DOING PERIODIC WORK

ANOTHER ANDROID HEADACHE

CommonsWare

SCENARIOS FOR PERIODIC WORK

- In the UI (e.g., remaining-time counter): use `postDelayed()`
- In a running service: use `ScheduledExecutorService`
- Independent of a running app (e.g., server sync)

OPTIONS FOR INDEPENDENT PERIODIC WORK

- Everlasting `Service`, coupled with a `WakeLock` and `ScheduledExecutorService`...**THIS IS TEH EVILZ**
- `AlarmManager`: `setAlarmClock()`, other `set...()` methods
- `JobScheduler`
- `GcmNetworkManager` (quasi-backport of `JobScheduler`, requires Play Services)
- Wrapper libraries around the above

`AlarmManager` OPTIONS

- Wake Up, or Not?
- Repeating, or Not?
- Inexact, or Not?
- Absolute Time, or Not?
- What Happens

AlarmManager ALARM TYPES

- RTC
- RTC_WAKEUP
- ELAPSED_REALTIME
- ELAPSED_REALTIME_WAKEUP

`AlarmManager`: ORIGINAL RECIPE

- Get an `AlarmManager` via `getSystemService()`
- Create a `PendingIntent` to indicate what should happen
- Configure alarm using `AlarmManager`
  - `set()`
  - `setRepeating()`
  - `setInexactRepeating()`

`AlarmManager`: API 19 METHODS

- `setExact()`
- `set()` is now inexact
- `setWindow()`
- `setInexactRepeating()` (as `setRepeating()` is inexact)

`AlarmManager`: EARLY TWENTIES

- API 22: cannot have a period under one minute (undocumented)
- API 23: `set...AndAllowWhileIdle()`
- API 24: cannot have custom `Parcelable`

`JobScheduler`

SMARTER THAN YOUR AVERAGE `AlarmManager`

- Intrinsically inexact
- Criteria along with time
  - Are we on a charger?
  - Do we have an Internet connection? How about WiFi?
  - Is the device "idle"?
- `WakeLock`, restore-on-reboot work managed for you

`JobScheduler` RECIPE

- Create a `JobService`, implementing `onStartJob()` and `onStopJob()`
- Create a `JobInfo.Builder` and configure
- Get a `JobScheduler`, use it to schedule or cancel the job from the `JobInfo`

`JobScheduler` ON ANDROID 7.0

CONTENT MONITORING

- Register a `Uri`
- Have a job invoked when that content changes
- Like a `ContentObserver`, minus the everlasting service bit
- Example: watching for changes in contacts
- Likely to be a future pattern for eliminating system broadcasts

GcmNetworkManager

- More or less JobSchedulerCompat
- Part of the Play Services SDK, not available for all Android devices

WRAPPER LIBRARIES

- Firebase Job Dispatcher
  - From Google
  - Pluggable drivers
  - Limited signs of life
- Evernote's `android-job`
  - Single API, using `JobScheduler` where possible and `AlarmManager` elsewhere
  - Opt-in for `GcmNetworkManager` option

DOZE MODE

- Full
  - Device has not moved, screen is off, and is not charging for 1+ hours
  - Background processing becomes infrequent ...as in "once every few hours"
- "On the Go"
  - Device's screen is off and is not charging for "a short time"

APP STANDBY

- Device is not charging
- User has not been in your app for quite some time
  - No activities
  - No foreground service or other notification
- Background processing becomes infrequent ...as in "once every day"

SPECIFIC IMPACTS

FULL DOZE AND APP STANDBY

- No `AlarmManager` alarms
- No `JobScheduler` jobs
- No `SyncManager` syncs
- Outstanding partial `WakeLocks` ignored
  - Exception: foreground service
  - Exception to the exception: on Android 6.0 needs to be separate process
- No network access

SPECIFIC IMPACTS

"DOZE ON THE GO" IN ANDROID 7.0

- No `JobScheduler` jobs
- No `SyncManager` syncs
- No network access

WORKAROUNDS

- User whitelist in Settings ...just don't try asking the user to do it
- `AlarmManager` and `setAndAllowWhileIdle()` / `setExactAndAllowWhileIdle()`
- GCM ...which doesn't work if the device is offline, and requires Play Services

WHY?

> *I need to send a message to the server... once a minute... for 2-3 hours. To achieve this I am currently using AsyncTasks (for sending the request), launched once a minute via a foreground Service, which is scheduled with alarmManagers' setExactAndAllowWhileIdle. The foreground service does have a partial wakelock too!*

(posted April 7, 2016)

THIS IS WHY WE CAN'T HAVE NICE THINGS

SPECIAL SNOWFLAKE SYNDROME

- Too many developers being wasteful with the battery
- Too few developers honored the API 19 nudges
- Too many users complaining about horrible battery life

SO, NOW WHAT?

- Design and Market for Unreliability
  - Tout: manual refresh options
  - Downplay: automatic refresh
  - Don't offer: priority alerts
- Use GCM... if you can live with the limitations
- Use `JobScheduler` or `android-job`
- Cry in your beer