MOBILE APP DEVELOPMENT

Mobile Apps

A mobile app or mobile application is a computer program or software application designed to run on a mobile device such as a phone/tablet or watch. Apps are generally downloaded from application distribution platforms which are operated by the owner of the mobile operating system, such as the App Store (iOS) or Google Play Store.

Mobile App Development

There are three broad approaches to developing mobile applications: Web, Hybrid, and Native.

Figure 4: Mobile App Developing Approaches

Web application:

Mobile web development leverages the same skills and workflow traditionally associated with "desktop" web development. Developers build websites using HTML, JavaScript and CSS that are then accessed on mobile devices via mobile browsers.

There are two ways developers target apps for mobile devices with the web:

1. Responsive Web Design: With responsive web design (RWD), developers primarily focus on modifying the layout and display of existing desktop websites to adapt to the smaller screen size and touch-input of mobile devices.

Advantage

Single web code base for both desktop and mobile users.

Disadvantage

RWD is generally limited in its ability to create a “tailored” mobile experience that imitates the look – and - feel of native apps.

2. Mobile Web App: Alternatively, developers can build web-based experiences designed specifically for mobile users. In this scenario, mobile devices are usually detected and directed to a mobile optimized web app where developers can build tailored experiences that conform to mobile specific UI conventions.

Advantage

Developers can choose to make mobile web apps look and feel exactly like installable mobile apps.

Disadvantage

This approach does require developers to build and maintain separate view (HTML/CSS) implementations for both mobile and desktop clients.

Web based mobile Apps

Essential skills: HTML, JavaScript, CSS

Essential tools: Anything capable of developing web apps

Platform reach: iOS, Android, Windows Phone or any HTML5 capable mobile browser

Sharable cross-platform codebase: 100% (UI + Logic)

PROS

- Familiar, very low developer learning curve
- Easy to deploy, no software installs
- Easy to share code with desktop websites
- Maximum reach
- Reuse existing security and software management solutions
- Open standards-based platform (no vendor lock-in)

CONS

- Limited access to device hardware, APIs

• Poor offline support, requires “always on” Internet connection

• Unable to “install” on a device or publish via an app store

• Unable to match native performance for rich, animated interfaces

Native application

As the name implies, native apps are built using platform-specific SDKs and development tools provided by the platform vendors.

For iOS, that means apps are built using Objective C in Apple’s XCode.

For Android, that means apps are built using Java and Google’s Android SDKs.

For Windows Phone, .NET and Visual Studio are the application building environment.

Every platform has its own SDKs, and often, its own programming language.

Essential skills: Objective-C, Java, .NET, HTML/JavaScript

Essential tools: XCode (for iOS), Eclipse (for Android), Visual Studio (for WinPhone)

Platform reach: Each app only reaches one platform

Sharable cross-platform codebase: 0% (No UI, No logic)

PROS

• Complete access to device hardware, APIs

• Installable, can be app store deployed

• Maximum control over performance

• Powerful platform-specific development and debugging tools direct from platform vendors

CONS

• Multiple implementations required to reach multiple platforms

• Multiple skill sets and programming languages

• Requires installation (and device provisioning if private deployment desired)

• New tools needed to manage app security, enforce data security policies

Hybrid Application

Hybrid attempts to blend the benefits of web and native mobile app development. Hybrid apps are developed using standard web technologies, like HTML, JavaScript and CSS, but are able to overcome the limits of "pure" web apps by using platform-specific native "wrappers" to package and deploy the code. The native wrappers allow hybrid apps to be installed on devices, deploy via app stores and access native device APIs via JavaScript.

Essential skills: HTML, JavaScript, CSS, Hybrid container (such as Apache Cordova)

Essential tools: Anything used for web development + hybrid SDKs

Platform reach: Limited to reach of hybrid container, but most reach all major platforms

Sharable cross-platform codebase: Almost 100% (Some platform specific UI may be desired)

PROS

• Low learning curve for web developers

• Installed, can be app store deployed

• One code base for all platforms

• Easy to transition from web to hybrid development, reuse code

• Extensive access to device hardware, APIs

CONS

• Performance limited by web's capabilities

• Requires installation (and device provisioning if private deployment desired)

• New tools need to manage app security, enforce data security policies