G53MDP
Mobile Device Programming

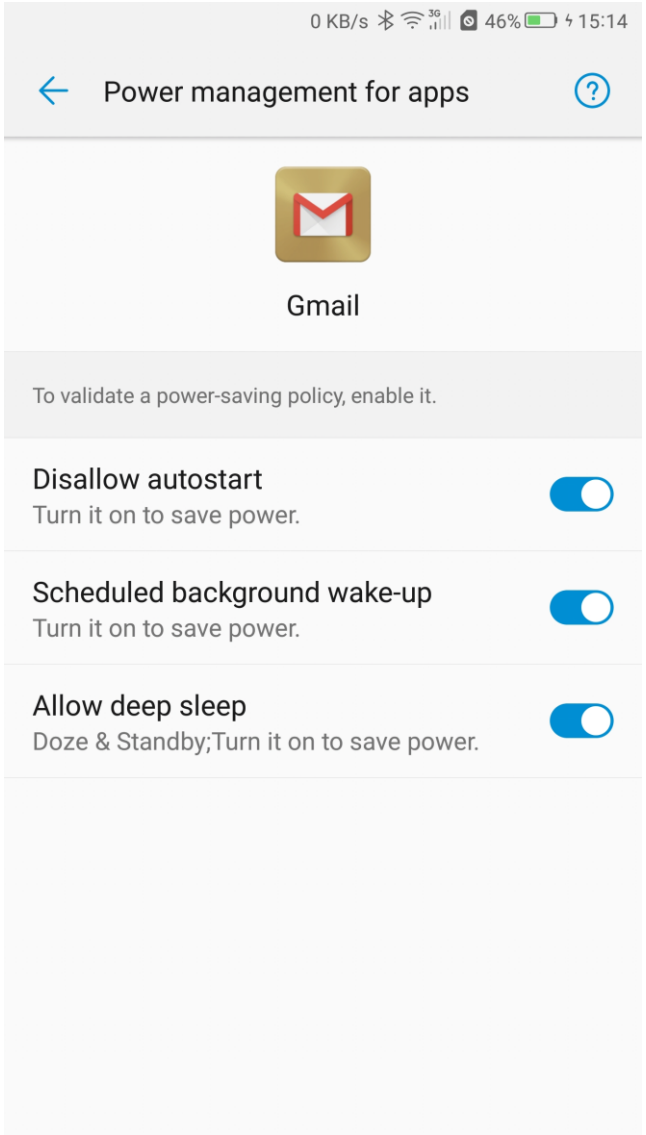
Lecture 18 – Power (2)

Learning Outcomes

- Knowledge about different modes for battery life enhancement in Android (App Standby & Doze)
- Knowledge about firebase cloud messaging
- Effective use of AlarmManager & JobScheduler

Battery Life Enhancement in Android (Android 6.0 onwards)

- **App Standby:** defer background activity for apps with no recent user interaction.
- **Doze**: deep sleep if user has not actively used the device for extended periods of time
- **Exemptions:** system apps and cloud messaging services preloaded on phone are exempted from App Standby & Doze.
- Test apps in App standby & Doze mode for the desired performance

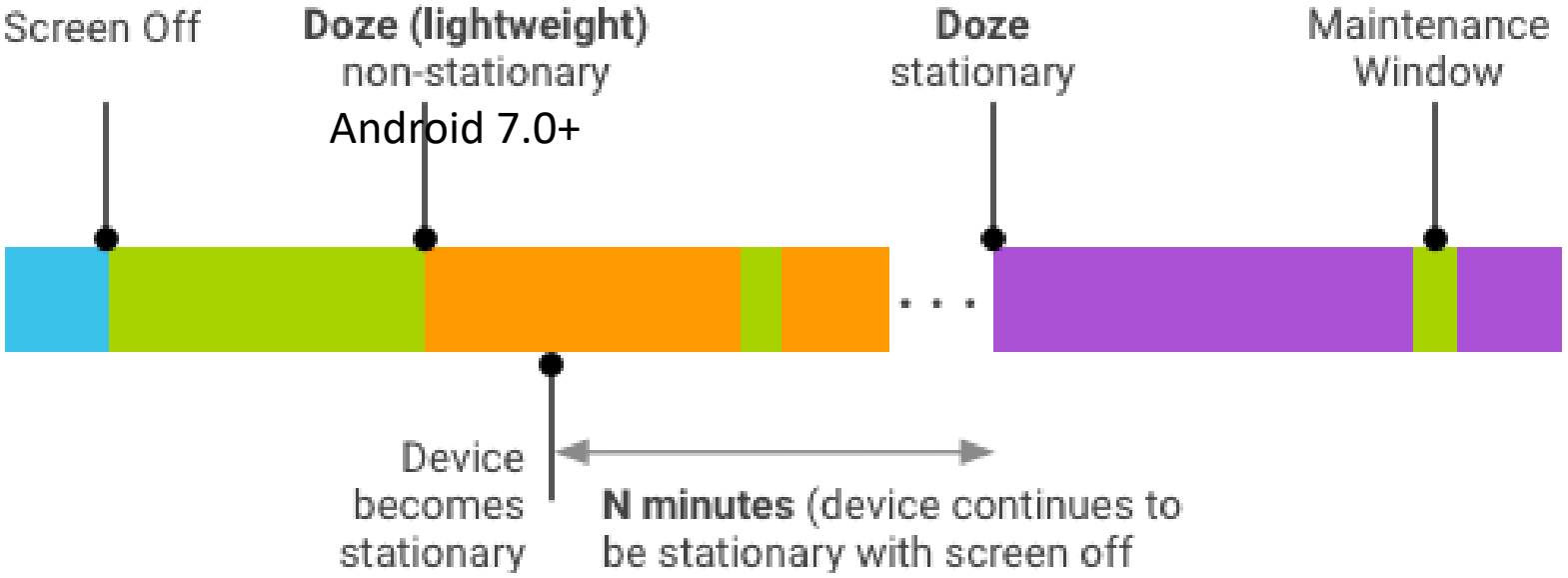


App Standby

- **Start conditions:** an app is not actively used for a certain time will be placed in an idle mode
  - Not doing foreground work (activities/service, pending notifications)
  - Hasn't been explicitly launched for certain time (days).
- **Actions:** No network access, no background jobs, can set Alarms, can use wake locks, allow network access once per day
- **Exists conditions:** plug in the device, do some foreground tasks

Doze

- **Starts Doze:** when device is in idle - screen off, on battery & stationary.
- **In Doze**: no network access; no CPU-intensive work; wakelocks ignored; no wifi scan; deferred alarm manager alarms; only high priority notification received; no job scheduler; no sync adapters.
- **Exits Doze:** user interaction, device motion, screen on, or AlarmClock alarm. Notification do not cause Doze exits.
- **Maintenance window:** complete pending activities (syncs, jobs, etc).
- MMS/SMS/Telephony services are **excluded** from Doze



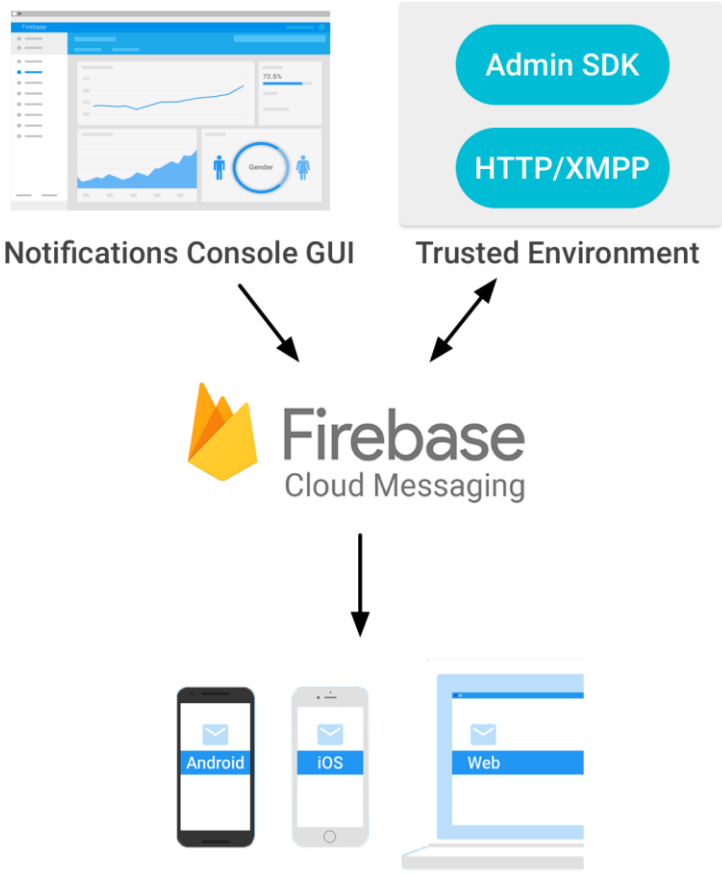
https://source.android.com/devices/tech/power/mgmt

Exemptions

- System apps and cloud messaging services preloaded on phone are exempted from App Standby & Doze.
- Use whitelist for apps to be partially exempt from Doze & App Standby
  - isIgnoringBatteryOptimiztions() to check if in the whitelist;
  - ACTION_IGNORE_BATTERY_OPTIMIZATION_SETTINGS intent to direct user to battery optimization options
  - REQUEST_IGNORE_BATTERY_OPTIMIZATIONS allow user to add app to whitelist directly
- Whitelisted apps can use network and hold partial wake locks during Doze & App Standby, but jobs & syncs are still deferred.
- App should not be on the whitelist unless can't use FCM high-priority messages or App's core function is affected

Firebase Cloud Messaging (FCM) (Replaced Google Cloud Messaging)

- A cross-platform reliable battery-efficient messaging solution that allows deliver message between server and client
- Provide single and persistent connection to the server
- Can notify a client app that new email or other data is available to sync.
- Can deliver to single device, groups of devices or devices subscribed to topics
- Send messages from client apps to server (e.g. chats, messages, etc.)



https://firebase.google.com/docs/cloud-messaging/

Whitelisted from Doze?

- Does the app need to run when screen off, stationary & on battery?
- Does the app can use FCM/FCM high priority messaging?
- Does the core functionality affected by Doze/Standby?
- Does the core functions need persistently requires a peripheral device for internet access?

- *WhatsApp*?
- The *activity tracker* for CW2?

Sustained Performance Mode (new API in Android 7.0)

- Thermal throttling prevents a long running apps maintain its performance (e.g. game, camera, virtual reality application etc.)
- App can request the platform to enter a SPM, and keep a consistent level of performance
  - Only tests on Android 7.0 on Nexus 6P devices
  - Set using Window.setSustainedPerformanceMode()
  - System automatically disables the mode when no longer in focus

Schedule Jobs in Android

- AlarmManager (outdated)
  - Based on time
- JobScheduler (preferred)
  - Based on conditions
  - Run on the main thread! Need to be simple or carefully use Thread or AsyncTask

What is Efficient Scheduling?

- Multiple scheduled alarms waking the phone
  - Suspending and waking the phone takes power
- More efficient to schedule multiple alarms at the same time
  - Wake once, do several tasks, sleep once
  - Wake, task, sleep, wake, task, sleep, wake…

AlarmManager

- Schedule a task run at a specific time point/ time interval
- **For normal time operations (i.e. ticks, timeout), it is better to use *Handler* rather than AlarmManager**
- AlarmManager holds a CPU wake lock, when alarm receiver's onReceive() is executing
- Android possibly go to sleep even when checking alarms before the requested service is launched.
  - Requires a separate wake lock to ensure the service is launched.
- Alarm delivery is inexact (Android 5.0+); use setWindow and setExact for exact delivery.

AlarmManager Usage

- Specify an Intent to be broadcast at some time
  - setRepeating(int type, long triggerAtMillis, long intervalMillis, PendingIntent operation)
  - setExact(int type, long triggerAtMillis, PendingIntent operation)
- Extend Broadcast Receiver
  - onReceive method is called when the alarm goes off
  - Start a service to actually do the work

WAKE_LOCK

- Request the WAKE_LOCK permission in the manifest.
  - <uses-permission android:name="android.permission.WAKE_LOCK"/>
- Use newWakeLock() to create a PowerManager.WakeLock object

```
PowerManager pm = (PowerManager) getSystemService(Context.POWER_SERVICE);
 PowerManager.WakeLock wl = pm.newWakeLock(PowerManager.SCREEN_DIM_WAKE_LOCK, "My Tag");
 wl.acquire();
   ..screen will stay on during this section..
 wl.release();
```

Flag Value	CPU	Screen	Keyboard
PARTIAL_WAKE_LOCK	On* (don't sleep even press power button)	Off	Off
SCREEN_DIM_WAKE_LOCK	On (sleep when press power button)	Dim	Off
SCREEN_BRIGHT_WAKE_LOCK	On (sleep when press power button)	Bright	Off
FULL_WAKE_LOCK	On (sleep when press power button)	Bright	Bright

WakefulBroadcastReceiver

- Broadcast receiver that handles PARTIAL_WAKE_LOCK in a Service
  - To ensure the device is not in sleep while transitioning the work to a service.
  - Add the receiver in manifest: <receiver android:name=“.MyWakefulReceiver”></receiver>
  - Use startWakefulService() to start an intent service

Demo

- Used Partial Wake Lock (two methods) to make sure the alarm broadcast receiver is awake when calling
- App frequently output information with inexact execution time

JobScheduler

- Many apps perform tasks asynchronously outside the main activity. (e.g download files, sync database to cloud etc.)
- Job Scheduler collects pending jobs across all apps, and schedule them to run at about the same time-> sleep for longer-> save power
- Specify requirements for network and timing for each job, then the JS robustly optimise the execution time.
- May defer jobs that comply with Doze & App Standby
- **JobService will run on the main thread**. Need to manage any asynchronous tasks yourself (e.g. Threads).

JobScheduler Usage

- Construct *JobInfo* objects using *JobInfo.Builder using schedule conditions*
  - *Network type (metered/unmetered)*
  - *Charging and Idle*
  - *Centent Provide update*
  - *Backoff (retry) criteria*
  - *Minimum latency and override dealine*
  - *Periodic*
  - *Persistent*
  - *Extras (customised information from app)*
- Pass it to JobScheduler using *schedule(JobInfo)*
- Jobs implemented in app's JobService
- Jobs will be executed (batch & defer jobs smartly) at certain time but not exact
- Different job can be in different JobServices.

JobScheduler Demo

Summary

- Battery and power consumption calculation
  - Calculate power cost by tacking time & use of power profile
- Android Power Management
  - Wake locks, early suspend/late resume
- App Standby & Doze
  - Start/exit conditions & mode behaviours
  - What apps should be exemption from them?
- Firebase cloud messaging
  - Reliable battery-efficient messaging solution that allows deliver message between server and client
- Alarm manager/Job scheduler
  - Used for inexact task execution. Job Scheduler is preferred for most cases
- **Do not over think the activity tracking coursework**

References

- https://developer.android.com/training/scheduling/wakelock.html
- https://developer.android.com/training/scheduling/alarms.html
- https://developer.android.com/reference/android/app/AlarmManager.html
- https://developer.android.com/topic/performance/scheduling.html