Difference between WEP, hybrid and native app

WEP app

A web app or browser-based app can deliver similar functionality to an app as a website. In fact, with a little creativity, you can keep the differences to a minimum and design a web app so it looks and feels pretty much like a native app. There are different approaches to help you create successful mobile websites, such as responsive and adaptive design.

1. **Progressive Web Apps (PWAs):** These web applications deliver an app-like experience using modern web capabilities. They work on any standards-compliant browser platform. Features like offline access, push notifications, and device hardware integration make them similar to native apps. PWAs don't require app store downloads to simplify maintenance and updates.

2. **Responsive Web Apps (RWAs):** RWAs adapt to different screen sizes. They use responsive design to adjust the layout and content automatically. This ensures a consistent and smooth experience across all devices, from smartphones to desktops. RWAs prioritize fluid design and cross-device compatibility for diverse internet access methods.

Native app

A native app only runs on a specific mobile operating system. It won't run on other mobile operating systems. So, for example, if you develop a native app for iOS, you'd do the development in the Xcode environment using Swift. For Android, developers commonly use Android Studio and Java.

Native apps can normally access all the functionality of the chosen device easily. You can run them without error on the device if developed properly.

However, this comes with a trade-off. If you want your app to run on iOS and Android, you will have to develop the app twice, once for each operating system. This can make the development process both slower and more expensive.

Many companies will develop their app for a single operating system when they choose the native route. If the app is successful in that environment, they will then go back and recreate it for other operating systems.

HYBRIDE APP

A hybrid app works on multiple platforms. You write it with a single standard code language (such as C# or a combination of HTML5 and JavaScript) and then compile and execute it on each platform. The use of plugins for that operating system will manage device-specific interactions.

Hybrid development environments include:

1. Ionic: A popular open-source framework for building hybrid mobile apps using web technologies such as HTML, CSS, and JavaScript.
2. React Native: A framework for building mobile apps using React, a popular JavaScript library for building user interfaces.

3. Xamarin: A framework for building mobile apps using C# and .NET, which allows developers to share code across multiple platforms.

Differences

Feature	Native App	Web App	Hybrid App
Development Language	Platform-specific (e.g., Swift for iOS)	Web tech-Nologies (HTML, CSS, JavaScript)	Combination of native and web technologies
Performance	High with smooth interactions	Dependent on browser and internet speed	Good, but may not match mobile apps
Device Feature Access	Full access to device hardware	Limited access	Moderate access depends on plugins
Offline Access	Generally available	Limited or none	Often available
Compatibility	Specific to each platform (iOS, Android, etc.)	Universal across all devices with a web browser	Broad, but with some platform-specific tweaks
Development Cost	Higher due to platform specificity	Lower	Moderate, less than native
Maintenance	Frequent updates required	Easier to update, centralized	Regular updates needed, but fewer than separate mobile apps for each platform
Internet Dependency	Low	High	Low
User Experience	Optimal, tailored to each platform	Varies, generally good	Good, aims to mimic native experience

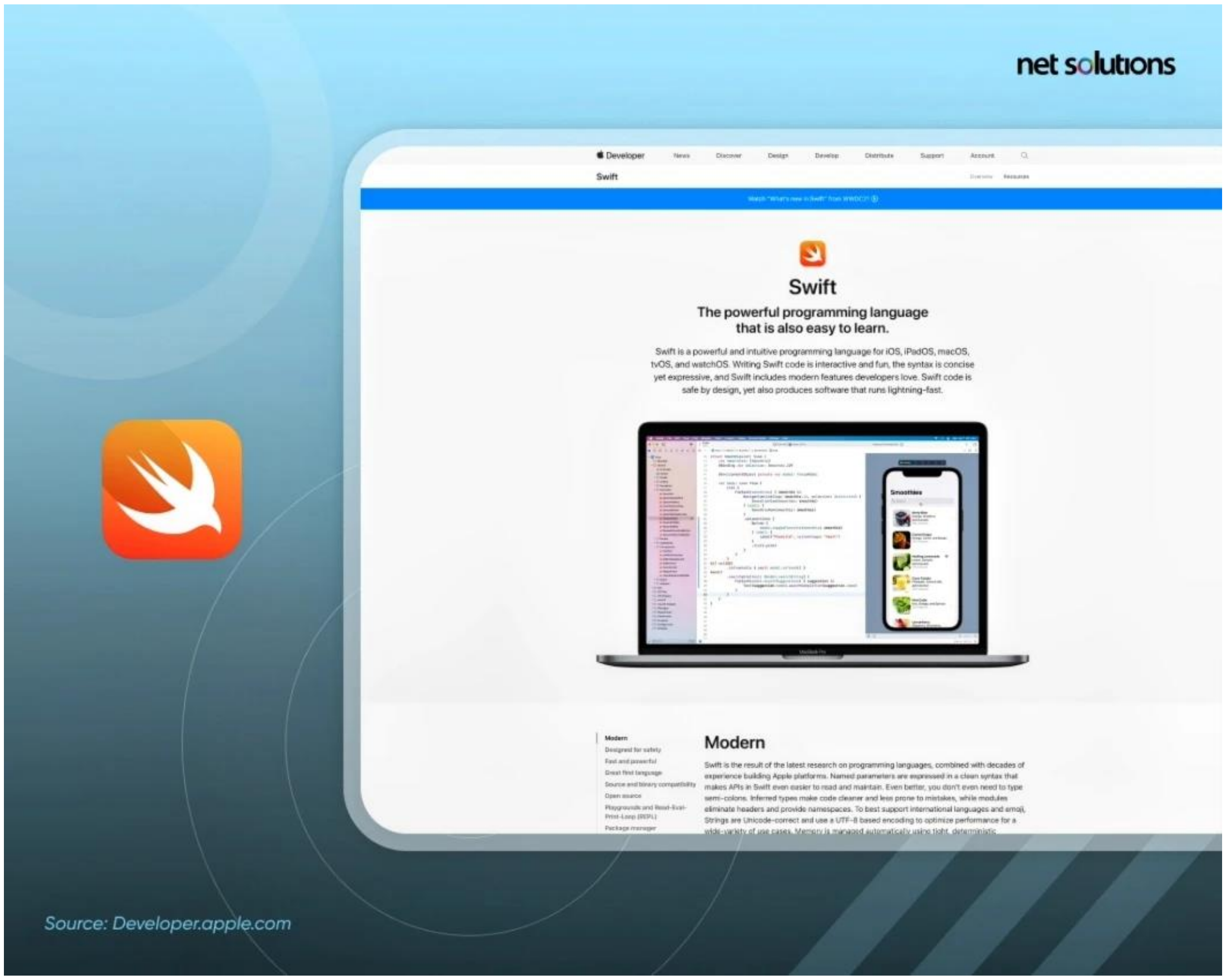
https://www.interaction-design.org/literature/article/native-vs-hybrid-vs-responsive-what-app-flavour-is-best-for-you

Popular Programming Languages for Mobile Apps

1. Swift

As the official language for iOS app development in 2022, Swift is still considered a younger language, overtaking Objective-C as the language of choice by Apple after it was introduced in 2014.

Swift is very user-friendly, ideally suited to new programmers, focusing on expressiveness, safety, and speed. However, Swift is highly dependent on third-party tools and lacks the maturity of other languages. Still, there is no more natural choice for iOS development than Swift.



Swift

The powerful programming language that is also easy to learn.

Swift is a powerful and intuitive programming language for iOS, iPadOS, macOS, tvOS, and watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and Swift includes modern features developers love. Swift code is safe by design, yet also produces software that runs lightning-fast.

Modern

Swift is the result of the latest research on programming languages, combined with decades of experience building Apple platforms. Named parameters are expressed in a clean syntax that makes APIs in Swift even easier to read and maintain. Even better, you don't even need to type semi-colons. Inferred types make code cleaner and less prone to mistakes, while modules eliminate headers and provide namespaces. To best support international languages and emoji, Strings are Unicode-correct and use a UTF-8 based encoding to optimize performance for a wide-variety of use cases. Memory is managed automatically using tight, deterministic

2. . **Kotlin**

The official Android programming language is Kotlin, although it is one of many languages that we can use for Android app development. Kotlin is a simple language with a powerful, clean syntax and combines both acquisitive and functional programming features to allow for faster compilation

3.JavaScript

JavaScript is a top language choice by many developers for mobile apps in coordination with frameworks such as React Native, Cordova, NativeScript, and Appcelerator. React Native was created by Facebook, giving it the backing to make it a dominant player in the development of mobile apps, particularly those focused on native UI elements – without the need to know native programming languages such as Swift or Kotlin.

4. Objective-C

Once the official language for iPhones, Objective-C remains a common iOS programming language due to its stable performance, resource availability, and compatibility with C++.

5. Java

Once the official language for iPhones, Objective-C remains a common iOS programming language due to its stable performance, resource availability, and compatibility with C++.

6. Python

Python has been around since the 80s and is still considered the top programming language, making it an ideal beginner choice due to its versatility in creating mobile apps and a wide variety of other software. It is English-syntax based with many robust libraries. Although not always considered a “mobile app coding language,” given the availability of native language options, Python is often relied upon for more specialized mobile development in areas such as data science.