



Example 5 — Battery Monitor (Observer)

Story

A device reports **battery level (0–100%)** and whether it's **charging**. Multiple components must react automatically when these values change (HUD, alerts, power saver, etc.).

Entities

- **Subject:** BatterySensor
- **Observers (examples):**
 - BatteryHUD — shows level + charging state.
 - LowBatteryAlert — warn when level drops below a threshold.
 - SaverModeController — toggles power-saving below a threshold, restores above.
 - ChargeCompleteNotifier — fires when battery reaches 100% while charging.

Data Model (push model for this example)

- Event payload: (level:int, isCharging:bool)
- Valid level: **clamp to [0, 100]** in the subject.
- Notify **only when a value actually changes** (level or charging).

Subject API (suggested)

- attach(observer*), detach(observer*), removeAll()
- setLevel(int), setCharging(bool)
(Both call notify() **only if** the new value differs.)
- notify() uses a **snapshot** of observers (so self-detach is safe).
- **Duplicate attach guard** (ignore repeats).
Tip: keep it single-threaded, beginner-friendly. We'll add threads later as an assignment.

Observer API (suggested)

- onBatteryUpdate(int level, bool isCharging)

Behavioral Rules

1. **Change-driven:** no notification if setLevel receives the same level and setCharging is unchanged.
2. **Ordering:** when both change (e.g., plug in and level jumps), apply updates then call one notify().
3. **Hysteresis (for alerts):** to avoid spam:
 - LowBatteryAlert triggers when level **falls below** lowThresh (e.g., 20%),
 - clears when level **rises above** lowThresh + 2% (simple hysteresis).
4. **Edge cases:**
 - Level at 0% triggers "critical" variant in LowBatteryAlert (optional).
 - **ChargeCompleteNotifier** should fire when isCharging==true **and** level hits 100% (fire once until it drops below 100 again).
5. **Lifecycle:** observers may attach/detach at any time; removeAll() clears all.