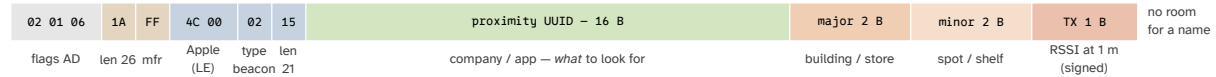


In one line: Apple's proximity features are **different frameworks on the same radio**: iBeacon is a BLE advert read by **Core Location** (not CoreBluetooth); **AccessorySetupKit** (iOS 18) pairs one accessory without the broad Bluetooth permission; **Nearby Interaction** measures distance + direction over **UWB** after BLE hands over tokens; **Find My** is a crowd relay of encrypted locations keyed by rotating BLE keys. Plain BLE GATT needs **no MFi**.

an iBeacon advert — 30 of the 31 bytes



Nearby Interaction: BLE bootstraps, UWB measures



3 NINearbyObject: distance (m) + direction (unit vector, may be nil)

Find My: offline finding, end-to-end encrypted



Apple and the finder cannot read it; rotation stops anyone linking adverts to one tag

How it works

- iBeacon = Core Location.** `CLBeaconRegion + CLBeaconIdentityConstraint(uuid:major:minor:)` (iOS 13; iOS 17: `CLMonitor`). **Monitoring** = enter/exit, OS-level, relaunches a terminated app (needs **Always**); max **20** regions per app, geofences included. **Ranging** = `startRangingBeacons(satisfying:)`, foreground: proximity `.immediate/.near/.far` + accuracy (m, from RSSI). CoreBluetooth hides iBeacon adverts. **Sending** one: `peripheralData(withMeasuredPower:)` → `CBPeripheralManager`, foreground only.
- AccessorySetupKit** (iOS 18) — `ASAccessorySession .showPicker(for:)` with `ASPickerDisplayItem` (name, image, `ASDiscoveryDescriptor: service UUID / company id / name`). The **system** scans and shows the sheet; the app gets an `ASAccessory` (`bluetoothIdentifier`) and uses CoreBluetooth for *that* accessory only — no global Bluetooth prompt. Info.plist: `NSAccessorySetupKitSupports` + descriptors.
- Nearby Interaction** (iOS 14) — U1 (iPhone 11–14) / U2 (iPhone 15+), AirTag, some Watches; **both** sides need UWB. Tokens are `NSData`: archive, send, `NINearbyPeerConfiguration(peerToken:)`. Third-party UWB chips: `NINearbyAccessoryConfiguration` (iOS 15). Check `NISession.deviceCapabilities` (iOS 16) — direction may be missing. Foreground-oriented; a paused session needs fresh tokens.
- Find My network** — third-party trackers join through the **MFi** Find My network accessory program (Apple's spec). No public app API to read other people's reports.
- MFi** — only for `ExternalAccessory` (iAP2 over Classic BT / Lightning / USB, `UISupportedExternalAccessoryProtocols`), Find My and similar programs. A BLE GATT peripheral (our ESP32s) needs **none**.
- Beyond BLE** — `MultipeerConnectivity` (`MCSession`: Wi-Fi, peer-to-peer Wi-Fi, Bluetooth; local-network usage string + `NSBonjourServices`); Wi-Fi Aware (iOS 26). HomeKit: entitlement + usage string; Matter (iOS 16.1+): `MatterSupport`, CSA-certified devices.

Remember

Beacon = Location, Accessory = ASK then CB, Precise = UWB after BLE, Lost = Find My relay, MFi = Classic/Find My only.

Which framework, what it needs

Job	Framework	Needs
GATT to a peripheral detect iBeacon	CoreBluetooth Core Location	Bluetooth usage string; background modes location auth (Always to relaunch); no BT prompt
be an iBeacon onboard 1 accessory distance + direction	CL payload + CB AccessorySetupKit NearbyInteraction	BT permission, app in foreground iOS 18, Info.plist descriptors; then CB UWB both sides, NI usage string, a token channel
lost-item relay classic / Lightning	Find My network ExternalAccessory	MFi program; hardware MFi + protocol strings

Example

```
let c = CLBeaconIdentityConstraint(uuid: hubUUID, major: 1)
let region = CLBeaconRegion(beaconIdentityConstraint: c,
                             identifier: "hub")
manager.requestAlwaysAuthorization() // relaunch on entry
manager.startMonitoring(for: region) // background, 20 max
func locationManager(_ m: CLLocationManager,
                     didEnterRegion r: CLRegion) {
    m.startRangingBeacons(satisfying: c) } // detail, fg
// UWB: my token out, the peer's token in, over GATT
let blob = try NSKeyedArchiver.archivedData(
    withRootObject: ni.discoveryToken!, requiringSecureCoding: true)
ni.run(NINearbyPeerConfiguration(peerToken: peerToken))
```

Interview traps

- ! Ranging iBeacons with `CBCentralManager` — Core Location's job; CB hides them.
- ! “iBeacon gives me distance” — RSSI buckets, metres of error. Precision = UWB.
- ! Ranging in the background — monitoring only; ranging is foreground.
- ! “ASK replaces CoreBluetooth” — it replaces *discovery* + *permission*; GATT is still CoreBluetooth.
- ! UWB without a side channel — it has no discovery; tokens travel over BLE/network.
- ! “MFi for my BLE sensor” — no. And an iOS advert in the background drops the name and moves **all** service UUIDs to the overflow area.

Likely questions

- Wake an app near a hub? — beacon region monitoring (Always).
- Onboard without the BT prompt? — `AccessorySetupKit`.
- Find My privacy? — rotating keys; encrypted to the owner.