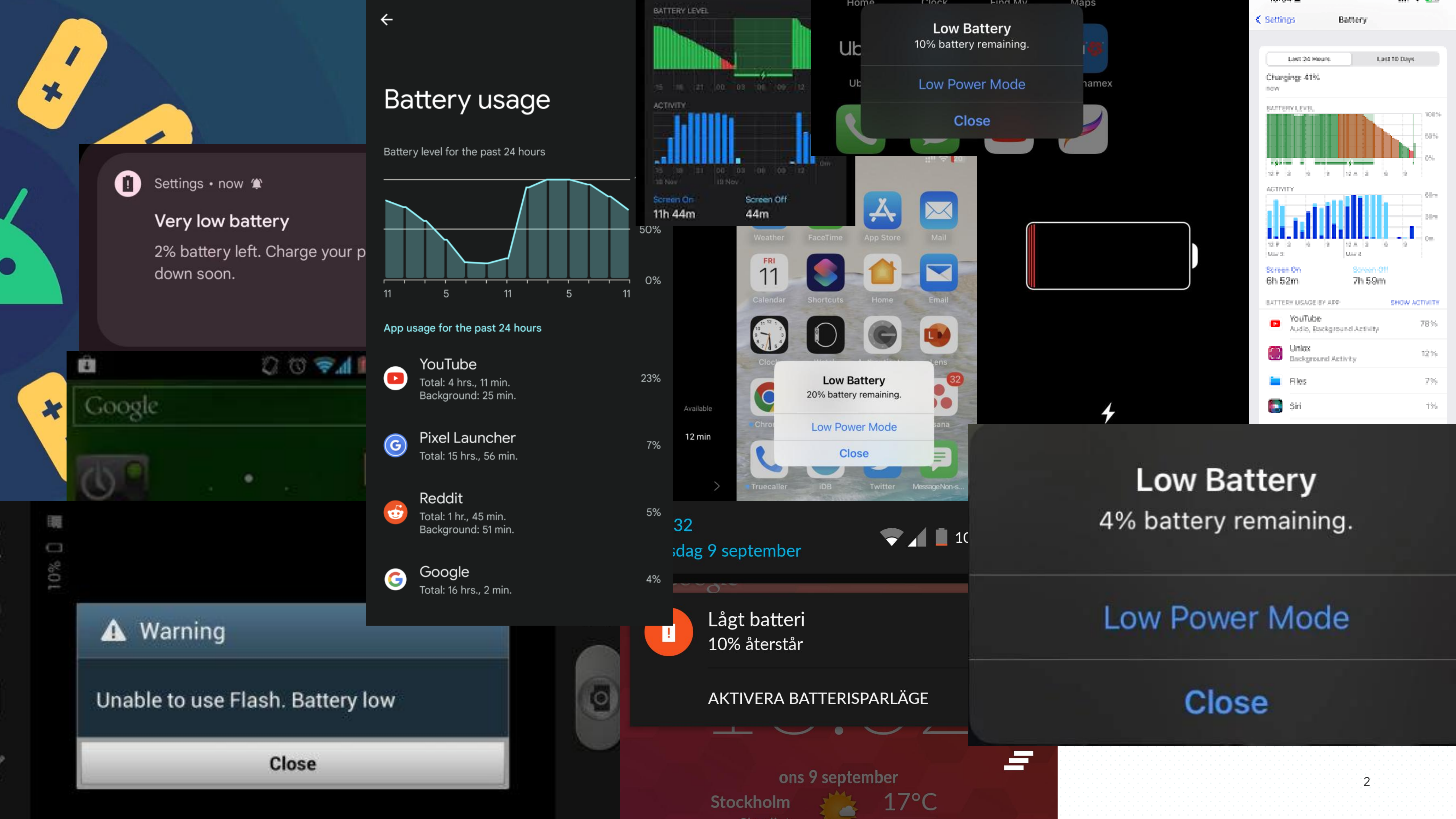
Battery Optimization for Mobile Devices – an overview of techniques to extend battery life, focusing on software solutions.

Presenters:
Marty Lauterbach,
Youmna Samouneh,
Luis Wehrberger

Fakultät für Informatik
Reutlingen University

[1] K. Shankari, D. E. Culler, R. H. Katz, "Doing Nothing Well: OS-Application Coordination for Energy Saving," **UC Berkeley EECS Tech. Rep. UCB/EECS-2016-119**, June 2016www2.eecs.berkeley.eduwww2.eecs.berkeley.edu.
[2] K. Athukorala et al., "How Carat Affects User Behavior: Implications for Mobile Battery Awareness Applications," *Proc. ACM CHI*, 2014amplab.cs.berkeley.eduamplab.cs.berkeley.edu.
[3] A. Carroll, G. Heiser, "An Analysis of Power Consumption in a Smartphone," *Proc. USENIX ATC*, 2010usenix.orgusenix.org.
[4] S. Elmalaki et al., "A Case for Battery Charging-Aware Power Management and Deferrable Task Scheduling in Smartphones," in *Proc. HotPower*, 2014usenix.org.
[5] D. Flores-Martin, S. Laso, J. L. Herrera, "Enhancing Smartphone Battery Life: A Deep Learning Model Based on User-Specific Application and Network Behavior," *Electronics*, vol. 13, no. 24, 4897, 2024mdpi.commdpi.com.
[6] J. Smith, S. Rosen, C. Gamble, "DeepMind, meet Android," *Google DeepMind Blog*, May 8, 2018deepmind.google.
[7] Freescale Semiconductor (NXP), *"Low-Power Sensing"* White Paper, 2010nxp.com.
[8] N. Ali, "Samsung's solid-state batteries: A new era for wearable tech with 200 Wh/L energy density," *NotebookCheck News*, Jan. 2024notebookcheck.net.


Settings • now
Very low battery
2% battery left. Charge your p
down soon.
Google
Warning
Unable to use Flash. Battery low
Close
Battery usage
Battery level for the past 24 hours
50%
0%
11
5
11
5
11
App usage for the past 24 hours
YouTube
Total: 4 hrs., 11 min.
Background: 25 min.
23%
Pixel Launcher
Total: 15 hrs., 56 min.
7%
Reddit
Total: 1 hr., 45 min.
Background: 51 min.
5%
Google
Total: 16 hrs., 2 min.
4%
BATTERY LEVEL
ACTIVITY
Screen On
11h 44m
Screen Off
44m
Low Battery
10% battery remaining.
Low Power Mode
Close
Weather
FaceTime
App Store
Mail
FRI
11
Calendar
Shortcuts
Home
Email
Low Battery
20% battery remaining.
Low Power Mode
Close
Truecaller
iDB
Twitter
Available
12 min
32
dag 9 september
Lågt batteri
10% återstår
AKTIVERA BATTERISPARLÄGE
ons 9 september
Stockholm
17°C
Settings
Battery
Charging: 41%
Screen On
6h 52m
Screen Off
7h 59m
YouTube
78%
Files
7%
Siri
1%
Low Battery
4% battery remaining.
Low Power Mode
Close

What are you supposed to learn?

Key Chunks:

- **What you can do yourself**
- **What you can tell the User**
- **What you have in your pocket right now**
- **What you can code**

Agenda

1. Introduction & Context

(Battery characteristics, power draw by component, user impact) *Interactive* (1-10) Marty

2. OS-Level Strategies

(Background processing, power modes, task scheduling) – Both *What you can do*, and *what the Device does* (11-13) Youmna + (14-15) Luis

3. AI-Based Optimization

(Adaptive battery/brightness, ML prediction) (16-17) Luis

4. Hardware & User Factors

(Low-power hardware, new battery tech, user habits) (18-21) Luis

5. How to Code with the Battery

/ Device State in React Native (22-27) Youmna

Its interactivity time!

One second while I open Menti – QR Code will be right up again!

What do you think can be done to save some battery juice?

How much Percen hones Battery-Usage (mW) do you thin mponents are responsible for?

Power Consumption Breakdown

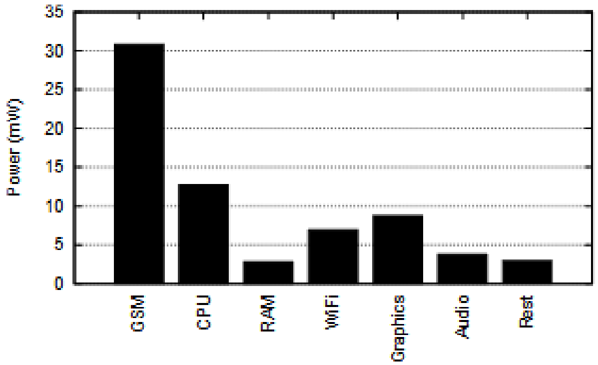


Figure 2: Power breakdown in the suspended state. The aggregate power consumed is 68.6 mW.

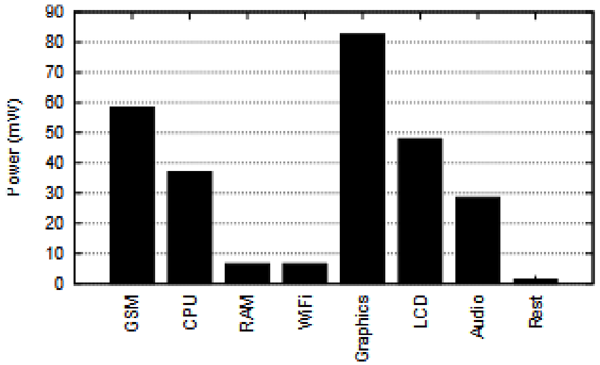


Figure 3: Average power consumption while in the idle state with backlight off. Aggregate power is 268.8 mW.

usenix.org :4

- **Display:** Typically the **largest power consumer** (up to ~50% of idle power; ~80% at max brightness) usenix.org
- **CPU/GPU:** Intensive tasks (games, video processing) heavily draw power. usenix.org
- **Other:** RAM, audio, and peripherals draw smaller amounts when idle. Battery health and temperature also affect efficiency.
- **Radio (Cellular/Wi-Fi):** Keeping network connections and scanning for signals consumes ~30–45% in sleep/idle usenix.org
- **Sensors (GPS, accelerometer, etc.):** Location and motion tracking can be energy-intensive when active amplab.cs.berkeley.edu

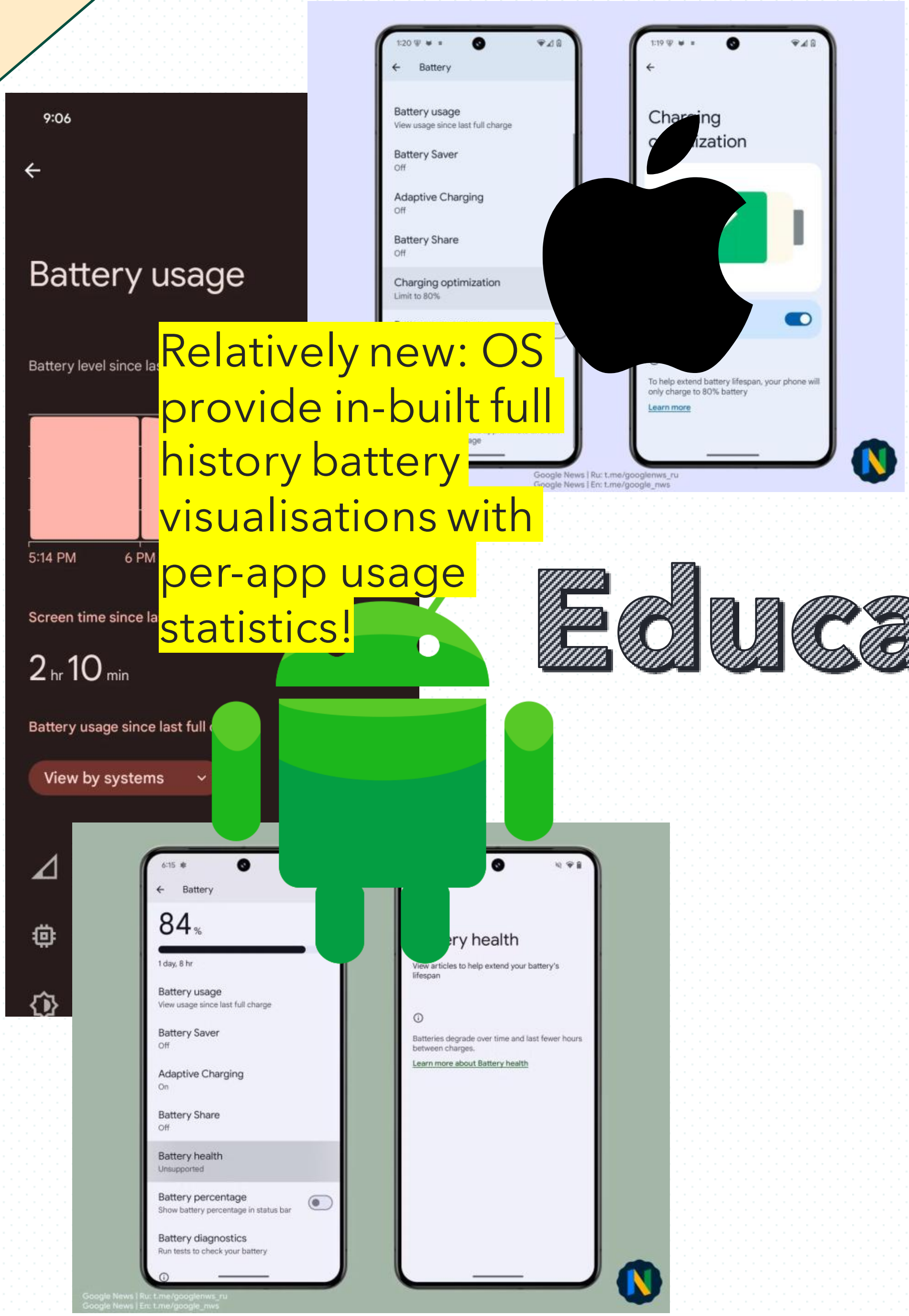
Why Battery Optimization?

Modern mobile devices often need recharging more than once a day. Smartphones have **limited battery capacity**, and heavy use (screen, CPU, radios) quickly drains power.

Even when idle, background tasks and connectivity consume energy.

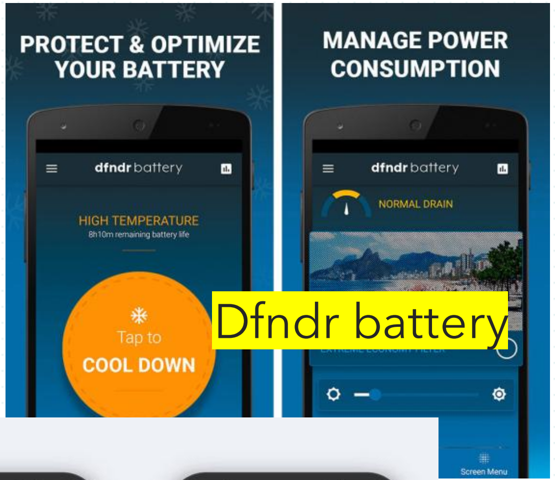
Battery optimization **extends usage time** and improves user experience by reducing "out-of-power" incidents.

amplab.cs.berkeley.eduamplab.cs.berkeley.edu



Relatively new: OS provide in-built full history battery visualisations with per-app usage statistics!

Educate & Provide



Dfndr battery

GO Battery Pro

Battery Monitor

User Behavior & Awareness

• **Usage Patterns:** Excessive screen brightness, unused connectivity (Bluetooth/GPS on) and background apps shorten battery life. **Awareness of settings is key.**

• **Battery Apps (Awareness):** People using tools that remind them of settings *save more battery and charge less often*. These users better manage resources over time - *save more battery and charge less often*. amplab.cs.berkeley.edu

• **Education:** Users often lack direct feedback on what drains power. Good UI/feedback (e.g. notifying which app sleeps unused) can influence behavior – **many OS feature a latest Overview!**

.. (old) battery optimization app – but the point stands!

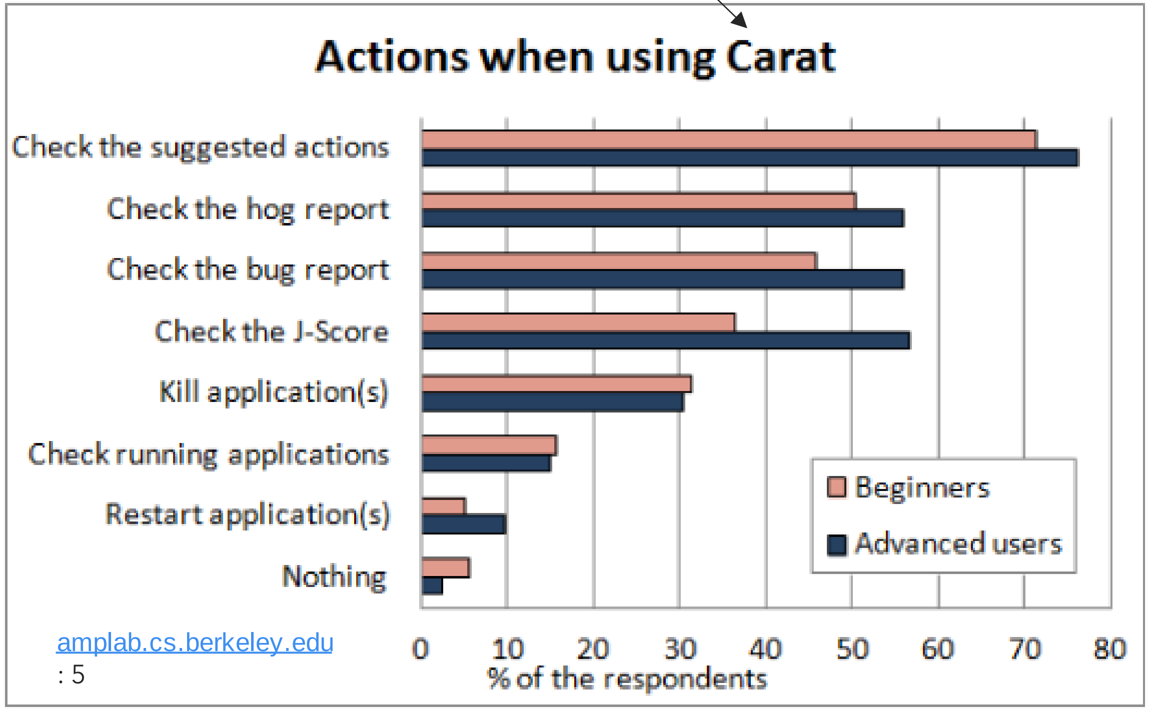


Figure 4. The actions performed when Carat is open.

User Habits and Tips

- **User-Controlled Settings:** Encouraging users to enable power-saving modes, close unused apps, or limit push notifications can yield significant savings.

- **Case Study:** Some phones now prompt users to activate “sleep mode” at bedtime, cutting background data. These nudges rely on understanding user routines.

- **Behavioral Impact:** Education (via battery apps or OS hints) is essential: knowing that a “battery-savvy” habit (turning off GPS) does help.

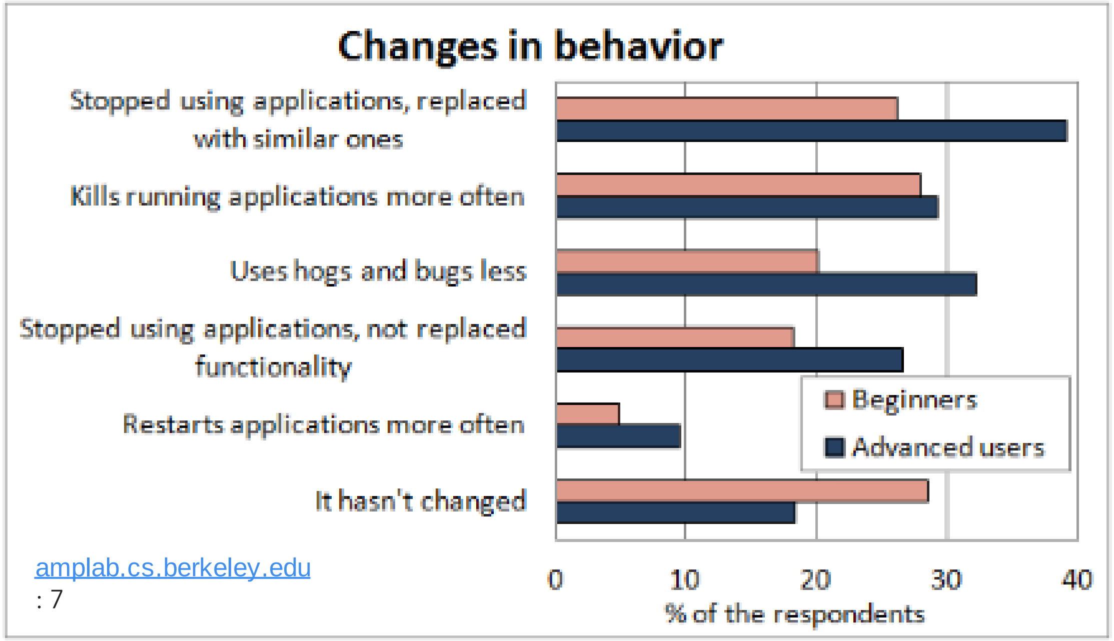


Figure 7. Summary of responses to the survey question “In what ways using Carat has changed the way you use your device?”.

2: OS-Level Software Strategies

- **iOS:** Historically limits apps running in background; no CPU activity allowed unless via special modes. As a result, iPhones show *nearly zero* overnight drain with no active use (geofencing for example) www2.eecs.berkeley.edu.

- **Android 6+ (Doze/App Standby):** Introduced *Doze mode* which batches background tasks into maintenance windows. A Nexus 6 saw only ~6% overnight drain thanks to Doze www2.eecs.berkeley.edu.

- **Unified Trend:** Both OSs now restrict background work aggressively. Developers must use scheduled jobs or listeners (e.g. iOS LocationTriggers) to comply www2.eecs.berkeley.edu www2.eecs.berkeley.edu.

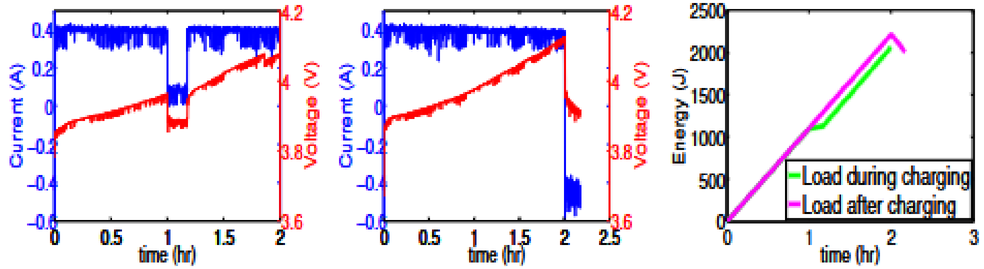
Power-Saving APIs and Modes

- **Adaptive Brightness & Screen:** OS controls can auto-adjust backlight. For example, *Adaptive Brightness* learns user prefs and saves display power.
- **Low Power/Saver Modes:** Both Android and iOS offer modes (e.g. Android Power Saver, iOS Low Power Mode) that limit CPU speed, syncs, and animations.
- **CPU/GPU Scaling:** OS dynamically lowers CPU/GPU frequency when possible (DVFS) to save energy. Idle cores are power-gated.

2018:
https://www.theverge.com/2018/5/8/17327104/android-battery-management-ai-artificial-intelligence-google-io-2018
Android is now using AI to help manage your battery life

Charging-Aware Scheduling

- **Battery Charging State:** Tasks can be deferred until the phone is charging. Research shows scheduling intensive jobs (updates, backups) during charge increases net energy gain.
- **Study Insight:** Elmalaki et al. propose *charging-aware* schedulers: deferring tasks can improve battery availability by *~18.9%* for many users usenix.org.
- **Implementation:** OS or apps can check isCharging() and then perform updates or sync. This ensures energy is "free" and does not reduce battery run-time.



(a) App runs while phone is charging. (b) App runs after unplugging phone. (c) Energy gained by the battery.

Figure 7: Effect of running an app during vs. after the charging period.

usenix.org : 4

Things that can be done in the **newest Android version (14, 15)**

User actions

Battery Saver / Extreme Battery Saver
– throttles CPU, dims screen and freezes most apps when power is low [1]

Adaptive Battery
– ML learns your habits and limits rarely-used apps' wake-ups and network [2]

Adaptive Charging or "Limit to 80 %"
– slows or caps charging to protect long-term battery health [3]

Manually restrict or hibernate high-drain apps in Settings ▸ Battery ▸ Battery usage [4]

[1] Google Pixel Help – Use Battery Saver
[2] Android Dev – App Standby & Adaptive Battery
[3] Google Pixel Help – Charging optimization
[4] Google Pixel Help – Fix battery drain

OS automatic

Doze mode
– suspends network/CPU while idle, waking briefly for maintenance windows [5]

App Standby Buckets & Adaptive Battery quotas
– buckets apps (active→rare) and cuts background jobs [6]

Adaptive Refresh Rate
– drops screen as low as 1 Hz when static to cut display power draw [7]

App Hibernation
– after months unused, revokes permissions & stops jobs/notifications [8]

[5] Android Dev – Doze & App Standby
[6] Android Dev – App Standby Buckets
[7] Android Open Source Project – Adaptive Refresh Rate
[8] Android Dev – App hibernation

Things that can be done in the **newest iOs Version (18)**

User actions

Low Power Mode
– disables 5G, lowers refresh rate and pauses background fetch when battery is low [1]

Optimized Battery Charging / 80 % limit
– learns routine and delays charging past 80 % to slow aging [2]

Clean Energy Charging
– tops-off when your grid is using cleaner energy (US-only) [3]

Tame heavy apps
– turn off Background App Refresh & Live Activities for power-hungry apps [4]

OS automatic

ProMotion adaptive refresh
– drops display to 1 Hz when static, rising to 120 Hz only when needed [6][7]

BGTaskScheduler
– coalesces background jobs and network traffic when the device is on power/Wi-Fi [8]

Dynamic performance management
– iOS scales CPU/GPU and disables 5G in low-power or thermal events [4]

Thermal & idle clean-up
– iOS offloads maintenance like indexing to charging times to spare battery [4]

[1] Apple Support – Use Low Power Mode
[2] Apple Support – Optimized Battery Charging & Charge Limit
[3] Apple Support – Clean Energy Charging
[4] Apple Support – Charge & maintain your iPhone battery

[5] Apple User Guide – Low Power Mode (iOS 18)
[6] Apple Tech Specs – ProMotion (iPhone 15 Pro)
[7] Apple Dev – Optimizing ProMotion refresh rates
[8] WWDC Session – Advances in App Background Execution

AI-Driven Battery Management (Adaptive Battery)

- **On-Device ML:** Android's *Adaptive Battery* uses machine learning to learn user habits and restrict seldom-used apps. Google reports ~30% reduction in CPU wake-ups from this feature deepmind.google.
- **App Prioritization:** It anticipates which apps will be used soon and permits only those to run background work, putting others to sleep. This personalization saves energy.
- **Adaptive Brightness:** Similarly, *Adaptive Brightness* (DeepMind) learns preference curves to minimize wasted backlight power across environments.

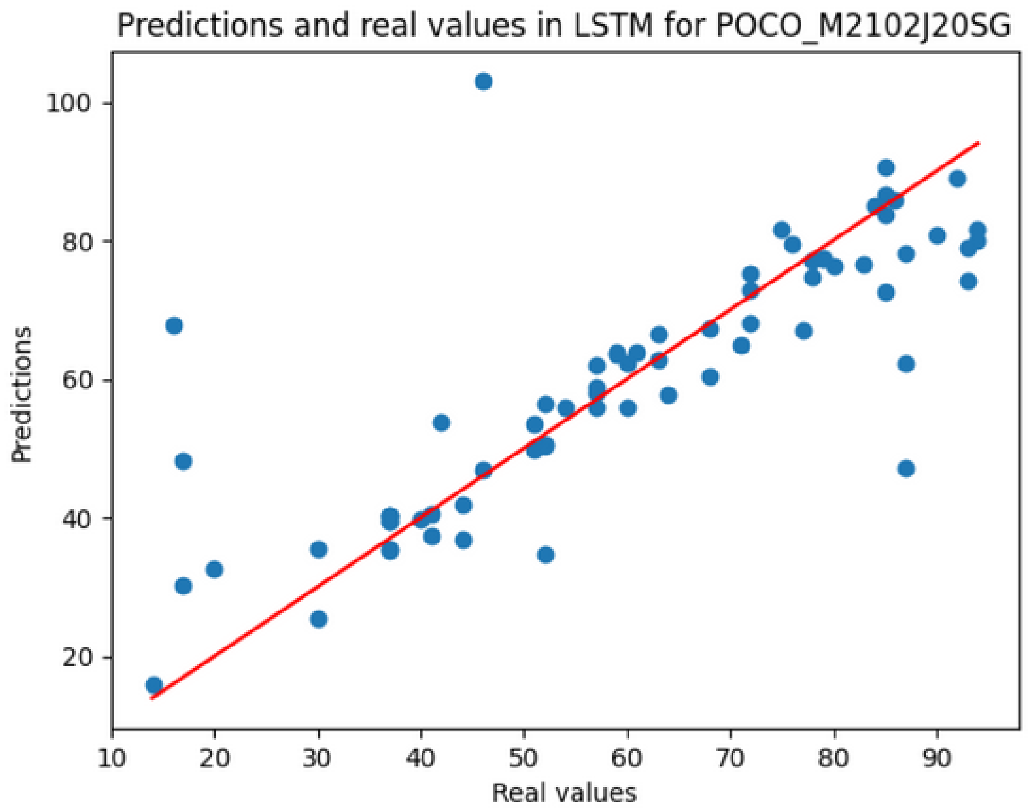
Personalized Usage Prediction

- **Battery Life Prediction:** ML predicts battery drain under current conditions. For example, Flores-Martin et al. use deep learning on per-user data (screen time, apps, temperature) to forecast remaining battery mdpi.com

- **Contextual Optimization:** By identifying which factors (apps in use, brightness, network) most impact drain, the system can offer specific advice or automatically tweak settings.

- **Privacy and Scalability:** Models run locally on-device for privacy. Their approach is *user-specific*, continuously adapting to individual patterns
mdpi.com[mdpi.com](mdpi.com)

Figure 14. LSTM predictions for POCO_M2102J20SG considering the predictions and real values.



mdpi.com : Figure 14

2021!

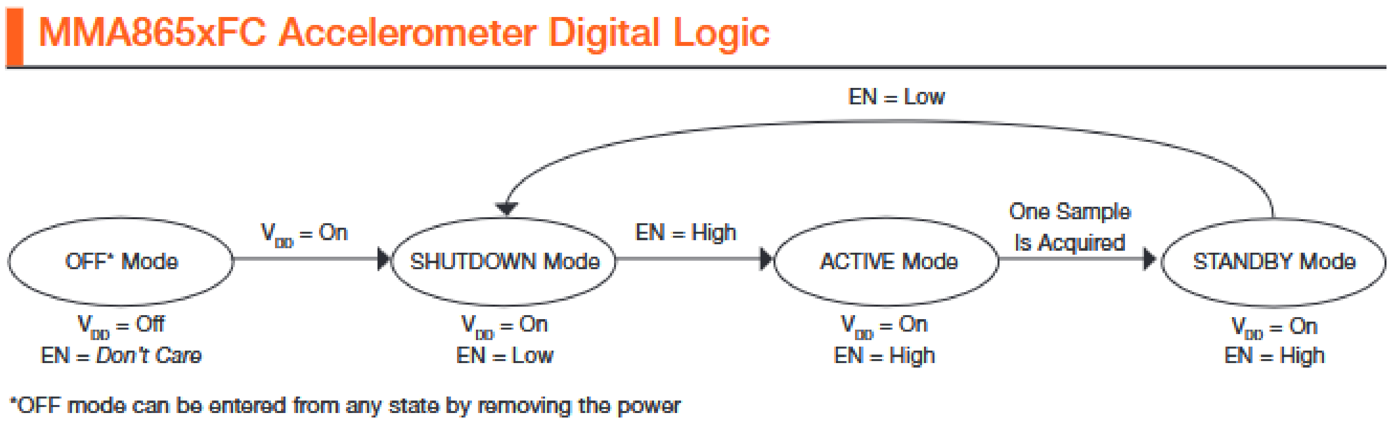


Figure 1: The operating modes of a sensor design with extremely low power consumption enable system designers to achieve specific design goals

Low-Power Hardware Components

- **Sensor Hubs & Co-processors:** Modern phones include dedicated low-power chips. For instance, Apple's M-series motion coprocessor handles step counting and GPS in the background without waking the main CPU

- **Context-Aware Sensing:** Accelerometers and gyros can detect *inactivity* (no movement) and signal the system to enter deep sleep nxp.com. For example, when a phone is stationary, the chip can cut power to big components.

- **Big.LITTLE CPU Architectures:** Many SoCs combine high-power and low-power cores, running background tasks on the latter. Unused cores are power-gated off.

Display & Power Electronics

- **Display Technology:** OLED screens use less power showing black pixels. Enabling dark mode or adaptive backlight can cut display energy.

- **PWM and Sensors:** High refresh rates or brightness consume more power. Using ambient light sensors to dim when possible saves energy.

- **Voltage/Frequency Scaling:** CPUs/GPU actively adjust voltage and clock rate based on load (DVFS), which hardware enables. Unused circuits are turned off (power gating).

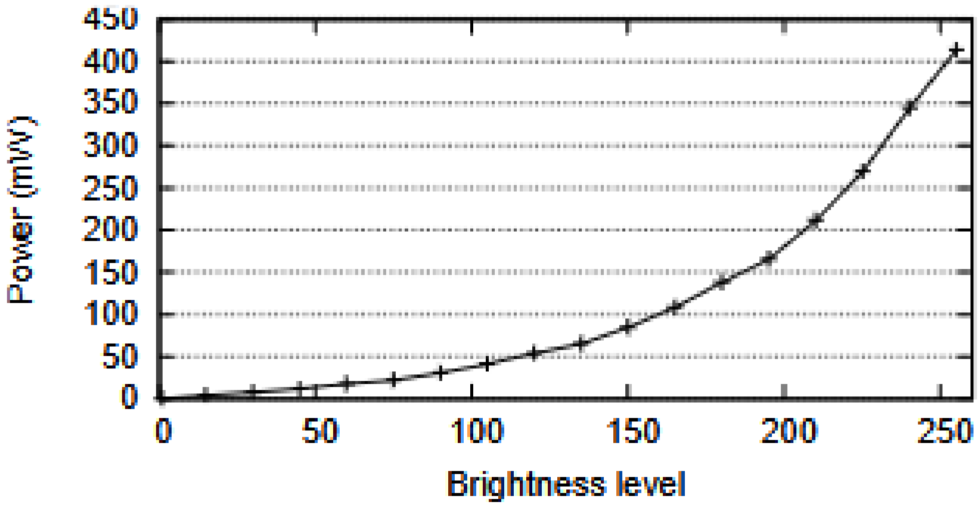


Figure 4: Display backlight power for varying brightness levels.

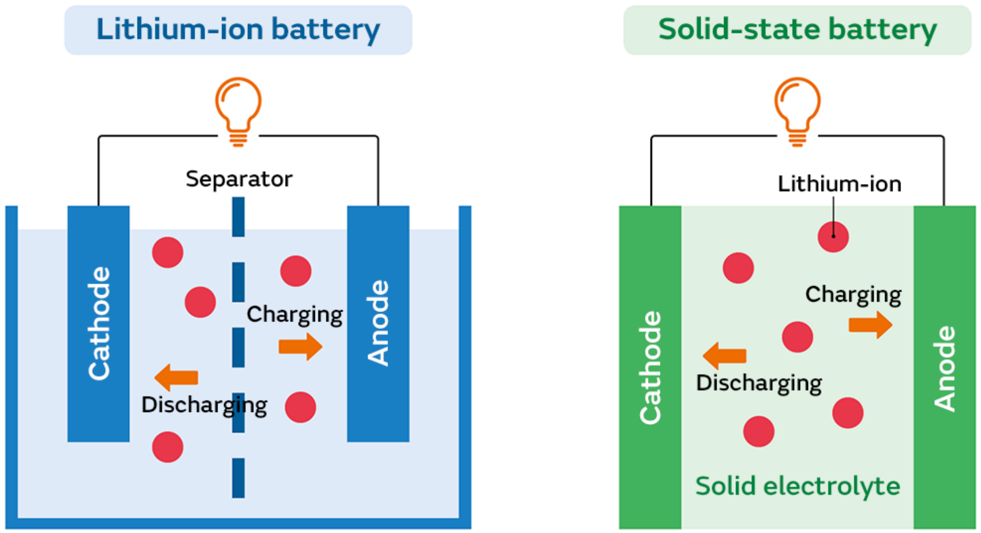
usenix.org : 5

Next-Gen Battery Technologies

- **Higher Energy Density:** Research into **solid-state batteries** promises much greater capacity. For example, Samsung's new solid-state cell achieves ~200 Wh/L, akin to larger Li-ion batteries notebookcheck.net. In future, phones may hold significantly more energy in the same volume.

- **Fast Charging & Preservation:** Intelligent charging algorithms (learned from EV tech) can reduce stress on batteries. Some phones slow charge overnight to extend battery lifespan.

- **Alternative Sources:** Integrating small solar panels or kinetic chargers can provide supplemental energy, though currently niche.



Summary & Future Directions

- **Holistic Approach:** Effective battery optimization uses **all levels** – hardware design, OS policies, ML, and user engagement. Each layer contributes.
- **Software Priority:** OS and application optimizations can yield large gains *today*, without new hardware. Background scheduling, charging-aware tasks, and adaptive algorithms already improve life significantly.
- **Emerging Trends:** Continued AI integration (e.g. predictive power capping), **edge computing** to offload heavy tasks, and new battery chemistries will shape the future.
- **Takeaway:** By leveraging smarter software and informed user habits, we can maximize each charge's value and extend device uptime

mdpi.comwww2.eecs.berkeley.edu.

Demonstration of Software-Implementation in **React Native / + Expo**

AGENDA

- Battery State

- Scheduling / conditional work

Battery State

- `react-native-device-info` or `react-native-battery` for level & charging events
- Expo: `expo-battery` exports identical API

- Subscribe to listeners to react to state changes
- Use data to gate energy-intensive features:

Use data to gate energy-intensive features

Category	What you can query in React Native / Expo
Battery	• Level (%), charging / discharging state (react-native-device-info, react-native-battery, expo-battery) • Low-Power / Power-Saver mode flag (react-native-device-info.iOS: isLowPowerMode())
Thermal / performance	• iOS thermal state (react-native-device-info → getThermalState()) – returns Nominal/Serious/Critical • Android battery temperature via getBatteryTemperature()
Power source	• Is device on external power (isBatteryCharging()); you can choose to run work only while charging
Network context	• Connection type & metered / low-data mode (@react-native-community/netinfo) – defer large downloads on cellular/Low-Data
App state & interaction	• Foreground/background & screen focus (AppState, React Navigation) • “User idle” timer or screen-off (Android react-native-screen-brightness)
Time & routine	• Local time / timezone (JS Date); run nightly jobs when the user is unlikely to need the app

Use data to gate energy-intensive features:

Combine several gates, e.g.:

```
if (
    level > 0.5 &&
    isCharging &&
    !lowPower &&
    connectionType === 'wifi' &&
    thermal === 'Nominal’
){
    runHeavySync();
}
```

Scheduling / conditional work

`react-native-background-fetch` wraps Android WorkManager & iOS BGTaskScheduler

- Configure: minimumFetchInterval, requiresCharging, etc.

- OS picks execution window; no exact background timers

- Foreground timers pause when app is backgrounded

- Local notifications fire at set times, but code runs only when user taps

Scheduling / conditional work

```
/* ---------- 1. periodic background task ---------- */

import BackgroundFetch, {
  HeadlessEvent,
  BackgroundFetchConfig
} from 'react-native-background-fetch';
import {AppState} from 'react-native';
import notifee, {
  TimestampTrigger,
  TriggerType,
} from '@notifee/react-native';

/* heavy job you only want when device is happy */
const syncStuff = async () => {
  /* …network / CPU work… */
};

/* configure native scheduler
   • Android → WorkManager
   • iOS     → BGTaskScheduler                    */

const cfg: BackgroundFetchConfig = {
  minimumFetchInterval: 15,      // minutes, OS ≥ 15
  stopOnTerminate: false,        // survive swipe-quit
  enableHeadless: true,
  requiresCharging: true,        // run only while charging
};

// called whenever OS decides it's OK to run
const onFetch = async (taskId: string) => {
  await syncStuff();
  BackgroundFetch.finish(taskId);
};

BackgroundFetch.configure(cfg, onFetch, err =>
  console.warn('BGFetch err', err),
);
```

← PERIODIC BACKGROUND TASK

← NATIVE SCHEDULER

```
/* ---------- 2. foreground timer (pauses in bg) ---------- */

let tick: NodeJS.Timeout | undefined;

const startInterval = () => {
  tick = setInterval(() => console.log('active tick'), 10_000);
};
const stopInterval = () => tick && clearInterval(tick);

AppState.addEventListener('change', s =>
  s === 'active' ? startInterval() : stopInterval(),
);

/* ---------- 3. exact wall-clock notification ---------- */

export const scheduleReminder = async (date: Date) => {
  const trigger: TimestampTrigger = {
    type: TriggerType.TIMESTAMP,
    timestamp: date.getTime(), // fires even in Doze (Android alarmManager flag)
  };
  await notifee.createTriggerNotification(
    {title: 'Tap to run job', body: 'Opens app; code runs only after tap'},
    trigger,
  );
};

/* ---------- 4. headless entry (Android only) ---------- */

export const backgroundFetchHeadless = async (event: HeadlessEvent) => {
  if (event.timeout) return BackgroundFetch.finish(event.taskId);
  await syncStuff();
  BackgroundFetch.finish(event.taskId);
};
BackgroundFetch.registerHeadlessTask(backgroundFetchHeadless);
```

← FOREGROUND TIMER

← NOTIFICATION TRIGGER

← ANDROID: headless background entry

Thank you for your attention

Bibliography

[1] K. Shankari, D. E. Culler, R. H. Katz, "Doing Nothing Well: OS-Application Coordination for Energy Saving," **UC Berkeley EECS Tech. Rep. UCB/EECS-2016-119**, June 2016 [www2.eecs.berkeley.eduwww2.eecs.berkeley.edu](www2.eecs.berkeley.edu). Access: 30.04.2025, Martin Lauterbach

[2] K. Athukorala et al., "How Carat Affects User Behavior: Implications for Mobile Battery Awareness Applications," *Proc. ACM CHI*, 2014 [amplab.cs.berkeley.eduamplab.cs.berkeley.edu](amplab.cs.berkeley.edu). Access: 30.04.2025, Martin Lauterbach

[3] A. Carroll, G. Heiser, "An Analysis of Power Consumption in a Smartphone," *Proc. USENIX ATC*, 2010 [usenix.orgusenix.org](usenix.org). Access: 30.04.2025, Martin Lauterbach

[4] S. Elmalaki et al., "A Case for Battery Charging-Aware Power Management and Deferrable Task Scheduling in Smartphones," in *Proc. HotPower*, 2014 usenix.org. Access: 30.04.2025, Martin Lauterbach Access: 30.04.2025, Martin Lauterbach

[5] D. Flores-Martin, S. Laso, J. L. Herrera, "Enhancing Smartphone Battery Life: A Deep Learning Model Based on User-Specific Application and Network Behavior," *Electronics*, vol. 13, no. 24, 4897, 2024 [mdpi.commdpi.com](mdpi.com).

[6] J. Smith, S. Rosen, C. Gamble, "DeepMind, meet Android," *Google DeepMind Blog*, May 8, 2018 deepmind.google. Access: 30.04.2025, Martin Lauterbach

[7] Freescale Semiconductor (NXP), *"Low-Power Sensing"* White Paper, 2010 [nxp.com](nxp.com). Access: 30.04.2025, Martin Lauterbach

[8] N. Ali, "Samsung's solid-state batteries: A new era for wearable tech with 200 Wh/L energy density," *NotebookCheck News*, Jan. 2024 notebookcheck.net. Access: 30.04.2025, Martin Lauterbach