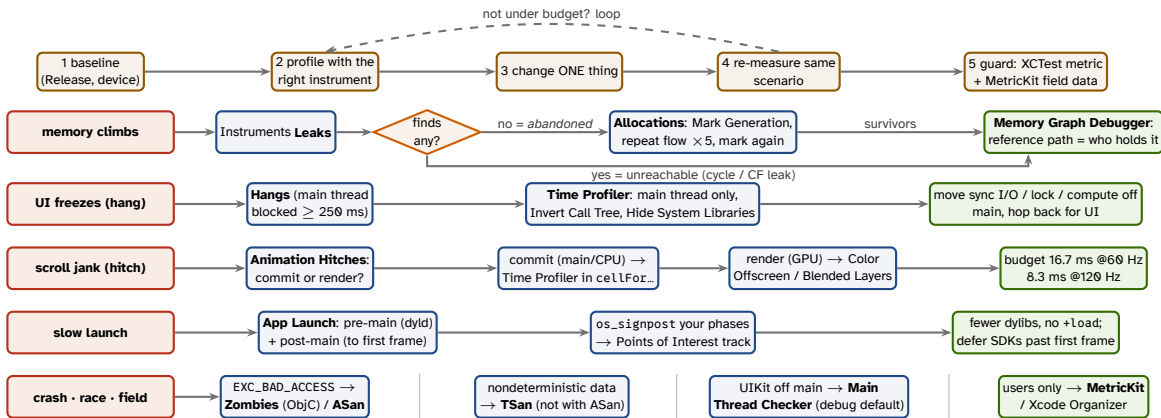


In one line: Measure, don't guess: baseline on a **real device** in the **Release/Profile** config → pick the instrument that answers *this* symptom → change **one** thing → re-measure the same scenario. Interviewers grade whether you can **name the tool, read its number**, and say what it *cannot* see.



How it works — the tools, precisely

- Memory Graph Debugger** (Xcode debug bar, *not* Instruments): pauses the app, snapshots every live object and its references. Purple ! = leak/cycle; select an object → *who holds it*. Snapshot only, no timeline. Enable **Malloc Stack Logging** (scheme → Diagnostics) to get the *allocation backtrace*.
- Leaks** (Instruments): periodically scans for **unreachable** blocks — cycles, un-released CF. **Blind to abandoned memory** (still reachable: unbounded Dictionary cache, observer array).
- Allocations**: every heap alloc over time. **Persistent vs Transient; Mark Generation** (a.k.a. heapshot) — objects created after a mark and still alive show under that generation. Stepwise growth per repetition = abandoned or leaked.
- Footprint ≠ live bytes**: Jetsam kills on *footprint* (dirty pages, decoded bitmaps). 4000×3000 image = 48 MB decoded, any JPEG size.
- Time Profiler** samples every thread's stack (~1 ms): statistical, cheap, no call counts. Read it: *Separate by Thread* → main thread, *Invert Call Tree*, *Hide System Libraries*; chase high **self** weight (total includes callees). 80 % CPU on a background thread is harmless; 300 ms of sync I/O on main is a hang.
- Hang** = main thread not servicing the run loop (Hangs instrument, MXHangDiagnostic). **Hitch** = frame presented late. Different metric, different fix. **Thread Performance Checker** flags priority inversions at runtime; **System Trace** shows *why* a thread is not running (lock, page fault).
- Hitch time ratio** = hitch ms per second of scroll: < 5 ms/s good, 5–10 warning, > 10 critical. Commit hitch = your layout/decode on main; render hitch = offscreen passes (masksToBounds+radius, shadow w/o shadowPath), blending.
- Launch**: pre-main = dyld loads/binds dylibs, ObjC setup, static initializers; post-main = main → didFinishLaunching → first frame. Measure *cold* launch; the watchdog (0x8badf00d) kills a stuck launch. CI: measure(metrics: [XCTApplicationLaunchMetric()]).
- MetricKit**: MXMetricPayload (launch, hang rate, memory, hitches — aggregated ~daily) + MXDiagnosticPayload (hang/crash/CPU/disk-write stacks). Tells you *which* problem to bring into Instruments.

Example — signpost + MetricKit

```

import os; import MetricKit
let sp = OSSignposter(subsystem: "app.feed",
                     category: .pointsOfInterest)

func loadFeed() async throws -> [Post] {
    let s = sp.beginInterval("Feed", id: sp.makeSignpostID())
    defer { sp.endInterval("Feed", s) } // ~free unless recording
    return try await api.fetchFeed()
}

final class Field: NSObject, MXMetricManagerSubscriber {
    func didReceive(_ p: [MXMetricPayload]) {} // aggregates
    func didReceive(_ p: [MXDiagnosticPayload]) {} // hang stacks
}
MXMetricManager.shared.add(field) // keep 'field' alive
  
```

Interview traps

- ! **"Leaks says 0, so no leak"** — memory still climbs = *abandoned* memory. Allocations generations, then Memory Graph for the holder.
- ! Memory Graph lives in **Xcode**, Leaks/Allocations in **Instruments**. Name all three (you answered "don't know").
- ! Profiling **Debug** or the **Simulator**: no inlining, Mac CPU — wrong hot spots. Profile scheme, slowest supported device. NSLog timing perturbs the run — use `os_signpost`.
- ! Dealloc test: `XCTAssertNil` inside `DispatchQueue.main.async` asserts *nothing* — the test already returned. Assert directly or await an expectation.
- ! Zombies: ObjC objects only, never frees. ASan + TSan: not in one run.
- ! "16 ms per frame" on a ProMotion phone: it is **8.3 ms**.

Remember

Growth: **Leaks** → **Allocations** → **Graph**. Freeze: **Hangs** → **Time Profiler** (main, inverted). Jank: **Hitches** → **commit or render**. Field: **MetricKit**. Always: *one change, re-measure*.

Likely questions

- Leak vs abandoned? — unreachable vs reachable-unused.
- Time Profiler? — samples stacks ~1 ms; samples ≈ time.
- Hang vs hitch? — main thread blocked vs frame late.
- Bench fine, users slow? — MetricKit; test the floor device.
- Pre-main cost? — dyld: dylibs, bind, +load, static inits.