Quiz #2
ICS 224
09 March 2018

App Architecture

Legal - Info.plist
Legal requirements in every country your app is available - ie privacy rights.
Apple very few cycles to apps when closing, and in background almost none. Save fast!
Only 1 app runs at a time, battery saving measure.
Appdelegate file to move between foreground and background.
Cannot block main thread! - UI must be on main thread.

- DispatchQueue.Global().async for background task
- DispatchQueue.main.async for UI

HCI Guidelines

Apple strong about interfaces. Want consistent feel. User centric. Emphasis on look & feel for their entire ecosystem. Your app must conform.

Requirements(Guidelines)

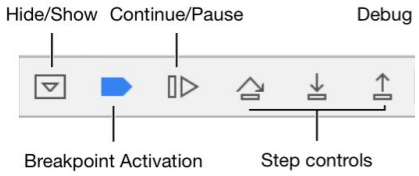
- Loading - Background load whenever possible (or progress bar) Better user experience
- Modality - Avoid dialog boxes. Always allow user to choose not to do something (cancel)
- Navigation - No mazes - let user know where in the app. Or index on left side.
- Onboarding - New user should get optional tutorial. Should work with custom setup optional
- Permissions - Necessary, must ask. OS will enforce. No unnecessary requests.
- Data Entry - Use date pickers etc. Less error prone. Minimal typing.
- Feedback - Dont annoy users for it.
- Buttons - Use standard icons, use verbs, brief

Review Guidelines

- Not for debugging - no buggy apps
- No “objectionable” content ie illegal drugs
- Need report abuse button
- No Beta Products - use TestFlight instead (alt store for dev)
- Only use Public API - Private calls not allowed
- Be original - No IP infringement
- Test locally on own hardware
- Deploy with Apple Developer Program $100 fee

Debugging, Testing & Performance

- Conditional breakpoints for high traffic code - breaks only when condition met.
- Blue arrow inside white arrow means conditional break
- Can specify not to halt program, just print
- Exception breakpoint to pause before exception to allow fiddling with variables



Step Over a fn, Step In, Step Out

Breakpoint Activation: If blue enabled, if grey disabled (no breakpoints will fire)
Debug variables: Auto(Best Guess, default), All or Local
OS_Log will allow retrieval during or after app execution - log dump accessed via Settings->Privacy -> Analytics -> Analytics Data -> Sys diagnose

Testing with XCTest (like JUnit)
Equivalence Sets - Grab a value from each set and test all boundary conditions
Ex 0 - 100 are valid. Test -5, 20 & 500 to represent each set. Then test -1, 0, 1 and 99, 100, 101 for the boundary conditions.
UI testing is possible but difficult.
Ensure all path coverage with conditionals.
Gauges for resource consumption. Mobile battery (cpu usage) and memory huge

Mobile Application Trends
Cross platform tools out there:

1. Facebook's Reactive Native
    a. Javascript, web feel. No compile -> slower. Can't access platform specific hardware, ie sensors. Not its purpose. Goal: 1 UI for both platforms. Generic apps.
2. Microsoft's Xamarin
    a. C# compiled. Open source, can access platform specific. But, seperate UI code per platform. Focus on business logic, not presentation. Consumes more memory in general.

Machine Learning (face recognition with photos)
Augmented Reality
WatchOS

Mobile Limitations:
Memory and Battery
Limited access to the hardware
Different controls