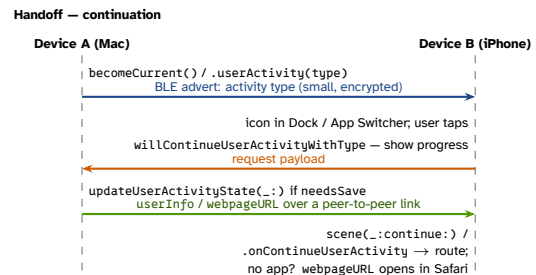
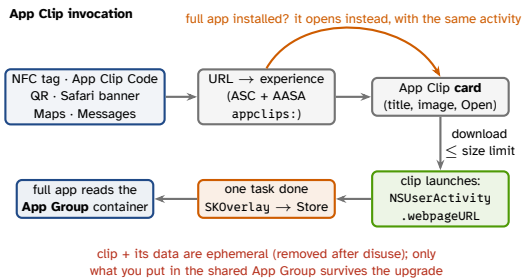


In one line: Both ride on `NSUserActivity`. An **App Clip** is a small, separately built slice of your app that a **URL** (from a code, tag, link or map) launches *without installing the app*; **Handoff** moves *your own* current activity to *your own* nearby device (same iCloud account, BLE + Wi-Fi). Neither is a deep link, and neither is state restoration.



App Clips — how it works

- **Target:** own bundle id (parent id + suffix, e.g. `.Clip`), shipped with the full app; share code via packages, strip the rest with your own `#if APPCLIP` compilation condition.
- **Size** (uncompressed, thinned): **10 MB** (iOS 14–15), **15 MB** (iOS 16+), **100 MB** (iOS 17+, digital invocations only — no physical codes, NFC, QR). Third-party SDKs eat it.
- **Experiences** (App Store Connect): default (one card) + **advanced** per URL prefix / location. Domain proof: `appclips:` associated domain + AASA. Launch: `webpageURL` of a `NSUserActivityTypeBrowsingWeb` activity = your routing key.
- **Limits:** no background execution, restricted frameworks (e.g. `HealthKit`), no cross-app tracking. **Ephemeral notifications:** `NSAppClipRequestEphemeralUserNotification` → up to **8 hours** without a prompt. `NSAppClipRequestLocationConfirmation + APActivationPayload.confirmAcquired(in:)` verifies the user is really at the code.
- **Upgrade:** `SKOverlay.AppClipConfiguration` (`.appStoreOverlay` in `SwiftUI`); the full app inherits only the **App Group** container. Test: `_XCAppClipURL` scheme variable; on device, Settings › Developer › Local Experiences.

Handoff & Continuity — how it works

- **Advertise:** `activityType` listed in `NSUserActivityTypes`; small plist `userInfo` (+ required `userInfoKeys`), `title`, `webpageURL`; `isEligibleForHandoff` (also `ForSearch`, `ForPrediction`). Current via `becomeCurrent()` or a responder's `userActivity`; refresh lazily with `needsSave + updateUserActivityState(·)`.
- **Requirements:** same Apple ID / iCloud, BLE + Wi-Fi, Handoff on, same **Team ID** and activity type in both apps. Big data: continuation streams (`supportsContinuationStreams`) or the cloud.
- **Continue** per *scene*: cold start → `connectionOptions.userActivities`; running → `scene(·:continue:)`.
- **Web↔native:** app → Safari via `webpageURL`; Safari → app needs the `activitycontinuation`: associated domain (Universal Links use `applinks:`).
- **Universal Clipboard:** `UIPasteboard` items reach nearby devices; keep secrets local with `setItems(·:options: [.localOnly: true, .expirationDate: ...])`. **Continuity Camera:** iPhone as Mac camera/scanner — nothing to build on iOS.

```
ArticleView(a)
.userActivity("com.acme.article") { act in // advertise
  act.userInfo = ["id": a.id] // what, not data
  act.webpageURL = a.webURL // Safari fallback
  act.isEligibleForHandoff = true }
.onContinueUserActivity("com.acme.article") { act in
  if let id = act.userInfo["id"] as? String { path.append(id) }
}
// clip AND full app: the same entry point
.onContinueUserActivity(NSUserActivityTypeBrowsingWeb) { act in
  if let url = act.webpageURL { route(url) } }
```

Four ways back into a screen

Handoff	same user, <i>other nearby device</i> , in-progress activity
Deep link	a URL anyone can send; opens the installed app (else web)
State restor.	<i>same device</i> after relaunch (<code>stateRestorationActivity</code> , <code>@SceneStorage</code>)
App Clip	a URL when the full app is <i>not</i> installed

Interview traps

- ! Quoting “10 MB” without the OS version, or meaning the *download* size.
- ! Full app does not handle the clip's URLs — once installed, every code opens the *full* app.
- ! Clip and app share data freely — no: only an App Group; and the clip may vanish.
- ! Handoff activity never appears: type missing from `NSUserActivityTypes`, different Team ID, not current, non-plist `userInfo`, BLE off / other iCloud account.
- ! Megabytes in `userInfo`; or expecting Handoff to carry your app's login — it resumes context, not auth.
- ! Reading the pasteboard on launch — iOS 16 asks “Allow Paste”; use `UIPasteControl` / `detectPatterns`.

Remember

“Clip: a URL, one task, a slot in the App Group. Handoff: my activity, my other device, same iCloud.”

Likely questions

1. How does a clip know what was scanned? — `webpageURL` of a browsing-web `NSUserActivity`.
2. Keep a clip's cart after install? — write it to the shared App Group container.
3. Handoff vs Universal Link? — own activity to own device vs a URL for anyone.
4. Notify from a clip? — ephemeral permission, 8 hours per launch.