcast.
- Alarms are retained while device is asleep.
- Firing alarms can wake device.

Get an AlarmManager

```
AlarmManager alarmManager =
    (AlarmManager) getSystemService(ALARM_SERVICE);
```

Scheduling Alarms

What you need to to schedule an alarm

1. Type of alarm.
2. Time to trigger.
3. Interval for repeating alarms.
4. `PendingIntent` to deliver at the specified time (just like notifications).

Schedule a single alarm

- `set()`—single, inexact alarm.
- `setWindow()`—single inexact alarm in window of time.
- `setExact()`—single exact alarm.

More power saving options `AlarmManager` API 23+.

Schedule a repeating alarm

- `setInexactRepeating()`
  - repeating, inexact alarm.
- `setRepeating()`
  - Prior to API 19, creates a repeating, exact alarm.
  - After API 19, same as `setInexactRepeating()`.

setInexactRepeating()

```
setInexactRepeating(
    int alarmType,
    long triggerAtMillis,
    long intervalMillis,
    PendingIntent operation)
```

Create an inexact alarm

```
alarmManager.setInexactRepeating(
    AlarmManager.ELAPSED_REALTIME_WAKEUP,
       SystemClock.elapsedRealtime()
          + AlarmManager.INTERVAL_FIFTEEN_MINUTES,
       AlarmManager.INTERVAL_FIFTEEN_MINUTES,
       notifyPendingIntent);
```

More Alarm Considerations

Checking for an existing alarm

```
boolean alarmExists =
 (PendingIntent.getBroadcast(this,
       0, notifyIntent,
       PendingIntent.FLAG_NO_CREATE) != null);
```

Doze and Standby

- Doze—completely stationary, unplugged, and idle device.
- Standby—unplugged device on idle apps.
- Alarms will not fire.
- API 23+.

User visible alarms

- `setAlarmClock()`
- System UI may display time/icon.
- Precise.
- Works when device is idle.
- App can retrieve next alarm with `getNextAlarmClock()`.
- API 21+.

Cancel an alarm

- Call `cancel()` on the `AlarmManager`
  - pass in the `PendingIntent.`

```
alarmManager.cancel(alarmPendingIntent);
```

Alarms and Reboots

- Alarms are cleared when device is off or rebooted.
- Use a `BroadcastReceiver` registered for the `BOOT_COMPLETED` event and set the alarm in the `onReceive()` method.

Learn more

- Schedule Repeating Alarms Guide
- `AlarmManager` reference
- Choosing an Alarm Blog Post
- Scheduling Alarms Presentation
- Optimizing for Doze and Standby

What's Next?

- Concept Chapter: 8.2 Alarms
- Practical: 8.2 The Alarm Manager

END