The Anatomy of a Native App
Defining Native

Whether accessing order history during a sales call or checking a flight status, users expect information to be instantly accessible and presented in a way that makes the most of a device’s capabilities and form factor. One in four apps will be abandoned after first attribute, so app publishers really only have one chance to make a first impression.

Performance, reliability, design, and usability are the main factors that determine user engagement and retention. With respect to these factors, a fully native app has conclusive advantages over web and hybrid apps, yet the term “native” is often misused and applied to these non-native approaches.

Fortunately, there are three definitive characteristics of native apps that make it easy to cut through the marketing hype.
These characteristics are:

**Native User Interface (UI)**
Apps are built with standard, native user interface controls in a manner that fully conforms with each platform’s design conventions. Apps not only look the way the end user expects, they behave that way too.

**High-fidelity API Access**
Apps have access to the full spectrum of functionality exposed by the underlying platform and device, including platform-specific capabilities, including fingerprint authentication, payments, health sensors, Bluetooth, and NFC.

**Native Performance**
Apps leverage platform-specific hardware acceleration, and are compiled for native performance. This can’t be achieved with frameworks that interpret the code at runtime.
Native User Interfaces

Xamarin is the only cross-platform solution that exposes 100% of the UI capabilities of each device platform.

For a framework to claim to support native user UIs, the apps developed with the framework must be displayed using the native controls of the underlying device operating system. For example, when a Xamarin developer creates a “table view” in Xamarin.iOS and Xamarin.Android, they use C# to create iOS UITableViews and Android ListView Views. Xamarin developers have complete access to all properties of the iOS UITableView and Android ListView APIs. When the app is running live, the native iOS and Android APIs are actually rendering the tables. Xamarin apps look native because they are native.
Xamarin’s no compromises approach

Other cross-platform vendors only support a lowest-common-denominator subset of UI APIs across all of their supported platforms, and within those APIs they often only expose a subset of the properties of each API. Developers have two choices. Adapt their apps to what the vendor supports, not what their end users need, which typically leads to poor app adoption and frustrated users; or write code in the platform-specific languages to get the desired look and behavior, which results in fragmented code that is harder to debug and maintain.

Anything less frustrates users and developers

Xamarin is the only cross-platform solution that exposes 100% of the UI capabilities of each device platform. This puts the power of each platform’s user interface paradigms into the hands of the developer, all in C#, with no compromises.
“The results from our new field sales app are phenomenal — our sales people love the app and are able to engage customers and close deals more effectively. Key to the app’s success is the beautiful, fast user experience made possible by Xamarin.”

Kim MacDougall
Senior Capability Development Manager
Kimberly-Clark Professional
**High Fidelity API Access**

*Anything you can do in Objective-C and Java can be done in C# with Xamarin.*

Beyond the UI, developers need access to all platform-specific device functionality to deliver truly native apps. Anything developers can do in Objective-C, Swift, or Java, they can do in C# with Xamarin. Our unique binding technology allows C# to call the same APIs and use the same UI controls as apps built in platform-specific languages. New form factors such as watches and other wearables are becoming ubiquitous, further diversifying the device landscape. Consumers expect their favorite apps to be accessible to them on these new form factors and developers need complete access to all device platform APIs, on the day they are made available by the platform vendors. Anything short of this does not qualify as native.
Anything less puts a mobile strategy at risk

Other cross-platform vendors support only a subset of each platform’s APIs. This puts the vendor, NOT the developer, in control of an organization’s mobile strategy. All other vendors lag in their support of new operating system releases. Many will claim same-day support, but that typically means that apps can be compiled against a new operating system version, not that the new version’s APIs are available. It may take months for the vendor to support the new API, if at all. If they do eventually add support, they may not support all of the properties within the API.

Xamarin’s future-proof approach

Xamarin exposes 100% of each platform’s APIs with a strong track record of same day support. We’ve had Day One support for iOS since iOS 7 with the Xamarin platform, and iOS 9 devices were available in Xamarin Test Cloud the day they the OS was released to the public. With Xamarin, you have all of the power of developing apps in the platform-specific languages right at your fingertips in C#. This means you can tackle any mobility use case today and tomorrow.
"We can move to new operating systems very inexpensively because Xamarin allows us to reuse the code from one platform to another."

Zurab Sajaia
Senior Economist, World Bank Group
Native Performance

Performance matters most with mobile as users are inherently on-the-go and need fast access to contextually relevant information. Fortunately, device hardware is getting faster and more computationally powerful for each platform. Therefore, it is important to be able to access each operating systems’ hardware-acceleration capabilities and not have artificial vendor limitations.

Gartner

“Xamarin’s architectural approach provides completely native UIs with better performance than other cross-platform techniques.”

Gartner Magic Quadrant for Mobile Application Development Platforms

Xamarin
Anything less threatens user adoption

Successfully mobilizing field sales and service, two of the most common use cases in enterprise mobility, depends in large part on the speed with which personnel are able to access customer and product information to complete an engagement with a customer.

Some vendors venture into a grey area regarding native code execution by claiming that “compiling things with Xcode” makes the resulting apps native. However, if the app is running in an interpreter or in a webview, then this “native” claim is indefensible, and the resulting app is less responsive. Slow apps frustrate employees and the customer alike, and threaten the success of the mobility projects.

Xamarin’s advanced native compilation technology

Xamarin apps are compiled as native binaries, not interpreted or executed in a webview. This results in high performance apps under the most demanding scenarios like complex data visualizations and high frame-rate simulations. Xamarin.iOS does ahead-of-time compilation to produce ARM binaries. Xamarin.Android uses just-in-time compilation for sophisticated optimizations. In either case, your app is a native platform binary.
Native in all 3 Ways

Xamarin provides all of the same native power—native UI, high-fidelity API access, and native performance—as Apple, Google and Microsoft without any compromises. Xamarin exceeds the individual capabilities of these platform-specific vendors by enabling the efficiency, and time-to-market advantages of code-sharing and re-use across device platforms. This unique approach puts Xamarin in a class by itself.

Visit xamarin.com/enterprise to download our “Key Approaches for Mobile Success” white paper to learn more about how Xamarin can help you deliver mobile excellence.
The **Native App Anatomy** checklist

<table>
<thead>
<tr>
<th>NATIVE UI</th>
<th>NATIVE API</th>
<th>NATIVE PERFORMANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>🟢 Access to 100% of the UI APIs</td>
<td>🟢 Access to 100% of each platform’s APIs</td>
<td>🟢 Apps are compiled as native binaries, that run just like apps built using Objective-C and Java</td>
</tr>
<tr>
<td>🟢 Access to 100% of the properties of the UI API</td>
<td>🟢 Same-day support for new APIs in OS releases</td>
<td>🟢 Apps take use standard controls and platform-level support for hardware-accelerated rendering</td>
</tr>
<tr>
<td>🟢 Controls rendered using the underlying platform’s API</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☒ Limited to a subset of APIs</td>
<td>☒ Limited to a subset of APIs</td>
<td>☒ Apps are interpreted at runtime or through a webview</td>
</tr>
<tr>
<td>☒ Limited to a subset of properties</td>
<td>☒ the ability to compile against a new OS version, but not use native APIs</td>
<td>☒ Apps use non-standard UI and/or that lack access to hardware acceleration</td>
</tr>
<tr>
<td>☒ Controls simulated via HTML and CSS, or by using images</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>