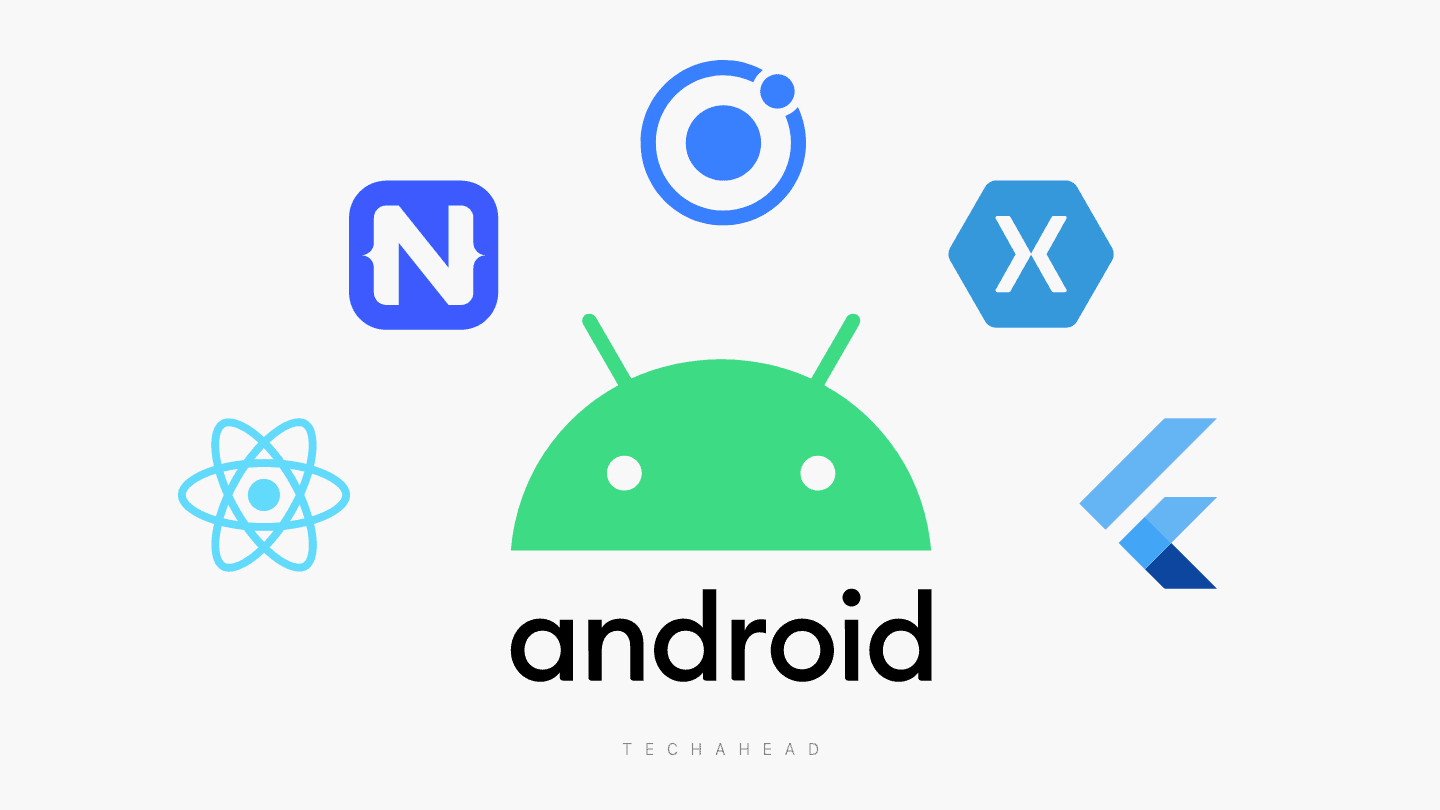
# Android Frameworks Presentation



## 

## Group Members

1. **Lontsi Boualang Marc Arsene**
2. **Achidi Manga Juvet**
3. **Maffo Wamba Nina**
4. **Nganfack Paul**
5. **Fotso Bernard**
6. **Ngenbe Georges**
7. **Aminatou Wassou Jennifer**

Table of Contents

[Android Frameworks Presentation 1](#_Toc130833000)

[Group Members 1](#_Toc130833001)

[Android Frameworks 2](#_Toc130833002)

[What is an Android Application Framework? 2](#_Toc130833003)

[Ionic framework 3](#_Toc130833004)

[Overview of the Ionic framework 3](#_Toc130833005)

[The scope and Limitations of the Ionic framework 3](#_Toc130833006)

[Scope 3](#_Toc130833007)

[Limitations 4](#_Toc130833008)

[Pros and Cons of the Ionic framework 4](#_Toc130833009)

[Pros 4](#_Toc130833010)

[Cons 5](#_Toc130833011)

[Example of companies (apps) that use Ionic 5](#_Toc130833012)

[FLUTTER 6](#_Toc130833013)

[Overview of the Flutter framework 6](#_Toc130833014)

[Technologies and tools used to develop apps in flutter: 6](#_Toc130833015)

[Scope and Limitations of the Flutter framework 6](#_Toc130833016)

[Scope 6](#_Toc130833017)

[Limitations 7](#_Toc130833018)

[Xamarin Framework 8](#_Toc130833019)

[Xamarin.Android 8](#_Toc130833020)

[Xamarin.Forms 8](#_Toc130833021)

[The scope and Limitations of Xamarin 9](#_Toc130833022)

[Scope 9](#_Toc130833023)

[Limitations 10](#_Toc130833024)

[Companies Using Xamarin 11](#_Toc130833025)

[References 12](#_Toc130833026)

# Android Frameworks

**The Android Operating system** is a mobile operating **system** developed by Google based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.

**A framework**, or software framework, is a platform that provides a foundation for developing software applications. Think of it as a template of a working program that can be selectively modified by adding code. It uses shared resources – such as libraries, image files, and reference documents – and puts them together in one package.

In other words, a framework is a Software toolkit that enables app developers to piece together a finished product that meet user requests. A framework provides the bones of an app, to be fleshed out with graphics, animations, and special features additional functionalities.

## ****What is an**** Android ****Application Framework?****

**An Android application framework** is a software toolkit that enables app developers to piece together a finished product that meets the requirements of its proprietor. A framework provides the bones of an application, to be fleshed out with graphics, animation, special features and functionality. Application frameworks are designed to simplify the app development process, and make it easy to manage, modify and fix bugs down the road. It’s necessary to distinguish between SDK (software development kit) and frameworks. SDK contains [IoT software development](https://tateeda.com/iot) tools, including frameworks.

Different Android app development frameworks share many of the same attributes, along with unique features that make them better suited to certain types of projects. **Naming the best framework for Android app development is barely possible**, because of multiple nuances, advantages, and drawbacks specific to each of these frameworks. The application framework you select should be chosen with the specific requirements of your mobile project in mind.

# REACT NATIVE

React Native is an open source JS based mobile framework for writing mobile apps for android and iOS. It is based on react (a JS library for building UIs for web apps). It was created by the Meta company in 2015.

React native apps are written using a mix of JS and XML known as JSX (a syntax that lets you write text like <text> text </text>.

React invokes the native rendering(process of loading ads using native code that already exists on a specific host platform which speeds up loading speed) APIs in Objective-C(for iOS) or Java (for Android), thus the app will render using real mobile UI components, not web views and will look and feel like any other mobile app.

## Advantages of React Native.

React native has the following advantages:

* React Native renders using its host platforms standard rendering API which differs in other frameworks like Ionic or Cordova that produce web views, which makes the produced app look and feels like other mobile apps.
* React Native translates your markup to real, native UI elements, using its means of rendering views on whatever platform you use.
* React Native works separately from the main UI thread, so your app can maintain high performance without losing in capability.
* For developers accustomed to working on the Web with React, this means you can write mobile apps with the performance and look and feel of a native application, while using familiar tools. React Native also represents an improvement over normal mobile development in two other areas: the developer experience and cross-platform development potential.

In addition to the above advantages, React Native has the following advantages for developer’s experience:

* React Native has strong developer tools and meaningful error messages.
* Because React Native is "just" JS, you don’t need to rebuild the app to see changes being reflected. You can simply refresh your app, just like any web page. All those minutes waiting for an app to build can really add up, so React Native's quick iteration cycle feels like a godsend.
* React Native also lets you take advantage of debugging tools and error reporting, like your browser's inbuilt debuggers.

Under cross-platform, we have:

Any dev that knows how to write react code can target any mobile platform with the same skillset, this enables many developers to work on projects and share resources together more effectively. Of course this comes with some limitations as you still require some minute specificities depending on the platform you use.

## Disadvantages

* One of its disadvantages is that some features of both platforms are still not supported.
* Because RN introduces another layer to your project, it can also make debugging hairier, especially due to the intersection of RN and the host platform.
* Zero support for multiprocessing, as it has no parallel threads feature.
* React Native is not able to use all the potential of a specific platform

## Companies that use React Native

* Uber Eats
* Discord
* Instagram
* Pinterest
* Facebook

Ionic framework

## Overview of the Ionic framework

Ionic framework is an open-source UI toolkit for building performant, high-quality mobile apps, desktop apps, and progressive web apps using web technologies such as **HTML, CSS, and JavaScript**. It allows developers to build once and run everywhere. It was created by Max Lynch, Ben Sperry, and Adam Bradley of Drifty Co. in 2013. The first beta version of the Ionic framework was released in March 2014.

The Ionic framework mainly focuses on front-end user experience or UI interaction, which handles all the look and feel of your app. It is easy to learn and can integrate with other libraries or frameworks such as **Angular, Cordova and React**. It can also be used as a standalone without a front-end framework using a simple script.

Officially, the Ionic framework has integration with Angular, but it also provides support for Vue.js and React.js, which are in development.

Technology use to develop the framework and used in the framework to develop other apps

## The scope and Limitations of the Ionic framework

### Scope

It provides features such as components, tools, and services that make it easy to create, test, and deploy hybrid mobile applications. Ionic also includes mobile-optimized components, gestures, and other features that make mobile development faster and easier.

Ionic is very versatile and the following points cover the aspects where Ionic are best in:

1. Ionic is a front-end UI framework for developing hybrid mobile apps with HTML5, CSS3, and JavaScript. It also provides tools and services for building iOS and Android apps using web technologies.
2. It focuses on the look and feel of your app, allowing you to create a beautiful, interactive user experience. Ionic also features a strong community and robust documentation.
3. It includes components such as navigation, data display, forms controls, and other elements that are commonly used in mobile apps.

### Limitations

1. Debugging of an application built on Ionic can be challenging and requires more time because of unclear error messages.
2. It is not suitable for large and complex projects as it has some limitations with respect to performance and scalability.
3. Ionic does not provide full access to hardware components such as cameras, microphones, etc. so developers must use third-party plugins in order to access those features.
4. Builds can randomly crash without any reason

## Pros and Cons of the Ionic framework

### Pros

1. **One codebase, running everywhere**[**​**](https://ionicframework.com/docs#one-codebase-running-everywhere)

Ionic is the only mobile app stack that enables web developers to build apps for all major app stores and the mobile web from a single codebase. And with [Adaptive Styling](https://ionicframework.com/docs/theming/platform-styles), Ionic apps look and feel at home on every device.

1. **A focus on performance**[**​**](https://ionicframework.com/docs#a-focus-on-performance)

Ionic is built to perform and behave great on the latest mobile devices with best practices like efficient hardware accelerated transitions, and touch-optimized gestures.

1. **Clean, simple, and functional design**[**​**](https://ionicframework.com/docs#clean-simple-and-functional-design)

Ionic is designed to work and display beautifully on all current mobile devices and platforms. With ready-made components, typography, and a gorgeous (yet extensible) base theme that adapts to each platform, you'll be building in style.

1. **Native and Web optimized**[**​**](https://ionicframework.com/docs#native-and-web-optimized)

Ionic emulates native app UI guidelines and uses native SDKs, bringing the UI standards and device features of native apps together with the full power and flexibility of the open web. Ionic uses Capacitor (or Cordova) to deploy natively, or runs in the browser as a Progressive Web App.

### Cons

#### Performance

The performance of the Ionic application is not as good as compared to native mobile applications. However, the performance gap is not noticeable for most of the average users.

#### Security

The app developed with the Ionic framework may not provide as much security as a native application. For example, if you are developing a financial app, e.g the app for a bank, the Ionic framework is not recommended.

#### Design limitations

Ionic has certain design limitations which can make it difficult to build complex user interfaces.

#### Need for specialists

JavaScript can be tricky to learn. So, there is a need for someone who can not only "do" JS, but also have deep knowledge of advanced libraries and technologies including Angular, Cordova, Ionic, etc.

#### Limited native functionality

There are some native functions that may not be available in the Ionic framework. In such a case, you need to develop the plugin to fulfill that function yourself. However, there are many plugins available to cover most of the native functionalities.

## Example of companies (apps) that use Ionic

* Burger King
* Duck Duck Go
* Adobe Creative Cloud
* H&R

# FLUTTER

## Overview of the Flutter framework

Flutter is an open-source software development kit (SDK) and free technology developed by Google in May 2017 for creating native Android and IOS apps with a single codebase. Flutter is based on Dart programming language and React Programming Architecture that supports Android Studio, Firebase APIs, inline videos & ads, and offers the development of high-performance applications.

The different technologies use to built flutter are C, C++, Dart and Skia.

## Technologies and tools used to develop apps in flutter:

* Android Studio & IntelliJ
* Visual Studio Code
* DevTools
* Flutter SDK
* Flutter and the pubspec file
* Hot reload
* Flutter Fix
* Code formatting. Terms.

## Scope and Limitations of the Flutter framework

### Scope

1. **Free and open source**

It is an open-source project which makes it available for use and study by start-ups for any given purpose. It is easily accessible and user friendly. It does not only gives you a diverse scope for design but also no other company provides you with as many options.

1. **Amazing Widget catalog**

The best part about flutter is that you have a wide range of widgets beautifully cataloged for you. Not only does it make it hassle-free but also helps you make a functionally amazing app. You use Dart to write your flutter app which is compiled to native code. The IntelliJ plugin makes for good integration.

1. **It’s Inherent Graphic Library**

Flutter uses the Skia-built-in library for rendering. This makes it more platform-independent; The SDK provides a rich set of widgets, in particular the material and Cupertino collections for rendering native-like widgets for android and iOS.

1. **Hot Reload**

One of the drawbacks of compiled languages before scripting languages is the loss of time for building a project. With frequent edits, it can take up a substantial part of the working time. Flutter Hot Reload allows you to display the effect of your edits in the code immediately.

1. **Compatibility**

Since widgets are part of the app and not the platform, you’ll likely experience less or no compatibility issues on different OS versions. This in turn means less time spent on testing.

1. **Development Expectations for Fuchsia OS**

Flutter is currently the only tool for creating applications for the Fuchsia OS from Google. For several years now, the new OS has aroused the interest of developers and questions about its purpose. Nevertheless, with the further development experience with flutter will allow you to quickly master a new niche in the software development market.

### Limitations

1. **Lack of Third-party Libraries and Widgets**

Flutter is not too old unlike its contemporaries and lacks the presence of third-party libraries. Although, it gives an amazing UI package, yet the requirement of third-party libraries for extensive development is still awaited.

1. **Code Pushing**

Code pushing allows developers to instantly push patches to their app without going through the usual app store release process. Bugs can be fixed without a new release, allowing a more web-like continuous development process. It is supported by React native, Cordova and Ionic. Flutter does not support this.

1. **TVs Watches & Cars**

You can’t use flutter to build apps for tvOS, watchOS, CarPLay or Android Auto. There’s some limited support for Wear OS. Flutter has to add Bitcode support to deploy to tvOS and watchOS.

# Xamarin Framework

Xamarin is an open-source platform for building modern and performant applications for iOS, Android, and Windows with **.NET**. Xamarin is an abstraction layer that manages communication of shared code with underlying platform code. Xamarin runs in a managed environment that provides conveniences such as memory allocation and garbage collection.

Xamarin enables developers to share an average of 90% of their application across platforms. This pattern allows developers to write all of their business logic in a single language (or reuse existing application code) but achieve native performance, look, and feel on each platform.

Xamarin applications can be written on PC or Mac and compile into native application packages, such as an **.apk** file on Android, or an **.ipa** file on iOS.

## Xamarin.Android

Xamarin.Android applications compile from C# into **Intermediate Language (IL)** which is then **Just-in-Time (JIT)** compiled to a native assembly when the application launches. Xamarin.Android applications run within the Mono execution environment, side by side with the Android Runtime (ART) virtual machine. Xamarin provides .NET bindings to the Android.\* and Java.\* namespaces. The Mono execution environment calls into these namespaces via **Managed Callable Wrappers (MCW)** and provides **Android Callable Wrappers (ACW)** to the ART, allowing both environments to invoke code in each other.

For more information, see [Xamarin.Android architecture](https://learn.microsoft.com/en-us/xamarin/android/internals/architecture).

## Xamarin.Forms

Xamarin.Forms is an open-source UI framework. Xamarin.Forms allows developers to build Xamarin. IOS, Xamarin.Android, and Windows applications from a single shared codebase. Xamarin.Forms allows developers to create user interfaces in XAML with code-behind in C#. These user interfaces are rendered as performant native controls on each platform. Some examples of features provided by Xamarin.Forms include:

* XAML user-interface language
* Databinding
* Gestures
* Effects
* Styling

For more information, see [Xamarin.Forms](https://learn.microsoft.com/en-us/xamarin/xamarin-forms/).

## The scope and Limitations of Xamarin

### Scope

Xamarin combines the abilities of native platforms, while adding features that include:

1. **Complete binding for the underlying SDKs** – Xamarin contains bindings for nearly the entire underlying platform SDKs in both iOS and Android. Additionally, these bindings are strongly-typed, which means that they’re easy to navigate and use, and provide robust compile-time type checking and during development. Strongly-typed bindings lead to fewer runtime errors and higher-quality applications.
2. **Objective-C, Java, C, and C++ Interop** – Xamarin provides facilities for directly invoking Objective-C, Java, C, and C++ libraries, giving you the power to use a wide array of third party code. This functionality lets you use existing iOS and Android libraries written in Objective-C, Java, or C/C++. Additionally, Xamarin offers binding projects that allow you to bind native Objective-C and Java libraries using a declarative syntax.
3. **Modern language constructs** – Xamarin applications are written in C#, a modern language that includes significant improvements over Objective-C and Java such as dynamic language features, functional constructs such as lambdas, LINQ, parallel programming, generics, and more.
4. **Robust Base Class Library (BCL)** – Xamarin applications use the .NET BCL, a large collection of classes that have comprehensive and streamlined features such as powerful XML, Database, Serialization, IO, String, and Networking support, and more. Existing C# code can be compiled for use in an app, which provides access to thousands of libraries that add functionality beyond the BCL.
5. **Modern Integrated Development Environment (IDE)** – Xamarin uses Visual Studio, a modern IDE that includes features such as code auto completion, a sophisticated project and solution management system, a comprehensive project template library, integrated source control, and more.
6. **Mobile cross-platform support** – Xamarin offers sophisticated cross-platform support for the three major platforms of iOS, Android, and Windows. Applications can be written to share up to 90% of their code, and Xamarin. Essentials offers a unified API to access common resources across all three platforms. Shared code can significantly reduce both development costs and time to market for mobile developers.
7. **Safe and Secure**

Apps built on Xamarin are safe and secure as various elements are hosted on the cloud (Azure). The Azure comes with highly secure HTTP Certificates so, there is a secure connection between apps and cloud data.

### Limitations

* **Limited Access To Open Source Libraries.**

Xamarin requires the use of elements provided by the platform, as well as .NET open source libraries. As you would expect, the choice is not as rich as it is for iOS and Android native development, and you might end up with a lot of native coding.

Also, Xamarin does not support all available third-party libraries for Android and iOS without specific wrappers and developers often have to spend some additional time developing the element they need.

* **Not Suitable For Apps With Heavy Graphics.**

If an app has/requires a rich UX/UI, you might need to consider using native app development tools. Apps that heavily rely on appearance or use a lot of user interactions will take time to develop using Xamarin and require extensive knowledge of iOS and Android native technologies.

* **Larger App Size.**

Mobile apps developed in Xamarin can vary in size and end up bigger compared to native-built ones. The size depends on the data libraries, Mono runtime, and Base Class Libraries used for Xamarin apps.

They are a few ways to mitigate large app sizes. Many developers use additional optimization like linking as the best way to remove code and keep the file size in control. Another way to decrease the file size is to remove nonessential code.

* **Slightly Delayed Support for the Latest Platform Updates.**

It’s impossible for third-party tools to provide immediate support for the latest iOS and Android releases because it takes time to implement changes. Using Xamarin means waiting on the support team to catch up to new releases.

* **Compatibility Issues with Third-Party Libraries and Tools.**

If you choose to use third-party resources with your app, there’s a chance that you’ll need a specific plugin or integration that’s not provided by the Xamarin platform. Most libraries are packed with solutions for native technologies, but not for Xamarin. If there’s a need for third-party integrations, using wrappers might be the best way to handle it, or you will have to create bindings before you get to the next stage in app development.

* **Expensive Solution for Enterprise Users.**

As mentioned, Xamarin is a free, open-source platform for individual developers. When it comes to enterprises, it can be a costly solution. Usually, the first year comes with the highest price tag as a license for Visual Studio’s core set of tools is $1,199 and Visual Studio Enterprise $5,999. If you decide to go for renewal, Visual Studio will cost you $799, while Visual Studio Enterprise offers a 50% discount and gives you access to all of their tools for $2,569.

* **Xamarin Talent Pool and Community.**

Compared to native Android and iOS talent pools, Xamarin still has a relatively smaller community of developers. What this implies is that it can be somewhat hard to find an experienced Xamarin developer. In a Stack Overflow survey from 2020, only 5.8% of developers used Xamarin.

To improve this situation, Microsoft provides extensive support to all developers through Microsoft Learn and free courses. More experienced C# and .NET engineers find the Xamarin platform easy to use.

* **Basic Knowledge of Native Languages.**

As Xamarin. Native (Xamarin.iOS and Xamarin.Android) requires a platform-specific layer of code using the native APIs, knowing the basics of [**Java**](https://softjourn.com/java-development-services) or Kotlin for Android and Objective-C or Swift for iOS is necessary. Xamarin.Essentials library provides multiple cross-platform APIs and can be used to access native features.

## Companies Using Xamarin

* Academy of Motion Picture Arts and Science
* American Cancer Society
* Azure
* BBC Good Food
* JustGiving
* Microsoft News
* UPS

# References

1. <https://ionicframework.com/docs>
2. <https://www.javatpoint.com/what-is-ionic-framework>
3. <https://learn.microsoft.com/en-us/xamarin>
4. <https://ukad-group.com/blog/6-pros-and-3-cons-of-ionic-development/>
5. [www.oreilly.com](http://www.oreilly.com)
6. <https://docs.flutter.dev/>