Choosing the best programming language for mobile app development

By Ajay Chebbi - Published July 2, 2019

When considering the programming languages, frameworks, and SDKs for mobile apps, you have to consider the frontend (UI) development environment but also be aware of the backend (server-side) development environment. The developers who are coding the frontend are often not the ones who are coding the backend, but they do usually work with each other (they are in enterprises anyway). In this article, we'll review today's popular programming languages and development frameworks for developing mobile apps and mobile backends.

Types of mobile apps

From a coding perspective, developers can choose to create one of three types of mobile apps:

- **Native mobile apps** are apps that are written in a language that's natively supported by the device OS vendor. The underlying platform APIs are fully available to the app code and the OS provides the UI component library. The build process converts this code into an executable app with native bytecode of the OS.
- **Cross-platform mobile apps** are apps that are written in one language and can run on all the platforms. With cross-platform mobile apps, the language in which you write your app may not be the native language of the underlying device OS. This implies that the code might be contained, bridged, or cross-compiled to the bytecode that can execute directly on the OS with the help of some support libraries.
- **Mobile web apps** are a lightweight paradigm where you open a URL in your device's browser, but they look and feel like a regular mobile app. Mobile websites are not delivered as an installed app on the device, but they are an important paradigm to consider when building for a mobile device. Developers can use two web site programming paradigms to create mobile web apps:
    - **Responsive web programming** is a style of web site programming where the web site layout organizes itself to suit the device's form factor. So instead of having to browse horizontally, the content "snaps" in place to fit the width of the device. These web sites can also be used from a desktop.
    - **Progressive web programming** creates progressive web apps that are normal websites that appear like a traditional mobile app when navigated to from a browser on a mobile device. They deliver an almost app-like experience. Progressive web apps have the ability to add a link to the home screen (so that you get a app icon), and they also have the ability to run offline when not connected to the internet by caching the content locally on the device.

You can read more about the types of apps in this IBM Developer article, "Deciding between native and cross-platform mobile frontend programming frameworks."

Developing mobile apps for the iOS platform

The iOS platform is a proprietary platform made by Apple. The iOS platform consists is available for phone devices (iPhone) and tablet devices (iPad). You can develop apps for the iOS platform, and then target the same app to both an iPhone and an iPad. While developing the app, you have to account for the real estate constraints that are available on each of the devices. iOS apps can be installed on your iPhone or iPad from the Apple app store.

For building apps for iOS, you must have an Apple developer account and the Xcode IDE on a Mac computer. You cannot effectively build and debug on a Windows machine alone. Xcode comes with all the required Apple development toolkit: SDKs, a code editor, compile/build tools, simulators, and a debugger. You will have to use either CocoaPods or Carthage as the package managers to bring in 3rd party SDKs or deliver an SDK that you write.

Apps can be built for iOS devices either by using the native iOS SDK with Objective-C and Swift or with the various cross platform technologies that are written against the SDK of that framework but targeted for iOS.

Developing mobile apps for the Android platform

Android is an open source platform that is primarily developed and promoted by Google. Google promotes its own mobile device brand, Pixel (and the earlier brand, Nexus). However, there are large number of manufacturers like Samsung, Huawei, Xiaomi, and Oppo that sell their own branded phones and tablets that are powered by Android and derivatives of the Android OS like CyanogenMod and MIUI. Android devices come in a large number of form factors, from phones to tablets, because of a variety of manufacturers with multiple models that address different user preferences.

For building Android apps, you need the Android development toolkit that comes with the required SDK, debuggers, and emulators. For an IDE, the Android Studio is by far the most popular, but there are other equally popular IDEs like Netbeans and IntelliJ Idea. You can have the development environment on any OS (Mac, Windows or Linux). Android uses Gradle for the build system, and Android Studio comes with Code templates and other performance assessment and enhancement tools.

Third party libraries are added to the app using a gradle package directive. A large number of 3rd party Android SDKs are distributed by using a package manager called maven, popularly known as mavencentral. The apps built for Android are distributed in the default Google Play store that is managed by Google.

Apps can be built for Android devices either by using the native Android SDK with Java and Kotlin or with the various cross-platform technologies that are written against the SDK of that framework but targeted for Android.

Developing mobile apps for both the iOS and Android platforms

With native app building approaches, you have to maintain one code base per platform. The code repository size is proportional to the platforms that need to be supported. You often need to develop for both the platforms, but you don't need the native language capabilities. Luckily, there are technologies that allow you to write in one language or framework and target the app for both platforms, which means that developers that are not fully familiar with Java and Swift but are experts in other technologies like Web or C# can use their skills to develop apps for Android and iOS.

Various open-source communities or companies have come up with frameworks that have an affinity to their developer base to develop for Android and iOS. These frameworks come with their own ecosystem of tools and IDEs that make it convenient for the developers familiar with those frameworks. For example, C# Xamarin tooling is geared towards using Microsoft Visual Studio and Apache Cordova is geared towards using the web IDEs like VSCode. However, you still need the Android and iOS development environment tools installed (as I explained in the Android and iOS sections earlier) so that the cross-platform tooling can leverage the native SDKs and packaging. Sometimes the frameworks come with their own simulators, but mostly land up using the simulator of the underlying native Android or iOS development environments.

HTML5 and Apache Cordova

Apache Cordova started out as a project called PhoneGap from Adobe. With Apache Cordova you can run HTML and JavaScript (JS) code in a captive browser instance called the Webview. By using a webview, you can write code once and then execute it anywhere. One of the criticisms of the cross-platform approach is that the response can be sluggish owing to the JavaScript code being executed in the webview.

The HTML code is packaged along with the app itself and installed on the mobile device. This opens up the possibility for using a wide variety of web UI frameworks like Jquery, React JS, Bootstrap, Angular JS, or Vue. There are also other layered frameworks around it like Ionic that ultimately run on Cordova. Apart from the UI libraries, there is a concept of plugins that allow the JS code to access native device capabilities like the camera, contact list, or location. There is a large ecosystem of third party plugins available.

The most popular package manager is npm.

The IDEs that are popular among the developer community are Visual Studio Code, and Eclipse. Ionic, which comes as a wrapper on top of Apache Cordova, and currently uses Angular JS, also has its own IDE called Ionic Creator that's available for a purchase. However, you can still develop an Ionic Angular JS app using any other IDE as mentioned earlier.

JavaScript and React Native

React Native was released in 2015 by Facebook. React Native uses JavaScript as a programming language for writing mobile apps. There is no HTML used in writing React Native apps. This code is interpreted at runtime and executed using a "bridge" for accessing the device's native SDK capabilities. React Native apps use the platform native UI library to render the UI components, which makes the UI truly native. React Native has become very popular because the learning curve for JavaScript is very low.

Mobile-optimized websites

By using any of the frameworks like Material, Bootstrap, or Foundation, you can deliver a responsive web site. Responsive web design has layouts that adapt to the available real-estate by giving a native experience. For example a website might be a 5-column layout in a laptop browser, but that website will render in a 2-column layout when viewed from a browser on a mobile device in portrait mode. The two primary differences between a mobile-optimized website and viewing a regular website on a mobile phone is that you should not have to zoom in to view the content and there is no horizontal scrolling.

Progressive web apps (PWAs)

Progressive web apps are actually web sites that render like a native app. The browsers have also evolved to detect that it's a PWA and will remove the surrounding browser to give the web content the full device real estate. When you add a shortcut for this web site, it creates an icon on the mobile home screen, so that you can launch it just like a regular app. The browser also caches specified content locally on the device so that you have a fast and reliable load time and a progressive offline experience as well. This is achieved by using:

- A service worker that works in the background to pre-cache the content
- Responsive design for optimal rendering
- A web app manifest to notify the browser that it's installable

Conclusion

The mobile platforms may not see a drastic change other than Apple and Google enhancing new features to their platforms. But the cross-platform landscape has seen a lot of churn. You can expect every year there will be a new favorite framework. It's important to choose the development approach based on the team skills and app requirements. To be successful in Mobile development, developers must keep up with the changes in the tools and platforms since mobile development is still a evolving technology space with AI and Machine Learning coming to the mobile devices.

Adapted from: https://developer.ibm.com/technologies/mobile/articles/choosing-the-best-programming-language-for-mobile-app-development/