# Mobile Computing

## COMP5216

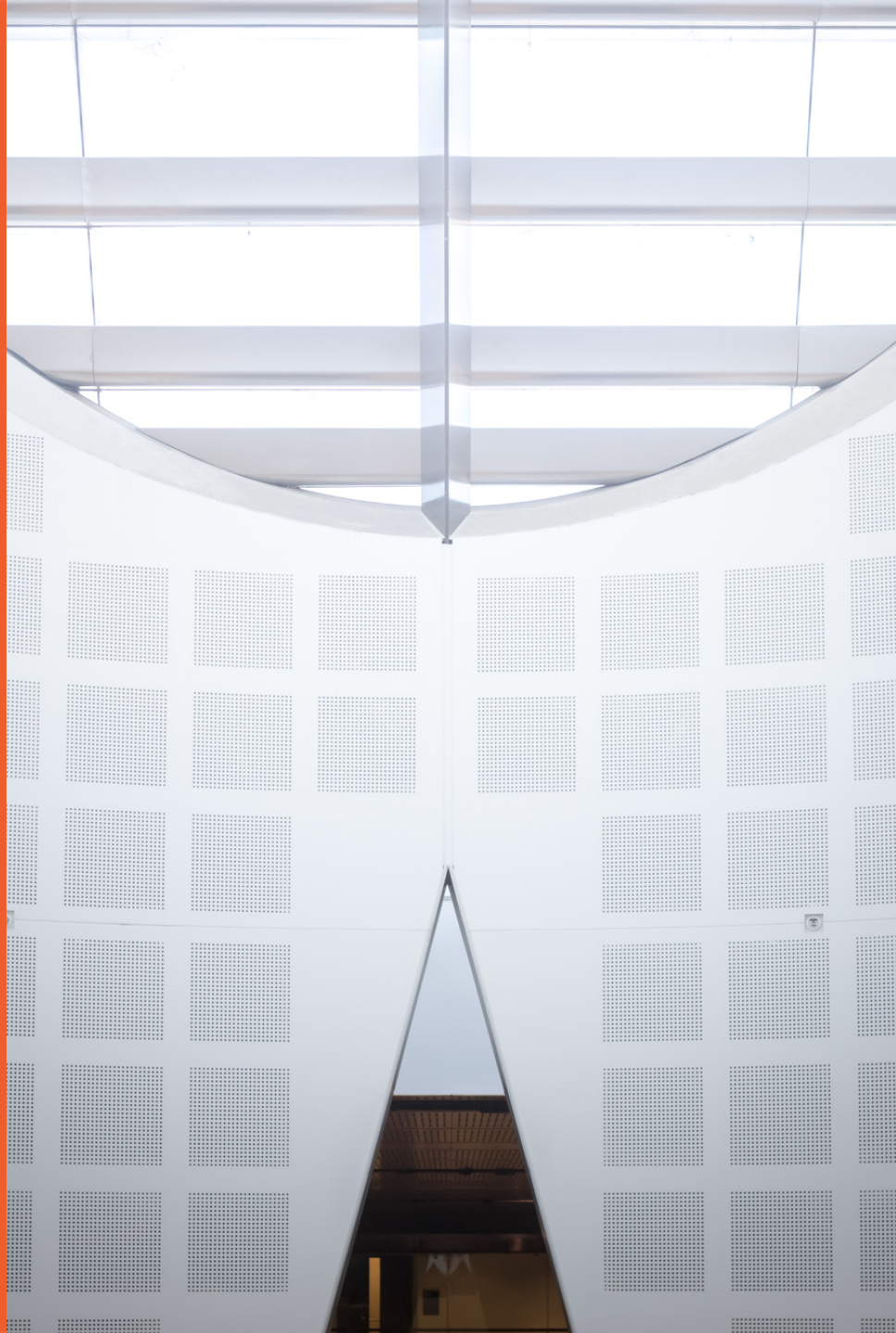
**Week 07**

**Semester 2, 2020**

Dr. Kanchana Thilakarathna

School of Computer Science

THE UNIVERSITY OF SYDNEY



# Outline

- Computation
  - Managing Computation tasks
  - Monitoring the load
  - Cloud computing support
- Energy Management
  - Energy consumption
  - Best practices for energy management
  - Platform supported energy management
- Towards battery less devices

# Computing Challenge

- Hardware updates screen every 16 milliseconds
- UI thread has 16ms to do all its work
- If it takes too long, app stutters or hangs

# Computing Challenge

- What are long running tasks ?
  - Downloading/Uploading files
  - Image processing, e.g. object detection
  - Loading data
  - Complex calculations

- How to check whether your app does well ?

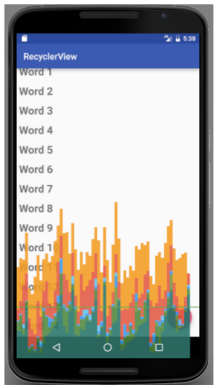
# Checking your frame rate

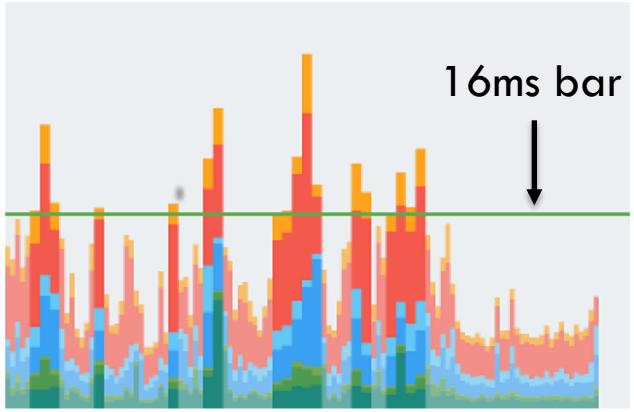
- What are long running tasks ?
  - Downloading/Uploading files
  - Image processing, e.g. object detection
  - Loading data
  - Complex calculations

- How to check whether your app does well ?
  - Settings > Developer options > Monitoring section > Profile GPU rendering > On screen as bars
  - How to find Developer options ?

# Checking your frame rate

- Settings > Developer options > Monitoring section > Profile GPU rendering > On screen as bars
- How to find Developer options ?
  - Hidden by default.



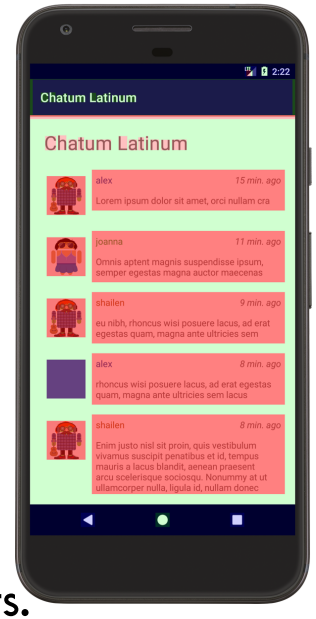

- One bar represents one frame of rendering
- Taller the bar, the longer it takes to render
- The horizontal green line represents 16 milliseconds.
- **Each frame needs to stay below this line → 60 frames/s**

# Visualize GPU overdraw

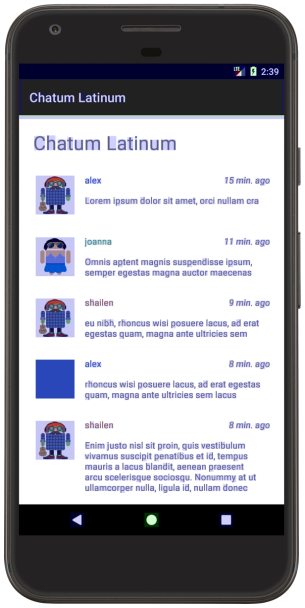
- Settings > Developer options > Hardware accelerated rendering> Debug GPU Overdraw > overdraw areas

  - **True color:** No overdraw
  - **Blue:** Overdrawn 1 time
  - **Green:** Overdrawn 2 times
  - **Pink:** Overdrawn 3 times
  - **Red:** Overdrawn 4 or more times

Lots of overdraw



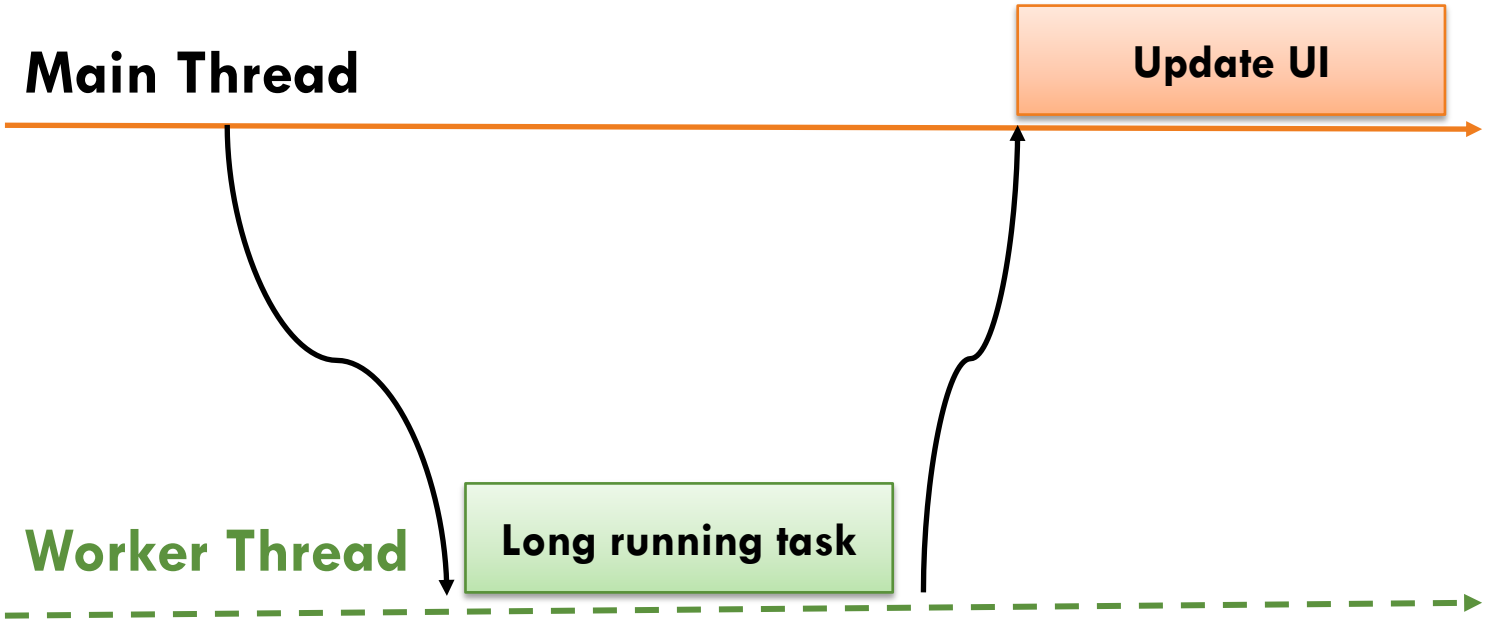

Little overdraw




- How to reduce overdraw ?
  - Removing unwanted backgrounds in layouts.
  - Flattening the view hierarchy.
  - Reducing transparency.
  - https://developer.android.com/topic/performance/rendering/overdraw

# Best Practice Computing

1. Complete tasks in less than 16ms
2. Move non-UI tasks to background thread




- In Android
  - AsyncTask, The Loader Framework, Services

# Best Practice Computing

1. Complete tasks in less than 16ms
2. Move non-UI tasks to background thread
3. Offload to a location with enough resources

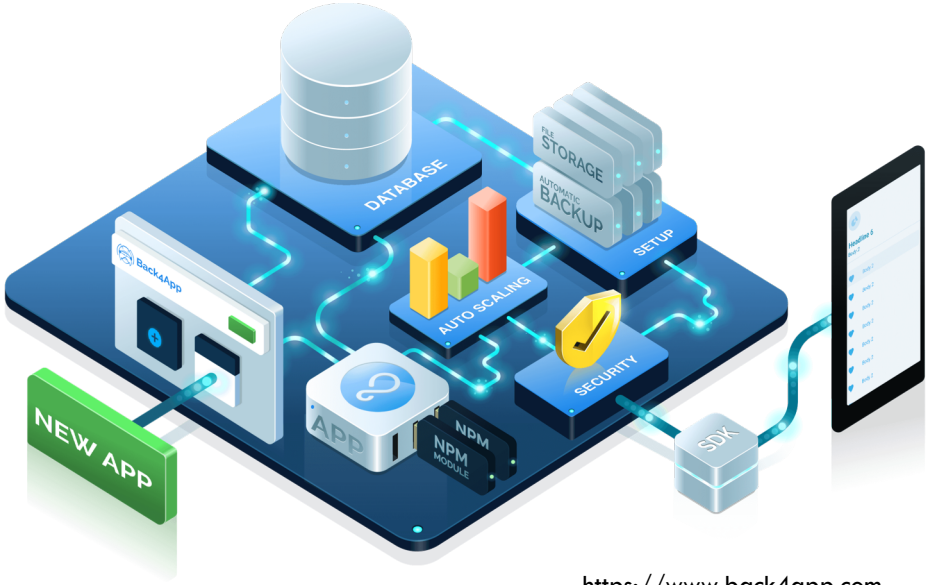
- **Mobile Computing + Cloud Computing**
  - Provides mobile application developers a way to connect their application to backend cloud storage and processing

# What is Mobile Cloud Computing ?

- **Mobile Computing + Cloud Computing**
  - Provides mobile application developers a way to connect their application to backend cloud storage and processing

- **"Anything-as-a-Service"**
  - Software-as-a-Service
  - Platform-as-a-Service
  - Infrastructure-as-a-Service
  - Mobile Backend as-a-Service

- Multiple monetization models
  - Pay per use
  - Subscriptions
- Private vs Public cloud resources




https://www.back4app.com

# Mobile Cloud Computing

- **Why ?**
  - Limited resources on mobile devices
    - Battery, Computation, Network, etc.
  - Abstract away complexities of app development
    - E.g. Google Play Services
  - Minimize launching and managing own infrastructure
    - Enable enterprises to treat IT as a **utility** than a **capital** expenditure
  - Focus more on front-end development instead of backend functions
  - Integration of multiple developers, apps, services
  - To enable data sharing
  - For permanent storage, backup
  - Easy app analytics
  - Security

- **Examples from current popular apps ?**

# Examples

- Apple Siri
  - Speech recognition → Too complex for a mobile device
- Dropbox, Google Drive, Apple iCloud
  - Unlimited storage → Not enough storage on mobile devices
- Google Play Location API
  - Hide complexity from the app developer
- Single Sign-on Authentication
  - Focus more on the front-end development
- Social Networking
  - Data storage and sharing, Push notifications

# MBaaS Providers

- Not a comprehensive list
  - Literary every company has a MBaaS Cloud Service







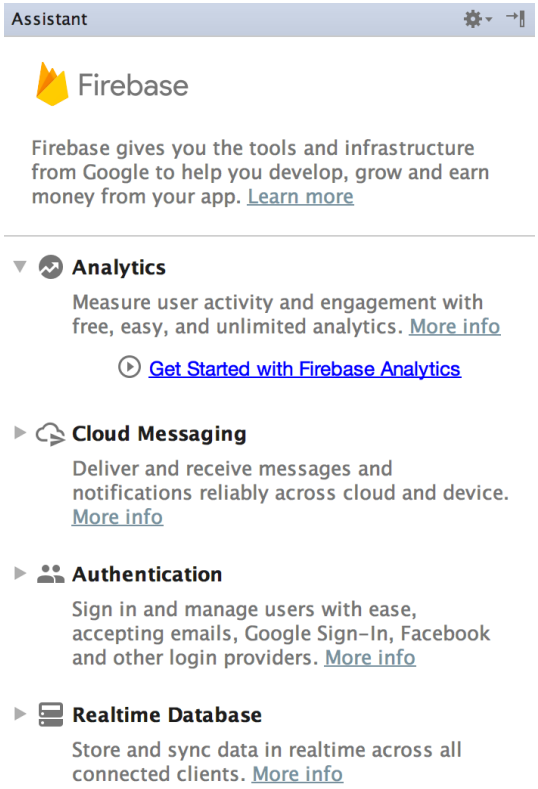


# Firebase

- Google MBaaS Solution
  - Not only for Android
  - Most Features supports


- Offer a number of features
  - Build better apps
  - Improve app quality
  - Grow your business

- Click **Tools >Firebase** to open the **Assistant** window
  - Tutorial 4

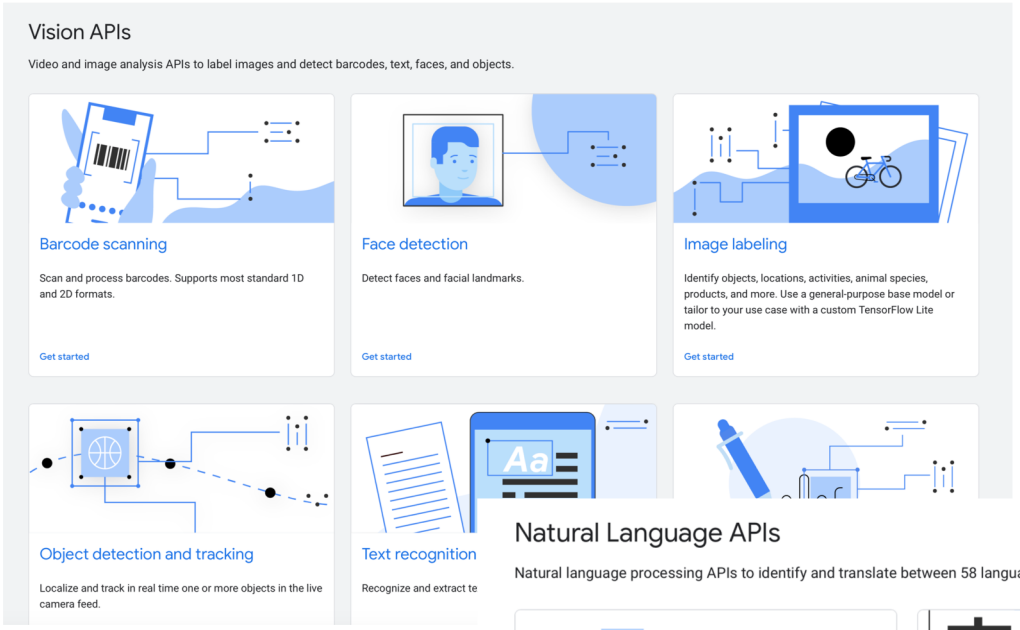


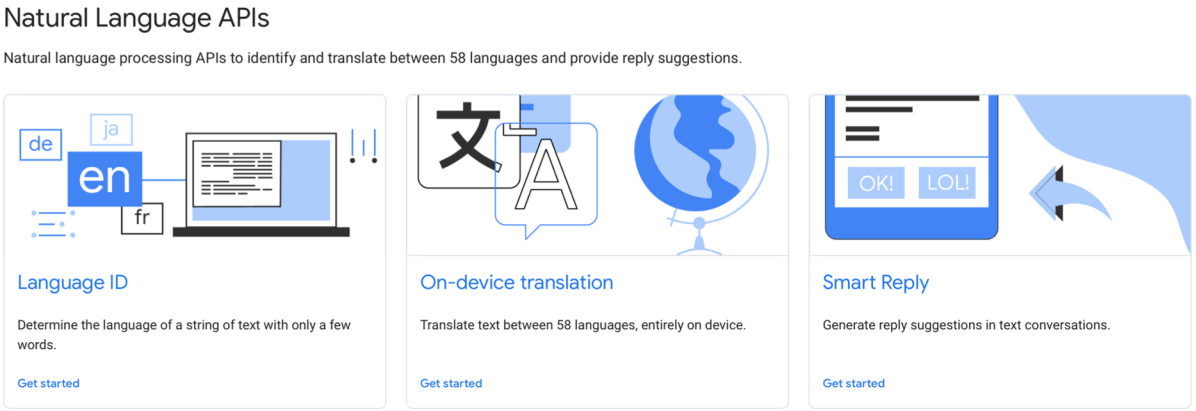
# Firebase

- Firebase ML - https://firebase.google.com/docs/ml
  - Still a beta version
  - APIs that work either in the in the cloud or on the device
  - Convenient APIs to use deep learning capabilities
  - For both iOS and Android
  - Text recognition
  - Face detection
  - Barcode reading
  - Label images
  - Landmark recognition
- Firebase Cloud Messaging - https://firebase.google.com/docs/cloud-messaging/
  - FCM provides a **single, persistent connection** to the cloud
  - All apps needing real-time messaging can share this connection

# ML Tool kit

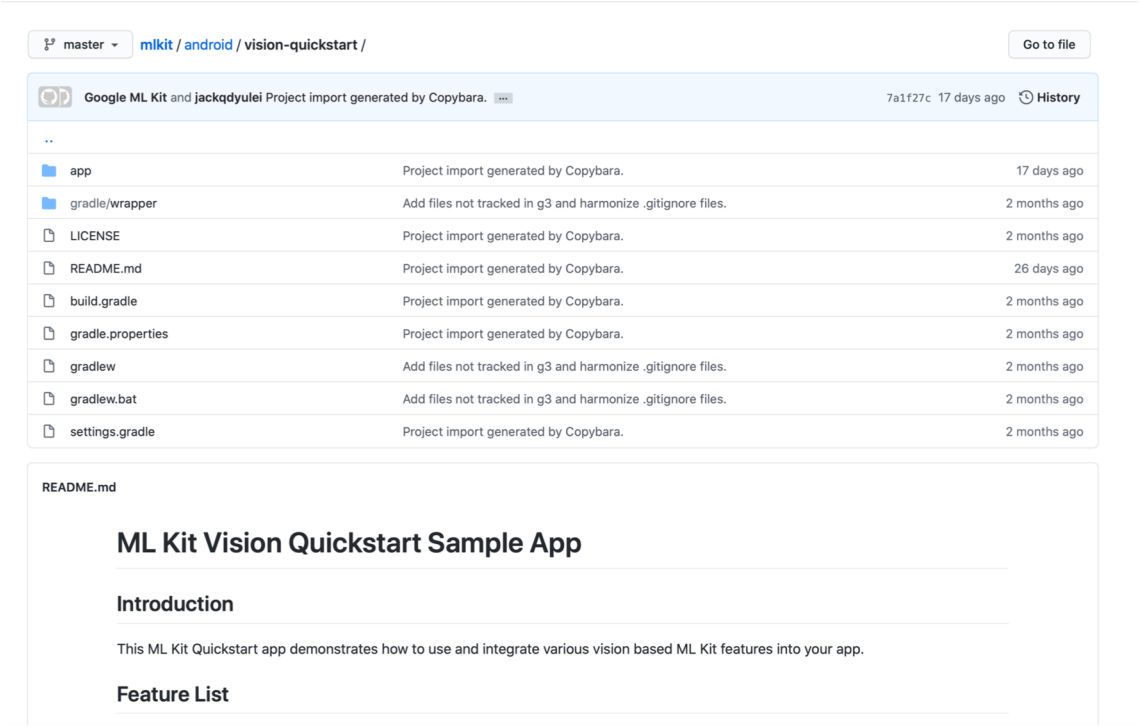
- ML Tool Kit – https://developers.google.com/ml-kit
  - Released on Jun 2020



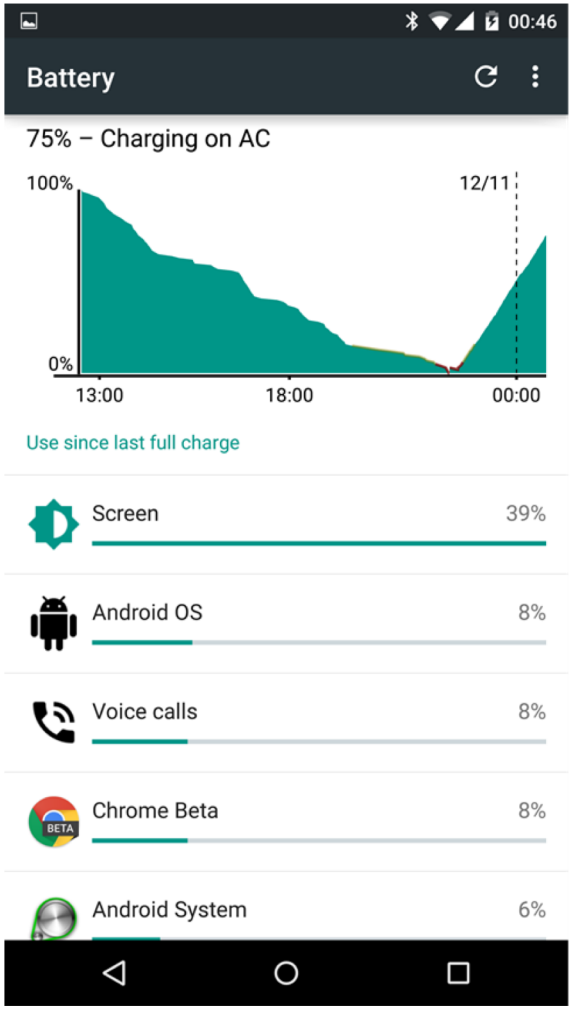
# ML Tool kit

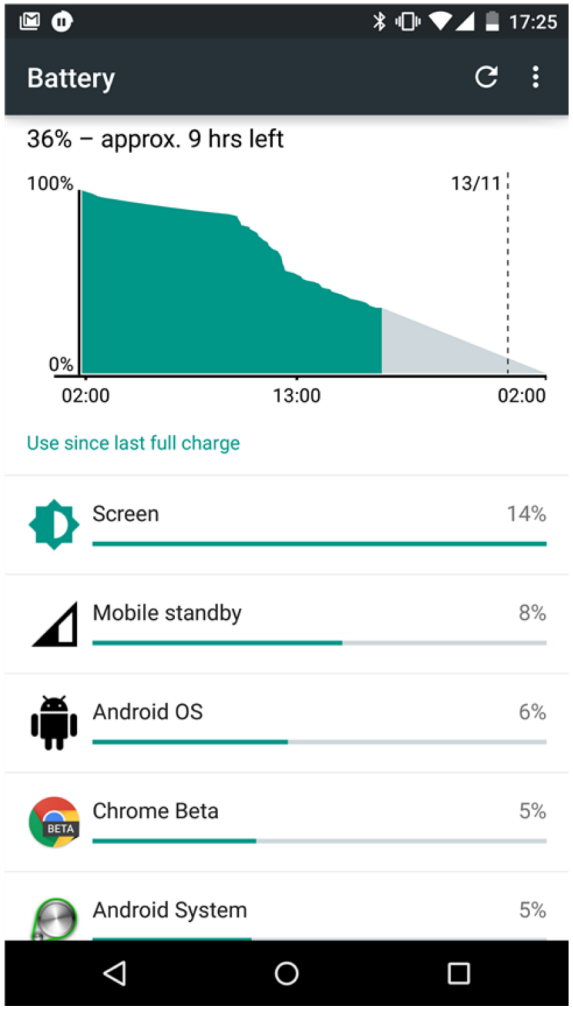
- ML Tool Kit – https://developers.google.com/ml-kit
  - Released on Jun 2020
- Samples: https://github.com/googlesamples/mlkit/tree/master/android/vision-quickstart



# Energy Management

# Battery consumption

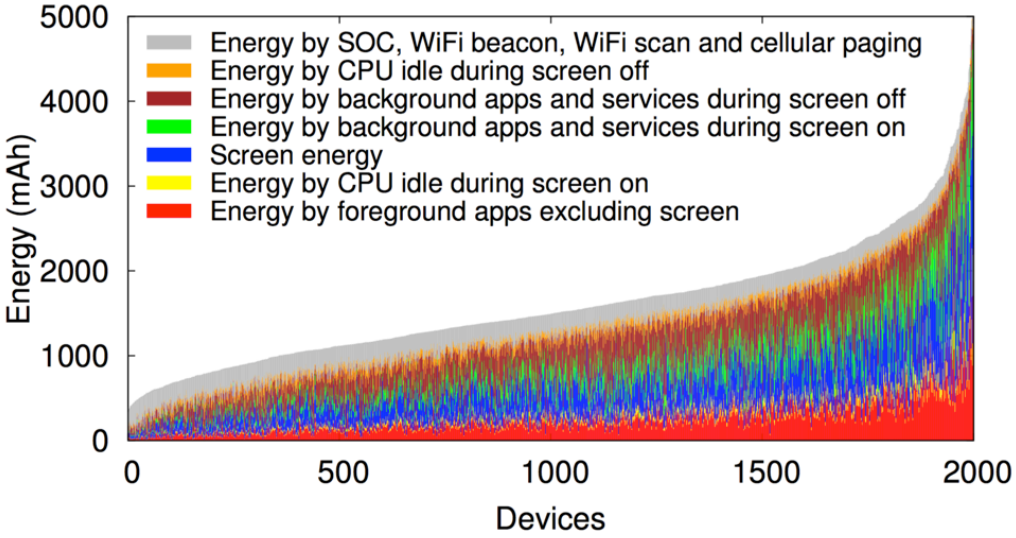


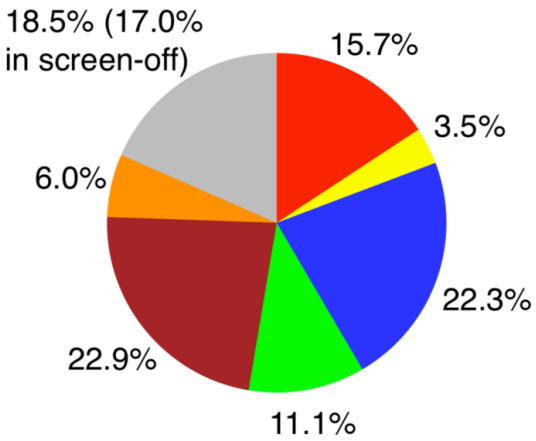



# Battery consumption

- **Screen & CPU**
  - Foreground vs Background
  - Activity vs Services
  - Sleep vs Active
- **Input modalities**
  - Type, Talk or Swype
- **Sensing**
  - Location (GPS vs Network)
  - Activity monitoring sensors - Accelerometer, Gyroscope, Magnetometer, etc.
  - Camera
- **Network interface**
  - Cellular>WiFi>Bluetooth in general
  - Communication protocol, e.g. HTTPS vs HTTP, content refresh rate

# Screen & CPU



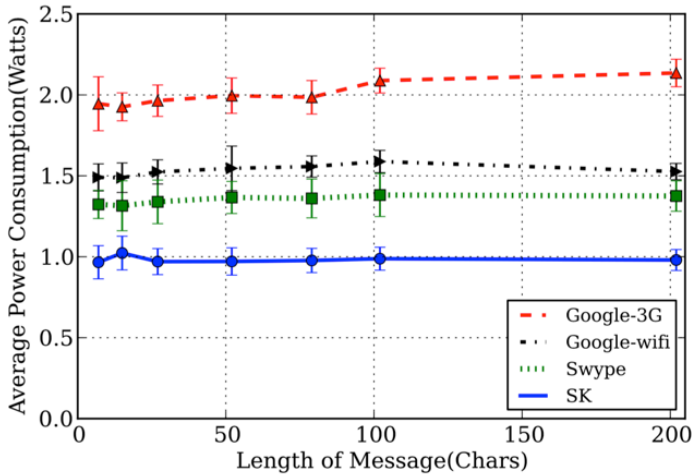




- Move the app to background, if the user interaction is not required.
  - Activity vs Services in Android
- Good programming practices
  - Efficient algorithms
  - Reduce disk access frequency

- Figure Reference: Chen, Xiaomeng, et al. "Smartphone Background Activities in the Wild: Origin, Energy Drain, and Optimization." Proceedings of the 21st Annual International Conference on Mobile Computing and Networking. ACM, 2015.

# Input Modalities

- Talk (Speech to Text), Type (Soft Key) and Swipe




CHARACTERISTICS OF DIFFERENT INPUT MODALITIES

| Input Modes | Accuracy | Convenience | Privacy | Speed | Energy Consumption *Short* | Energy Consumption *long* |
|---|---|---|---|---|---|---|
| **SK** | Highest | Low | High | Fast | Lowest | Low |
| **STT** | Lowest | High | Low | Fastest | High | Lowest |
| **Swype** | Medium | High | High | Slower | Low | High |

Reference: Jiang, Fangzhou, et al. "When to type, talk, or Swype: Characterizing energy consumption of mobile input modalities." Pervasive Computing and Communications (PerCom), 2015 IEEE International Conference on. IEEE, 2015.

# Sensing

- Sensor API are primarily used;
  - Identifying sensors and sensor capabilities
  - Monitor sensor events

- **Identify Sensor features**
  - getResolution() for sensor resolution
  - getMaximumRange() for maximum range of measurement
  - **getPower() for sensor's power requirements**
  - getVendor() and getVersion() to optimize for different sensors or different versions of sensor
  - **getMinDelay() to determine maximum rate at which sensor can acquire data**

# Sensor Delays

- Rely on the lowest possible sampling rate
  - Higher the sampling rate, higher the energy consumption

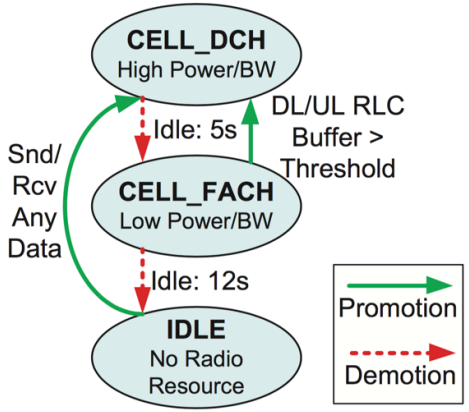
| Function | Energy cost | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Smartphone | | | | Smartwatch | | | |
| **Sensing** | Speed [mJ/s] | | | | | | | |
| | NORMAL | UI | GAME | FASTEST | NORMAL | UI | GAME | FASTEST |
| Accelerometer | 5.01 | 13.28 | 34.46 | 77.71 | 9.52 | 24.74 | 57.61 | 168.4 |
| Gyroscope | 11.71 | 20.33 | 36.44 | 80.15 | 16.23 | 33.34 | 60.44 | 182 |
| Magnetometer | 8.12 | 15.45 | 28.46 | 28.28 | 17.04 | 30.21 | 57.82 | 79.73 |

| **Connectivity** | Per Byte [mJ/B] | High Power Idle [mJ] | Low Power Idle [mJ] | Per Byte [mJ/B] | High Power Idle [mJ] | Low Power Idle [mJ] |
|---|---|---|---|---|---|---|
| Bluetooth | 0.0095 | 305 | 300 | 0.0024 | 126.07 | 64.23 |
| WiFi | 0.0005 | 66 | N/A | 0.0004 | 50 | N/A |

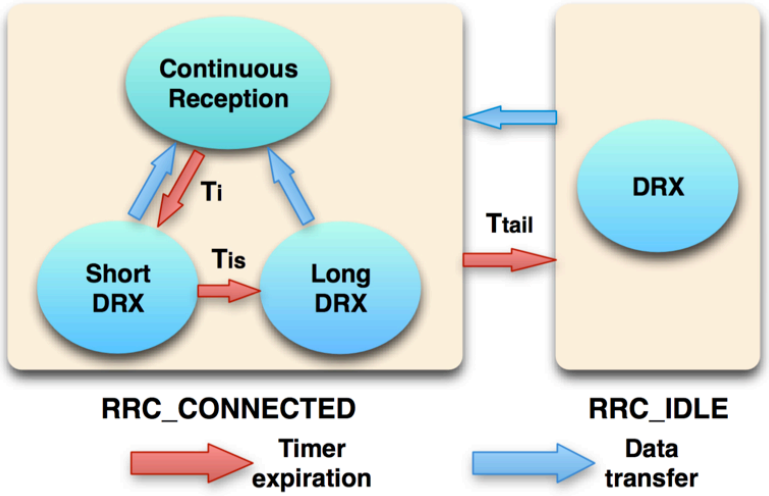
Kolamunna, H. D., Thilakarathna, K., Perino, D., Makaroff, D.,&Seneviratne, A. (2018). Seamless Resources Sharing in Wearable Networks by Application Function Virtualization. *IEEE Transactions on Mobile Computing*.

# Networking Radio/Chipset on mobile devices

- Not always ON
- Becomes ACTIVE, only when it is required to transfer data
- After a transfer, stays ON for some time
- Eventually, goes back to SLEEP




3G State machine.

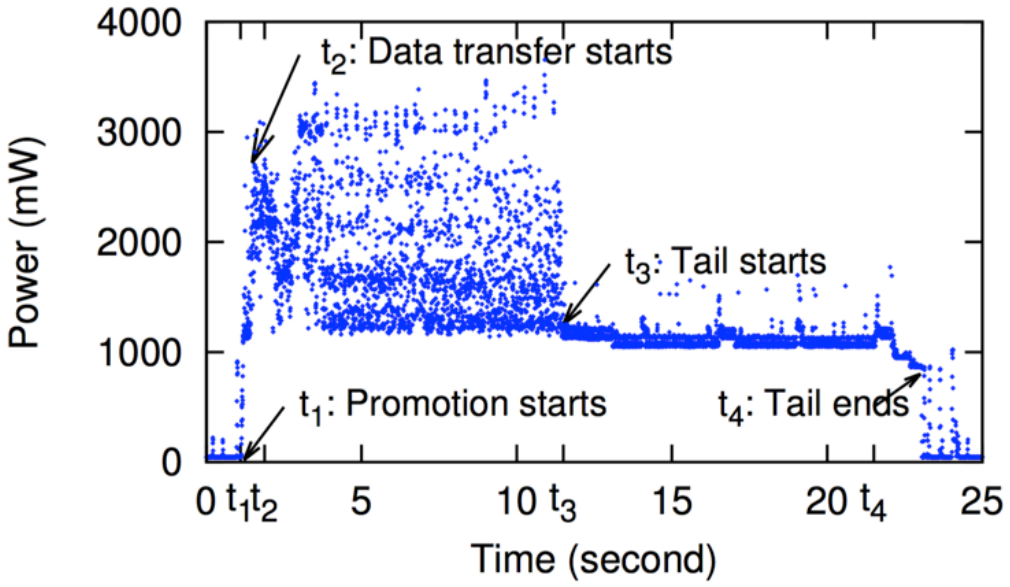



LTE State machine.

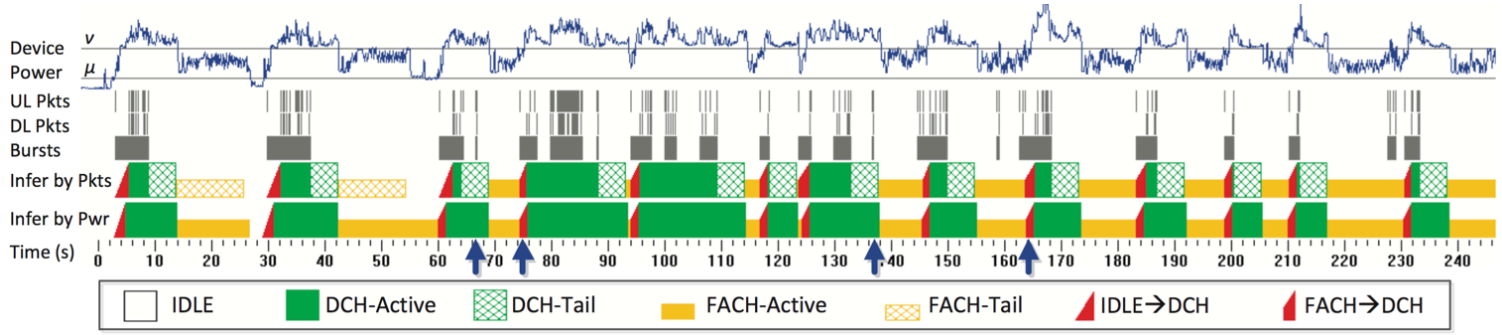
[1]. Qian, Feng, et al. "Profiling resource usage for mobile applications: a cross-layer approach". Proceedings of the 9th international conference on Mobile systems, applications, and services. ACM, 2011.
[2]. Huang, Junxian, et al. "A close examination of performance and power characteristics of 4G LTE networks". Proceedings of the 10th international conference on Mobile systems, applications, and services. ACM, 2012.

# LTE Network Power Usage

- Power consumption of LTE data communication

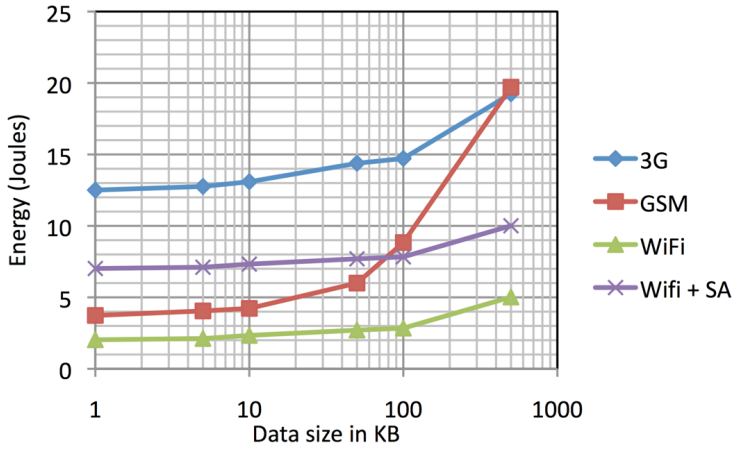


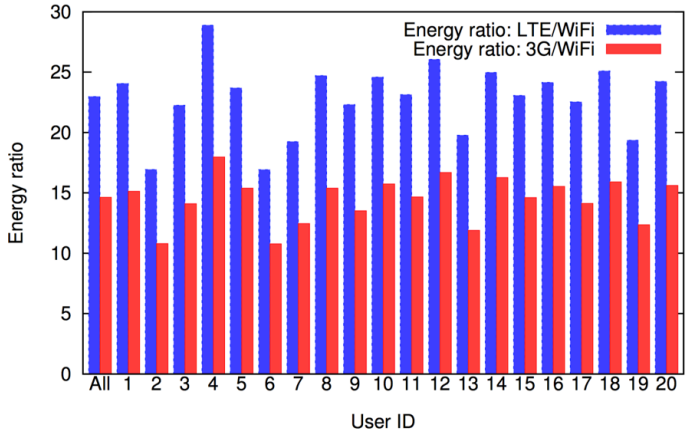

1. **High Power**
2. **Medium Power**
3. **Low Power**



# Network Power Usage

## LTE vs 3G vs WiFi



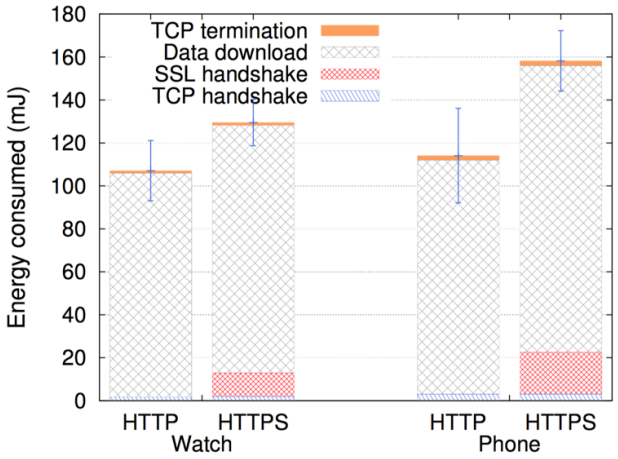




- In general, LTE>3G>WiFi.
  - Dependent on the network conditions, e.g. available bandwidth, signal strength, interference, etc.

Balasubramanian, Niranjan, Aruna Balasubramanian, and Arun Venkataramani. "Energy consumption in mobile phones: a measurement study and implications for network applications." Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference. ACM, 2009.

# Privacy and Security

- Energy cost of secure protocols is not negligible.
  - More data due to adding noise, encryption, channel coding, etc.
  - Extra communication due to key exchanges




- Security is important in today's mobile world.
  - **It is worth the cost of energy**

H. Kolamunna, J. Chauhan, K. Thilakarathna, D. Perino, D. Makaroff and A. Seneviratne, "Are Wearables Ready for Secure and Direct Internet Communication?", ACM GetMobile: Mobile Computing and Communications, vol. 21, no. 3, pp. 5-10, Sep 2017.

# Best Practices for Energy Management

- **"Lazy First"** design strategy

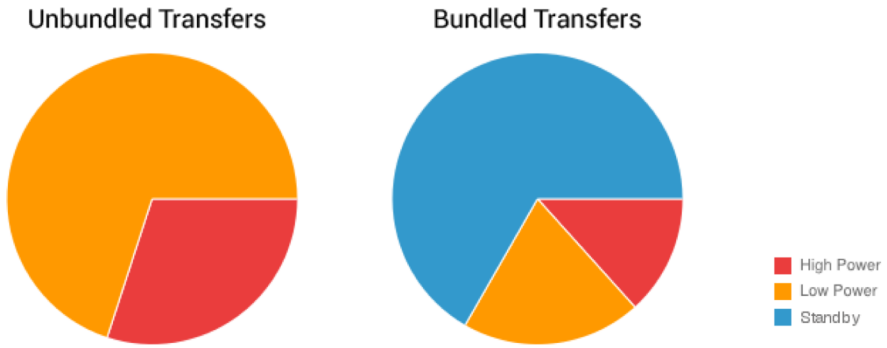
1. **Reduce**
   - Can we cache data with out re-downloading?
2. **Defer**
   - Can we wait until the device is charging ?
3. **Batch**
   - Can we batch downloads together?

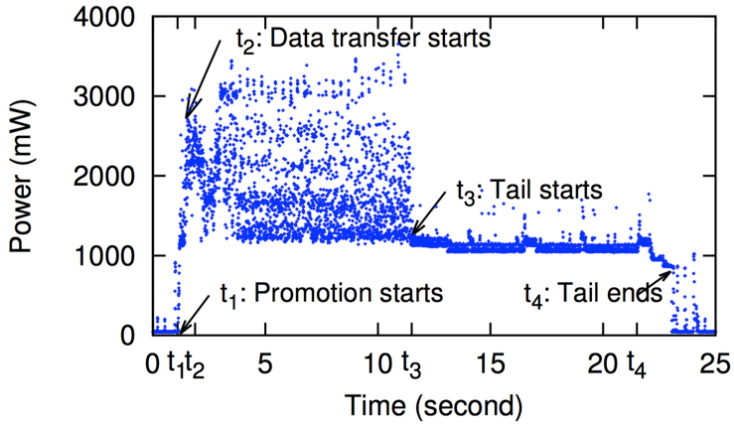
# Best Practices for Energy Management

- **Reduce**
  - Upload/Download only necessary data [Week 5]
  - Reduce – Caching [Week 5]
  - Compressing [Week 5]
- **Defer**
  - Offloading (reactive/predictive) to energy efficient networks [Week 5]
- **Batch**
  - Reduce the frequency of communication
  - Use an exponential back-off pattern when you sync or poll to extend the time between subsequent polls

# Best Practices for Energy Management

- Batch transfers and connections
  - E.g. analytics information should batched together and queued to be bundled with another download or upload
  - In Android, JobScheduler API or Firebase JobDispatcher



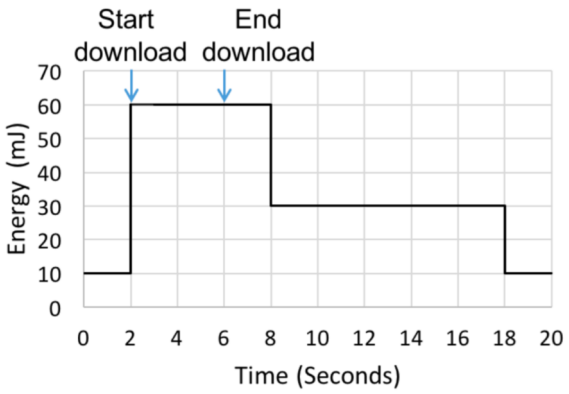



# Example Question

You have decided to develop your own ads library to reduce device energy consumption. To properly configure the relevant parameters, you first conducted an experiment to measure the energy consumption of an Android phone when downloading an image of 100KB using the 3G (UMTS) Cellular network. The obtained average energy profile is in the Figure. Note that it takes 4 seconds to download a 100KB file and the idle energy consumption is 10 mJ per seconds as shown in the Figure.

Q1. Consider a scenario where there is **5 ads to be displayed during a 2.5 minutes time frame and the ad library refresh ads at every 30 seconds.** Calculate the energy consumption of the app (in mJ). Note: assume each ad is of size 100KB.

Q2. Calculate the amount of energy that can be saved, if your ad library fetch all 5 ads at the same time.



# Best Practices for Energy Management

- Avoid polling server to see whether there is an update
  - Let the server push data to the app only when there is an update
  - Firebase Cloud Messaging (FCM) implements lightweight mechanism to transmit data.

- Measure the performance of your app using Network Profiler

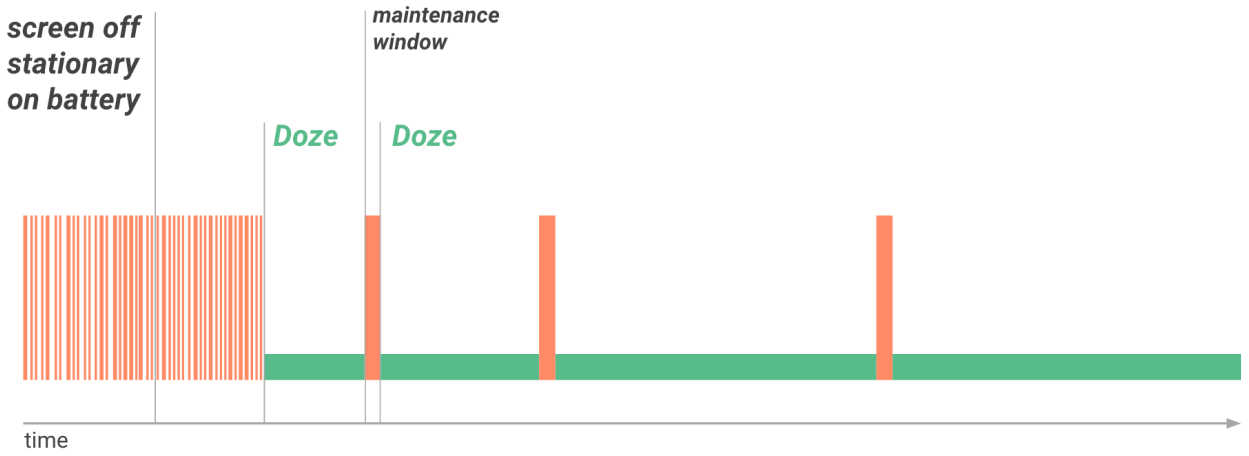
- Best programming practices
  - https://developer.att.com/video-optimizer/docs/best-practices

# Best Practices for Energy Management

## Android Platform Battery Management

### 1. Doze

- Deferring background CPU and network activity for apps when the device is unused for long periods of time
- If the device is **unplugged** and **stationary for a period of time**, with the **screen off**, the device enters Doze mode.
- After that, system provides **periodic maintenance windows**




https://developer.android.com/training/monitoring-device-state/doze-standby

# Best Practices for Energy Management

## Android Platform Battery Management

- Doze restrictions
  - Network access is suspended, including WiFi scans
  - Does not allow JobScheduler, SyncAdapters
  - Ignores wake locks.
  - Standard Alarms will not go off.

**2. App Standby**

- If the user does not touch the app for a certain period of time
- No current foreground processes
- No notifications
- Not a an admin app, Android move the app to Standby mode

- For list of all restrictions of Doze and App Standby
  - https://developer.android.com/topic/performance/power/power-details

# Best Practices for Energy Management

## Android Platform Battery Management

### 3. App Standby Buckets

The system dynamically assigns each app to a priority bucket

- Active: App is currently being used or was very recently used
- Working set: App is in regular use
- Frequent: App is often used, but not every day
- Rare: App is not frequently used

# Best Practices for Energy Management

## Android Platform Battery Management

- Restrictions
  - https://developer.android.com/topic/performance/power/power-details

| Setting | Jobs * | Alarms † | Network ‡ | Firebase Cloud Messaging § |
|---|---|---|---|---|
| **User Restricts Background Activity** | | | | |
| Restrictions enabled: | Never | Never | No restriction | No restriction |
| **Doze** | | | | |
| Doze active: | Deferred to window | Regular alarms: Deferred to window<br>While-idle alarms: Deferred up to 9 minutes | Deferred to window | High priority: No restriction<br>Normal priority: Deferred to window |
| **App Standby Buckets** (by bucket) | | | | |
| Active: | No restriction | No restriction | No restriction | No restriction |
| Working set: | Deferred up to 2 hours | Deferred up to 6 minutes | No restriction | No restriction |
| Frequent: | Deferred up to 8 hours | Deferred up to 30 minutes | No restriction | High priority: 10/day |
| Rare: | Deferred up to 24 hours | Deferred up to 2 hours | Deferred up to 24 hours | High priority: 5/day |

# Best Practices for Energy Management

## Android Platform Battery Management

- **Make sure your app functions well under Doze & App Standby modes**
  - If you need to set alarms that fire while in Doze, use setAndAllowWhileIdle() or setExactAndAllowWhileIdle().
  - Use **Firebase Cloud Messaging (FCM)**
  - FCM is optimized to work with Doze and App Standby idle modes by means of high-priority FCM messages
    - Gives the app temporary access to network services and partial wake locks, then returns the device or app to the idle state.

- Partial exemption from restrictions
  - Users can manually configure the list of exempted apps in **Settings > Battery > Battery Optimization**
  - REQUEST_IGNORE_BATTERY_OPTIMIZATIONS permission can trigger a system dialog without user setting it manually

# Best Practices for Energy Management

## Android Platform Battery Management

**Best practices to mange App Standby Bucket**

- Do not try to manipulate the system into putting your app into one bucket or another.
- If an app does not have a launcher activity, it might never be promoted to the active bucket.
- If the app's notifications (including high-priority FCM message) aren't actionable, users won't be able to trigger the app's promotion to the active bucket by interacting with the notifications.
- You should be sure to test such apps with the packages assigned to various buckets to make sure the app behaves properly.

- You should test your app fully in **Doze** and **App Standby**
  - Forcing the system into Doze mode by running the following command:
  - $ adb shell dumpsys deviceidle force-idle

  - Forcing the app into App Standby mode by running the following commands:
  - $ adb shell dumpsys battery unplug
  - $ adb shell am set-inactive <packageName> true

# Profile battery usage in Android

- **Batterystats** – a tool in Android framework
- **Battery Historian** – visualize data collected from Batterystats

1. **Install Battery Historian**
   - **https://github.com/google/battery-historian**

- Using Docker
  - Install Docker.
  - Choose a port number and replace <port> with that number in the commands below:
  - `docker -- run -p<port>:9999 gcr.io/android-battery-historian/stable:3.0 --port 9999`

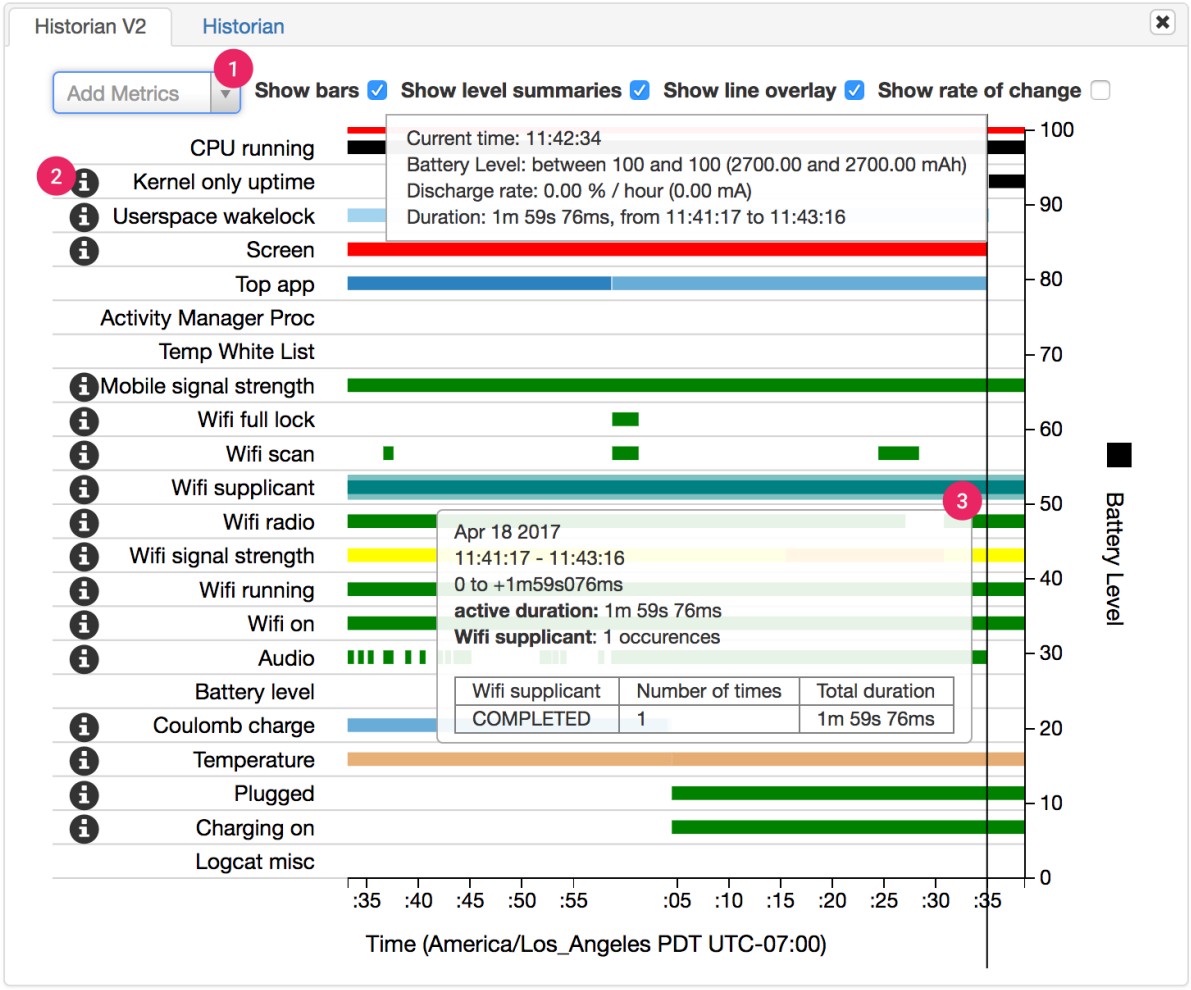
  - For Linux and Mac OS X:
    - Historian will be available at `http://localhost:<port>`.

  - For Windows:
    - You may have to enable Virtualization in your BIOS.
    - Find the IP address of docket.
    - Historian will be available at `http://<ip address>:<port>`.

# Profile battery usage in Android

- Collect data with BatteryStats
- Using Android Debug Bridge (ADB)
  - https://developer.android.com/studio/command-line/adb
  - Follow this walkthrough - https://developer.android.com/studio/profile/battery-historian
    - Connect the device
    - Restart battery stats collection
      - `adb shell dumpsys batterystats –reset`
    - Disconnect the device
    - Play with your app
    - Reconncet the devoce
    - Dump battery stats
      - `adb shell dumpsys batterystats>[path/]batterystats.txt`
      - `adb bugreport>[path/]bugreport.txt`
    - Run Battery Historian and open bugreport file

# Profile battery usage in Android




1. Add more metrics
2. More info including the color legend
3. More info of the metric at that particular time

# Towards battery less devices

- Ambient energy harvesting
  - Radiant – Light, Radio frequencies
  - Kinetic – Vibrations, human motion
  - Thermal – Temperature
  - Chemical – Metabolic reactions


- Using body heat to power wearables
  - PowerWatch
  - https://www.powerwatch.com


- Kinetic energy harvesting shoes
  - https://newatlas.com/energy-harvesting-shoes/41796/


- Kumara Kahatapitiya, Chamod Weerasinghe, Jinal Jayawardhana, Hiranya Kuruppu, Kanchana Thilakarathna, and Dileeka Días. 2018. Low-power step counting paired with electromagnetic energy harvesting for wearables. In Proceedings of the 2018 ACM International Symposium on Wearable Computers (ISWC '18)

# What's Next ?

- Next week
  - Mobile computing beyond smartphones

- Homework 2 is due next week !

- Guest/Industry Lecture – 26th October 2020
  - **Cross-Platform App Development**
  - **Glenn Stephens from Microsoft**

- Project proposal marks will be made available soon !
  - If you want feedback about your proposal, please talk to me !