Datos personales

Link: <https://drive.google.com/open?id=11OAU1fXyVqaGJivuRU-AzprvB_Q_79Kl>

Artículo

# Resumen

El mundo de la tecnología avanza a pasos agigantados y los proyectos desarrollados por las grandes industrias cada vez son más ambiciosos, llevando consigo un largo proceso de construcción. En el área de desarrollo de *software*, se busca reducir la complejidad y el tiempo que toma el hecho de producir código limpio y funcional. En este aspecto, el desarrollo móvil no es la excepción, ya que en la actualidad se requiere constantemente contar con aplicaciones móviles de gran rendimiento, interfaces amigables a los usuarios y, además, que sean multiplataforma. Estas características encapsulan la esencia de lo que es Flutter, un framework de desarrollo sencillo y completo que fue creado por Google para reducir la complejidad y la cantidad de tiempo invertido.

# Abstract

The world of technology is advancing by leaps and bounds and the projects developed by the large industries, are increasingly ambitious, carrying with it a long process of construction. In software development, the aim is to reduce the complexity and the time it takes to write clean and functional code. In this aspect, mobile development is not the exception, since at present it is constantly required to build apps with high performance, friendly graphic interfaces and that they work in any platform. These characteristics encapsulate the essence of what is Flutter, a simple and complete development framework that was created by Google with the aim to reduce the complexity and time invested.

# Introducción

Con el paso de los días y el crecimiento exponencial de la tecnología, surge la necesidad de que las aplicaciones móviles sean de alto rendimiento, amigables a los usuarios y, sobre todo, que puedan ser ejecutadas desde la mayor cantidad de dispositivos posible. Lo cual, lleva a los programadores a cuestionarse sobre qué herramientas utilizar para escribir aplicaciones nuevas o migrar las ya existentes pero que se han quedado obsoletas.

Contenido

[Resumen 1](#_Toc16566848)

[Abstract 1](#_Toc16566849)

[Introducción: 1](#_Toc16566850)

[Flutter: El presente de las aplicaciones móviles 3](#_Toc16566851)

[Conclusiones 6](#_Toc16566852)

[Referencias 7](#_Toc16566853)

# Flutter: El presente de las aplicaciones móviles

Coloca aquí tu foto en tamaño original y en alta resolución.

**Nombre:** Fabio César De Paz Vásquez.

**Correo electrónico:** fabiodepazv@gmail.com

**Fecha:** 29 de marzo de 2019

**Asesor (a):** Ing. Miguel Ángel Cancinos Rendón

**Palabras Clave:** Desarrollo de software, Aplicaciones móviles, Aplicaciones híbridas, Dart, Flutter, Google, Apps, Smartphones.

En el contexto de aplicaciones móviles, tomar una decisión sobre qué grupo de herramientas utilizar para crear aplicaciones nuevas, es una decisión bastante compleja, ya que, al existir distintos sistemas operativos, se requiere que las aplicaciones puedan ejecutarse de manera correcta en cualquiera de estas. Como consecuencia a esto, algunas personas tienden a inclinarse a implementar sus nuevos proyectos utilizando lo que se conoce como aplicaciones móviles híbridas. Para poder explicar ya adentrarse en esta pequeña parte del universo del desarrollo, es necesario presentar contexto respecto a esto.



**Descripción:** Imagen 1. Dos Smartphones: un Samsung Galaxy J5 (izquierda) y un iPhone 6S (derecha)

**Autor:** Wikipedia, User: Senado Federal

**Dirección electrónica de la imagen:**

<https://upload.wikimedia.org/wikipedia/commons/0/0e/Fotos_produzidas_pelo_Senado_%2830554309793%29.jpg>

Para el desarrollo móvil, se cuenta con tres distintas alternativas a elegir, como se describen a continuación:

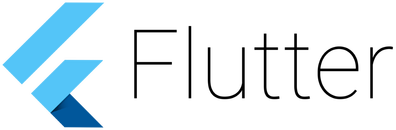
* **Desarrollo nativo:** A manera de simplificar su descripción, puede describirse como aquellas aplicaciones que son desarrolladas utilizando específicamente el lenguaje de la plataforma donde se ejecutan, por ejemplo: Java para Android o Swift para iOS. Esto podría complicar la situación de poder publicar la misma aplicación en distintas plataformas, ya que estas son excluyentes con los otros lenguajes nativos, lo que trae consigo el hecho de tener que realizar doble trabajo si se quisiera desarrollar para varias plataformas de manera nativa. Entonces, ¿por qué razones desarrollar de manera nativa?

1. Permiten mejorar el performance de la aplicación.
2. El look & feel que se logra generar es óptimo y su fluidez es evidente.
3. Es posible acceder a todos los componentes del dispositivo, como los sensores y actuadores.

Estas razones parecen ser convincentes para que los desarrolladores opten por esta alternativa, pero si se quiere realizar implementaciones de manera masiva e invirtiendo menos recursos, esta termina siendo una opción compleja y costosa, debido a esto han surgido las siguientes formas de implementación.

* **Desarrollo híbrido:** Se trata de aplicaciones que se desarrollan implementando estándares de aplicaciones *web*, mediante HTML, CSS y JavaScript, que finalmente es renderizado en el teléfono mediante el uso de un framework específico (como lo puede ser PhoneGap, Ionic o Cordova). Esta parece ser una opción bastante viable, ya que realizando un único desarrollo es posible visualizar y ejecutar las aplicaciones tanto en ambientes *web*, como Android y iOS. Recientemente, este método es utilizado por muchas empresas y programadores a nivel mundial, ya que se ahorran el tiempo de desarrollo y reducen la complejidad del código, simplificándolo a una misma sintaxis y compilación. Estos frameworks ofrecen a los desarrolladores el acceso a funcionalidades nativas a través de ellos, pero como muchas cosas en el mundo, existen sus limitantes. Esto no quiere decir que sea de mala calidad, sino que de cierta forma no tienen acceso a todos los componentes del teléfono y además la fluidez visual no será la misma, debido a la existencia de una capa intermedia que se encarga de renderizar las vistas.
* **Desarrollo generado o *bridged*:** Se trata del desarrollo a través de distintos frameworks que ofrecen su sintaxis y características para escribir el código, pero finalmente estas se encargan de traducir estas instrucciones hacia instrucciones del lenguaje nativo de la plataforma sobre la cual se ejecutan.

Teniendo esto en mente, es posible adentrarse un poco en lo que es Flutter [1]. Se trata de un framework de desarrollo de aplicaciones móviles creado por Google, teniendo su primer reléase al público en general en mayo de 2017, pero, presentando su primer reléase estable el 22 de febrero de 2019. Flutter se encuentra optimizado para crear aplicaciones tanto Android, como iOS y de manera novedosa, incluyen el sistema operativo en desarrollo propio de Google, Google Fuchsia.



**Descripción:** Imagen 2. Logo Flutter

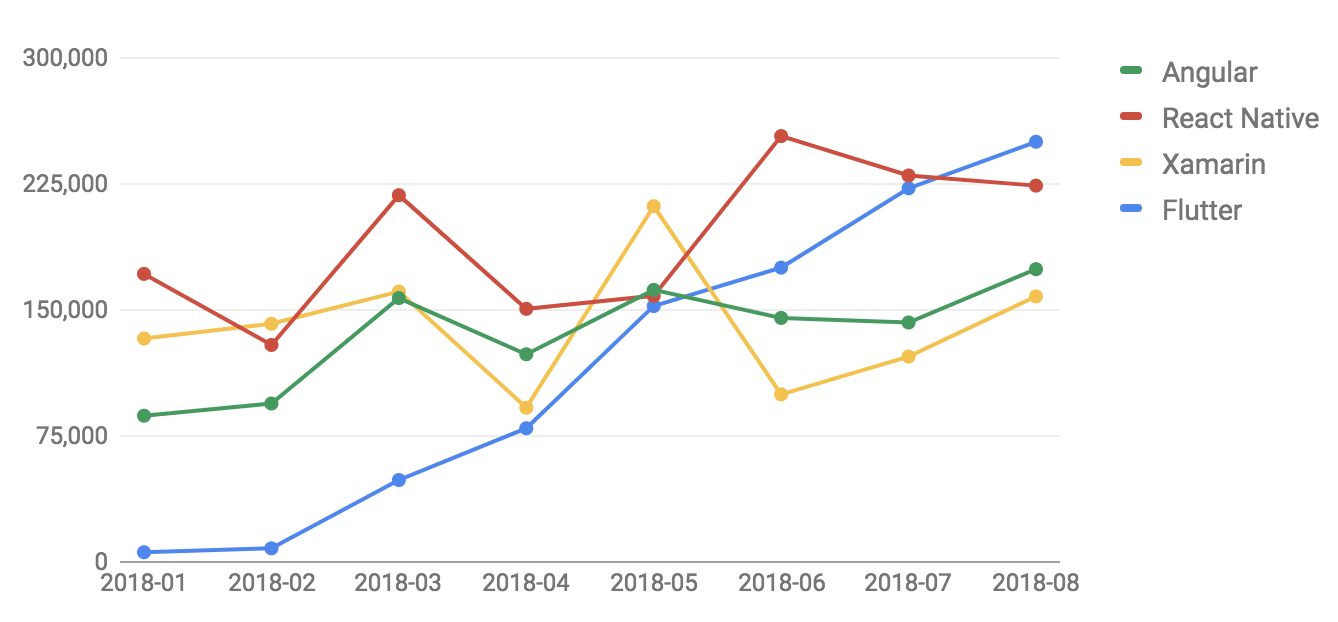
**Autor:** Google

**Dirección electrónica de la imagen:** <https://flutter.io/images/flutter-mark-square-100.png>

Una de las características ofrecidas por Flutter, es que como código fuente utiliza el lenguaje Dart, que es un lenguaje de propósito general, también creado por Google y que se utiliza para construir aplicaciones en distintas plataformas, como *web*, de servidores, de escritorio e incluso aplicaciones móviles. Dentro de sus atributos es posible mencionar que se trata de un lenguaje orientado a objetos, que utiliza definición por clases, cuenta con su propio Garbage Collector y su sintaxis es similar al lenguaje C. Para describir de mejor manera Flutter, es necesario describir sus componentes:

* **Flutter engine:** Es el *core* de Flutter, se encuentra desarrollado en C++ y proporciona soporte bastante amplio a bajo nivel del dispositivo, lo que permite el acceso al *hardware* del dispositivo. Se vincula de manera directa con los SDK de Android y de iOS.
* **Foundation library:** Son componentes escritos en Dart, que pone a disposición de los programadores, para que puedan reutilizarlo, se trata de clases y funciones básicas que son requeridas para ejecutar Flutter, un ejemplo de esto serían las APIs utilizadas para hacer llamadas a su *engine*.
* ***Widgets*:** El desarrollo de la interfaz de Flutter se basa en *widgets*. Cada componente visual que se puede apreciar es un *widget*. Dentro de las ventajas de Flutter, está en que es posible combinar distintos *widgets* con el objetivo de generar nuevas combinaciones, tal como las indique o requiera el programador. Dentro de los *widgets* se ofrece de manera original, los diseños visuales de Material Design (usado por Android) y de Cupertino (diseños usados por iOS).

Una vez descrito qué es Flutter y cómo es su arquitectura principal, el lector podría pensar: ¿por qué usar Flutter? ¿Por qué no simplemente utilizar algún otro framework de renderización *web*? [2] Pues existen muchas respuestas a estas preguntas, tales como el soporte de la comunidad, el auge de este nuevo framework, el cual puede notar su alta demanda mediante [3] la imagen 3. Otro motivo, es el respaldo que ofrece una empresa de gran trayectoria y actualización constante como lo es Google, quienes constantemente brindan soporte, implementación de nuevas características y por lógica, también la frecuente reparación de posibles *bugs*. Todo esto suena bien, pero va un poco por encima de lo que es en realidad la ventaja principal de Flutter, por lo que se requiere una especificación más técnica. Flutter no utiliza un render de tipo *web*, en cambio, utiliza un *canvas* optimizado en el que renderiza sus propios *widgets* que no es de tipo *web*, sino que con llamadas al sistema a bajo nivel, mediante la utilización de Dart. Esto permite tener acceso a la mayor cantidad de *hardware* del dispositivo (sensores, cámaras, actuadores), teniendo una renderización visual casi natural que es compatible con los sistemas operativos antes mencionados. Todas estas características dejan a Flutter como un fuerte candidato a considerar, sin mencionar que ya está siendo implementado por grandes compañías como lo son el propio Google, Tencent, Google Ads y, además, el gigante de ventas mundiales Alibaba, quien recientemente implementó Flutter para su aplicación móvil, que cuenta con más de 50 millones de descargas.



**Descripción:** Imagen 3. Número de vistas de preguntas de StackOverflow etiquetadas con cada uno de los cuatro UI frameworks populares a lo largo del tiempo

**Autor:** Google

**Dirección electrónica de la imagen:** <https://1.bp.blogspot.com/-dAf9QhjGWiU/W6AvC4RRDoI/AAAAAAAAGC8/-LxXBV8saNA4smHV5shyRhkfqUHpWDq_wCLcBGAs/s1600/pasted%2Bimage%2B0%2B%252813%2529image4.png>

# Conclusiones

* El hecho de decidir entre qué tipo de implementación realizar, ya sea nativa, híbrida o generada, es una decisión importante que deben tomar los desarrolladores y encargados de sistemas, debiendo evaluar sus características ofrecidas y las necesidades que deben cubrir.
* Flutter es un framework de desarrollo potente, de gran auge que permite a los programadores, simplificar el tiempo y complejidad del proceso de escritura de código.
* La forma de generar aplicaciones a través de Flutter es lo más similar a una aplicación nativa, guardando los accesos a *hardware* y la fluidez de las interfaces gráficas, permitiendo además la generación de aplicaciones multiplataforma en un único código fuente.

# Referencias

* [1] **Tipo de Fuente:** Sitio Web

**URL: \*** <https://medium.com/asos-techblog/flutter-vs-react-native-for-ios-android-app-development-c41b4e038db9>

**Nombre de Artículo:** \* *Flutter: the good, the bad and the ugly* (23/11/2018)

**Nombre de Página Web:** Medium

**Autor(es):** Marco Bellinaso

**Último Acceso:** 29 de marzo de 2019

* [2] **Tipo de Fuente:** Sitio Web

**URL: \*** <https://www.northware.mx/desarrollo-de-aplicaciones-moviles-hibridas/>

**Nombre de Artículo: \*** Desarrollo de aplicaciones móviles híbridas ¿Me conviene? (12/09/2017)

**Nombre de Página Web:** Northware

**Autor(es):** Guillermo Martínez

**Último Acceso:** 29 de marzo de 2019

* [3] **Tipo de Fuente:** Sitio web

**URL: \*** <https://developers.googleblog.com/2018/09/flutter-release-preview-2-pixel-perfect.html>

**Nombre de Artículo: \*** *Flutter Release Preview 2: Pixel-Perfect on iOS* (19/09/2018)

**Nombre de Página Web:** Google Developers

**Autor(es):** Flutter Team

**Último Acceso:** 29 de marzo de 2019