# Universidad Tecnológica Nacional

## FRRO

TP N° 1 - Xamarin

AÑO: 2020 - 3ELEC3

Tecnologías de Desarrollo de Software IDE

Alumnos Grupo 20:

Elias, Juan Ignacio - 45.860 - juanignacioelias1@gmail.com

Malgarín, Tomás - 46.198 - malgarojr.tm@gmail.com

Miño, Julián - 46.206 - julian.m.mino@gmail.com

Docentes:

Joaquín, Andrés

Sangrá, Andrés

Porta, Ezequiel

Fecha Presentación: 14/09/2020

# Introducción

Para entender qué es Xamarin, primero hay que contextualizar. ¿A qué nos referimos cuando hablamos de aplicaciones nativas? ¿y cuando hablamos de aplicaciones multiplataforma?

**Aplicaciones Nativas**: son aplicaciones cuyo desarrollo implica el uso específico del lenguaje de programación nativo del dispositivo: Objective C o Swift para iOS, Java o Kotlin para Android y .Net para Windows Phone. Es un modelo cien por ciento dependiente de la plataforma y las Apps no son portables, hay que desarrollar una por plataforma.
Los principales paradigmas asociados a las Apps nativas son:

1. Se puede lograr el mejor rendimiento posible.
2. Se puede lograr un look&feel óptimo acorde al sistema operativo
3. Se puede acceder a todas las capacidades del dispositivo

**Aplicaciones multiplataforma**: Son aplicaciones que se desarrollan en un lenguaje de programación general, en donde el desarrollo se realiza usando técnicas y lenguajes específicos de la herramienta y que luego se puede “exportar” a cualquier plataforma o dispositivo con unos cambios mínimos para ser compilada con las herramientas nativas. Es una manera de abaratar costes de desarrollo y mantenimiento.

Conociendo todo esto, podríamos preguntarnos lo siguiente: ¿Qué es Xamarin? ¿Para qué sirve? ¿Cómo funciona? Estas y otras cuestiones son las que este trabajo abarca a continuación.

## ¿Qué es Xamarin?

Xamarin es un kit de herramientas de desarrollo de aplicaciones multiplataforma, lanzado por primera vez en 2011, que nos permite elaborar aplicaciones de Android, iOS y Windows. Xamarin trabaja a través del framework Mono para comunicarse con la Interfaz de Programación de Aplicaciones (API) de funciones comunes de dispositivos móviles, aprovechando un código compartido para forjar compatibilidad y usabilidad en múltiples plataformas o sistemas operativos.

Xamarin representa una transición fácil al mundo del desarrollo de aplicaciones multiplataforma para los desarrolladores de C#, puesto que el lenguaje base de Xamarin es precisamente C#, y su entorno de trabajo es .NET.

A su vez, Xamarin permite a los desarrolladores reutilizar un promedio del 90% del código fuente de la aplicación, a través de un patrón de diseño donde la lógica de negocio y de acceso a datos es compartida por todas las plataformas, mientras que la parte visual difiere en la medida en que cambien los patrones de diseño de los distintos sistemas operativos.

Las aplicaciones Xamarin pueden ser desarrolladas en una PC o en una Mac indistintamente, y es en el proceso de compilación en el que se generan los paquetes nativos de la aplicación (archivo .apk en el caso de Android, o .ipa en el caso de iOS).

Estas son algunas de las funcionalidades que añade Xamarin a la plataforma:

- Un framework base para acceder a las funciones nativas.
- Usa XAML, un lenguaje extensible, para construir aplicaciones dinámicas para móviles usando C#.
- Librerías para patrones comunes, como Model View ViewModel (MVVM).
- Librerías específicas que incluyen acceso a API de Google, Apple, Facebook, etc para agregar más capacidades.
- Un editor que proporciona el resaltado de sintaxis, finalización de código, diseñadores y otras funciones específicamente para desarrollar aplicaciones móviles.

- Emuladores de varios dispositivos móviles para testear las aplicaciones y poder visualizar la UI en distintos tamaños de pantallas.

## Cómo funciona Xamarin

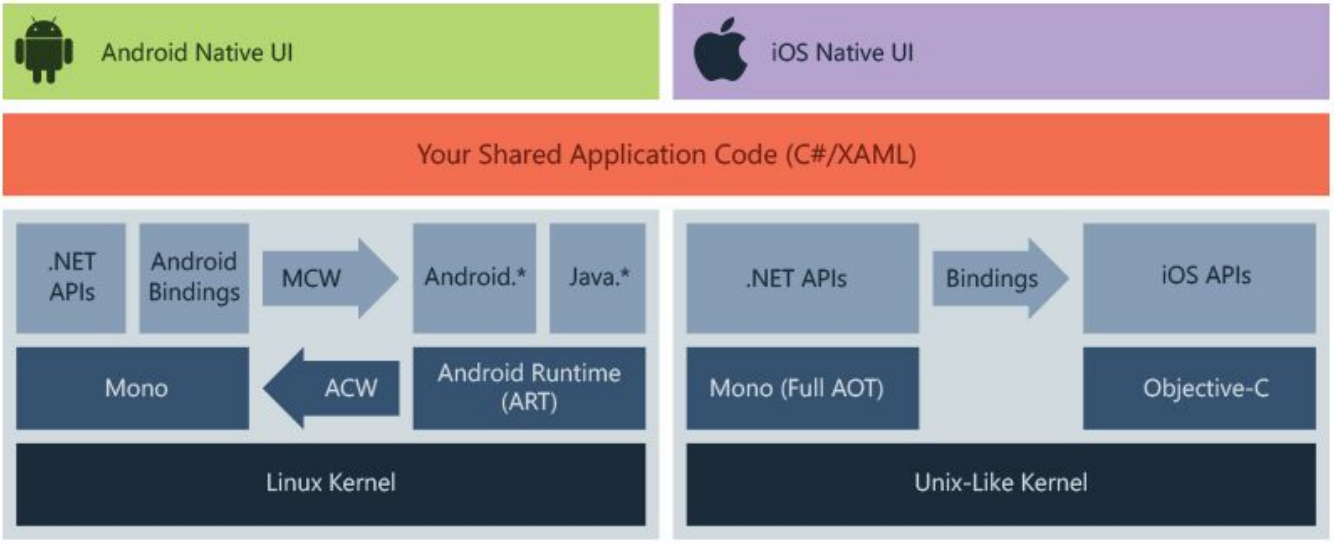


Imagen 1. Diagrama de la arquitectura general de una aplicación multiplataforma desarrollada en Xamarin.

El diagrama muestra la arquitectura general de una aplicación cross-platform hecha con Xamarin. Este framework permite crear UI en cada plataforma y programar la Lógica de Negocios (Business Logic), el acceso a los datos y la comunicación entre redes en C# que serán compartidas a través de las distintas plataformas. Xamarin está ensamblado por encima de .NET, lo que permite que automáticamente se manejen tareas como administración de memoria, garbage collection y operabilidad entre las distintas plataformas.

Más adelante se explicará en detalle el funcionamiento particular de cada una de las formas de utilizar Xamarin a través de Xamarin.Forms, Xamarin.Android y Xamarin.iOS.

## Funcionalidades agregadas

Xamarin combina las capacidades de las plataformas nativas y agrega además otras funcionalidades:

1. Binding (ligaduras/referencias) completo para los SDK (Kits de desarrollo de Software) subyacentes: Xamarin contiene bindings para casi todos los SDK de la plataforma subyacente tanto en iOS como en Android. Además, estos bindings están fuertemente tipados, lo que significa que son fáciles de navegar y usar, y proveen una sólida verificación de tipos en tiempo de compilación y durante el desarrollo. Los bindings fuertemente tipados conducen a menos errores de tiempo de ejecución y aplicaciones de mayor calidad.

2. Objective-C, Java, C y C++ Interop: Xamarin provee utilidades para invocar directamente librerías de Objective-C, Java, C y C++; otorgándole al programador el poder de utilizar todo el código de terceros que le sea necesario. Esto permite utilizar librerías ya existentes para iOS y Android escritas en Objective-C, Java o C/C++. Además, Xamarin ofrece referenciar proyectos que permitan realizar binding a librerías nativas de Objective-C y Java.

3. Las aplicaciones de Xamarin están programadas en C# lo que permite arrastrar las funcionalidades de dicho lenguaje al desarrollo de las aplicaciones móviles (lambdas, LINQ, programación en paralelo, clases genéricas, etc.)

4. Robusta Librería de Clase Base (BCL): las aplicaciones de Xamarin se desarrollan en .NET, una gran colección de clases que tienen funciones integradas y optimizadas como XML, base de datos, serialización, E/S, String, soporte de redes, y más. Todo código C# existente también puede ser compilado para el uso en la aplicación, la cual provee acceso a miles de bibliotecas que añaden funcionalidades más allá del BCL.

5. Moderno IDE: Xamarin usa Visual Studio, un IDE actual que incluye funciones que permiten completar automáticamente el código, un sofisticado

sistema de gestión de proyectos y soluciones, una biblioteca completa de plantillas de proyectos, control de fuente integrado y más.

## Xamarin.Forms

Xamarin.Forms es una herramienta que se usa para la creacion de paginas, diseños y controles que ayudan en el desarrollo de las apps desde un mismo API. Tiene mucho potencial ya que permite customizar los controles y su comportamientos o también crear tus propios controles y diseños. Xamarin.Forms permite a los desarrolladores crear interfaces en XAML con código subyacente (code-behind) en C#. Estas interfaces se crean como controles nativos para cada plataforma. flexible para cualquier tipo de proyecto que se defina.

Algunos ejemplos de las funcionalidades que provee Xamarin Forms son:

- XAML - lenguaje de interfaz de usuario
- Databinding
- Gestos
- Efectos
- Estilo de interfaz

## Xamarin.Android

Las aplicaciones hechas con Xamarin.Android compilan código C# a un lenguaje intermedio (IL) que luego es compilado Just-in-Time (JIT) en un ensamblador nativo cuando se inicia la aplicación. Éstas aplicaciones se ejecutan dentro del entorno de ejecución Mono, junto con la máquina virtual Android Runtime (ART). Xamarin provee de bindings a los namespaces de Android y Java. El entorno de ejecución Mono llama a estos namespaces a través de los Managed Callable Wrappers (MCW) y proporciona los Android Callable Wrappers (ACW) al ART, lo que permite que ambos entornos invoquen código entre sí.

## Xamarin.iOS

Las aplicaciones Xamarin.iOS están compiladas con anticipación o Ahead-of-Time (AOT) de código C# a código assembly ARM nativo. Xamarin usa selectores para exponer Objective-C a C# administrado y registradores para exponer código C# administrado a Objective-C. Los selectores y registradores se denominan colectivamente "bindings" y permiten que Objective-C y C# se comuniquen.

## Diferencia entre Xamarin Nativo/Clásico y Xamarin.Forms

En “Xamarin clásico” compartimos toda la lógica de la aplicación entre las diferentes plataformas, a excepción de la interfaz de usuario, la cual será independiente para cada una de las mismas. Es decir:

- Para el desarrollo en Android, la parte gráfica se implementaría con los bindings de los componentes nativos de Android.
- Para el desarrollo en iOS, la parte gráfica se implementaría con los bindings nativos de iOS
- Para el desarrollo en Windows, la parte gráfica se desarrollaría con las APIs propias de Microsoft.

En cambio, con Xamarin Forms podemos compartir, además de la lógica de la aplicación, la interfaz de usuario, aumentando así la reutilización de código. Xamarin Forms, es por tanto, un conjunto de herramientas destinadas a agilizar el desarrollo multiplataforma y a maximizar la cantidad de código compartido entre cada plataforma: conseguimos que el código compartido entre las plataformas llegue hasta un 90%, lo que permite un ahorro considerable en tiempos de desarrollo y costos de producción.

## Cuándo utilizar Xamarin Clásico y cuándo Xamarin.Forms

Utilizaremos Xamarin Clásico para:

- Apps que requieren de un comportamiento nativo (intrínseco de cada sistema operativo)
- Apps que utilizan muchas APIs específicas
- Apps que requieren de una interfaz personalizada

Xamarin Forms es más adecuado para:

- Apps que no requieren de un diseño muy complejo.
- Apps de código compartido, en las que no importa demasiado la personalización de la interfaz.
- Apps para entrada de datos.
- Prototipos

Por tanto, Xamarin Clásico es idóneo cuando se requiere un nivel muy elevado de personalización de la interfaz de usuario para cada una de las plataformas, siendo más importante el nivel de personalización de la app que la cantidad de código compartido.

En cambio, Xamarin Forms es la mejor opción cuando las aplicaciones requieren menos personalización en la interfaz, y donde la funcionalidad prima sobre la UI. Por ejemplo, para desarrollar apps empresariales en las que prima la funcionalidad y no es necesario enamorar al consumidor con animaciones o diseños muy complejos, Xamarin Forms podría ser tu opción.

## Algunas competencias de Xamarin

**Flutter:** es una tecnología desarrollada por Google que se basa en Dart y un motor portátil de C++ para implementar un potente framework de interfaz de usuario.

### Características principales

- Rendimiento de la aplicación
- Recarga en caliente
- Kit completo de widgets únicos
- Gráficos 2D
- Sencillez de diseño
- Gran cantidad de paquetes disponibles

**React Native:** React Native es un framework de desarrollo móvil de código abierto para el desarrollo móvil multiplataforma, que te permite desarrollar aplicaciones Android e iOS con JavaScript y un amplio enjambre de APIs de componentes nativos.

## Características principales

- Una gama amplia de componentes nativos:
- Desarrollo simplificado con programación declarativa
- Arquitectura modular
- Soluciones y bibliotecas listas para usar
- Soporte de terceros
- Permite la recarga en caliente
- Existe un gran ecosistema y comunidad