Hybrid Mobile App Development

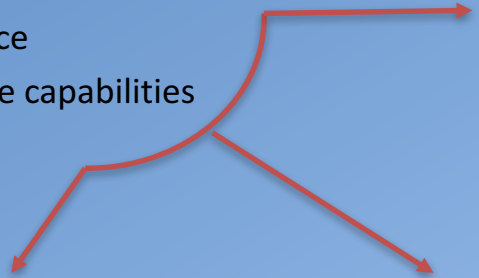
Jogesh K. Muppala

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

App Implementation Approaches

- Native Apps
  - Platform-specific skills
  - Highest performance
  - Full access to device capabilities



- Mobile Web Application
  - Fully hosted in the mobile browser
  - Slowest
  - No access to device capabilities

- Hybrid with WebView
  - Embedded web view based with partial implementation in native code
  - Slow, but comparable to native apps based on functionality
  - Some access to device capabilities

- Compiled/Interpreted/VM Hybrid
  - Makes use of Native UIs with the native platform's rendering engine, not WebViews
  - Near-native performance
  - Most access to device capabilities

Hybrid App Development Approaches

- WebView app
  - The HTML, CSS and JavaScript code base runs in an internal browser (called WebView) that is wrapped in a native app. Some native APIs are exposed to JavaScript through this wrapper
  - Examples: Ionic Framework + Cordova/Phonegap, Trigger.io
- Compiled/Interpreted/VM hybrid app
  - The code is written in one language (such as C# or JavaScript) and gets compiled/interpreted to native code or run on VM for each supported platform. The result is a native app for each platform, but less freedom during development
  - Examples: NativeScript, React Native, Appcelerator Titanium, Xamarin, Embarcadero FireMonkey

Hybrid Mobile App Development Frameworks

- Different types of frameworks aimed to build hybrid apps :
  - Frameworks targetting HTML5 content like Ionic + Cordova/Phonegap (both via JS byte code)
  - Frameworks like NativeScript and Appcelerator Titanium that render the UI using the platform's native controls but still working via JS
  - Free (or partially free) Frameworks aiming to produce real native code like Unity (C# orJS based, Games oriented), Kivy (Python Based) or libgdx (Java based, Game Oriented)
  - Commercial Frameworks aiming to produce real native code like Xamarin (using C#) or Embarcadero

Advantages of Hybrid Approach

- Developer can use existing skills
- One code base for multiple platforms
- Reduced development time and cost
- Easily design for various form factors (including tablets) using responsive web design
- Access to (some) device and operating system features
- Advanced offline capabilities
- Increased visibility because the app can be distributed natively (via app stores) and to mobile browsers (via search engines)

Drawbacks of Hybrid Approach

- Performance issues for certain types of apps (ones relying on complex native functionality or heavy transitions, such as 3D games)
- Increased time and effort required to mimic a native UI and feel
- Not all device and native features (fully) supported
- Risk of being rejected by Apple if app does not feel native enough (for example, a simple website)

Where Hybrid Apps Work Best

- Hybrid approach does not suit all kinds of apps
- Need to carefully evaluate your target users, their platforms of choice and the app's requirements.
- Mainly suitable for content-driven apps
  - Business and Productivity
  - Enterprise
  - Media