Abraham Castañeda Quintero

Objective-c vs Swift.

Características	Objective-c	Swift
Performance	✔Posee funciones del mismo lenguaje de C	✔ Alto rendimiento y mejorado
Seguridad	❌Objective-C es que utiliza punteros nulos. El puntero es el componente de C ++ y otros lenguajes basados en C y puede causar vulnerabilidades en la seguridad.	✔ Swift, por otro lado, no usa punteros. Si pierde un puntero en el código, tal vez un valor nulo, la aplicación se bloqueará. Este enfoque permite a los programadores encontrar y corregir errores rápidamente.
Mantenimiento	❌Objective-C se creó a partir de C y depende de él cuando se trata de cambios y mejoras. Los desarrolladores deben mantener dos archivos de código separados en Objective-C para mejorar la eficiencia y el tiempo de desarrollo de una aplicación.	✔Swift, como muchos idiomas modernos, es más fácil de mantener. El compilador LLVM y Xmind determinan los requisitos y completan automáticamente las compilaciones incrementales.
Sintaxis	❌Objective-C tiene una estructura de código compleja ya que se basa en el lenguaje C. Incluye muchos símbolos @, líneas, punto y coma y paréntesis condicionales con declaraciones internas "if" y "else".	✔Una de las razones por las que Swift se hizo tan popular es su sintaxis simple, que hace que el lenguaje sea fácil de leer y escribir. Al contrario de la complicada estructura de Objective-C, Swift evita numerosos símbolos @ y usa la lista de parámetros separados por comas entre paréntesis.
Complejidad del código	❌Si miramos más de cerca el código de Objective-C, veremos que las cadenas de texto son muy detalladas y requieren muchos pasos para vincular dos piezas de información. Los desarrolladores tienen que usar tokens de cadena especiales como% s,% d,% @, y proporcionar una lista de variables separadas por comas para reemplazar cada token.	✔Swift requiere mucho menos código para declaraciones repetitivas y manejo de cadenas. Esto se debe a las características de Swift, como la adición de dos cadenas mediante el operador "+". Swift utiliza la interpolación de cadenas que excluye la necesidad de memorizar tokens, por lo que los desarrolladores pueden insertar variables directamente en línea con las cadenas, como una línea de botón o una etiqueta.

Abraham Castañeda Quintero

Gestión de la memoria	❌Objective-C utiliza el ARC compatible con Cocoa API . ARC es una función para los lenguajes Objective-C y Swift que administra la memoria sin esfuerzo del programador.	✔El lenguaje Swift también usa ARC. La diferencia es que Swift admite ARC para todas las API que permiten una forma simplificada de administración de memoria similar a Cocoa Touch.
Soporte de bibliotecas dinámicas	❌Objective-C no admite bibliotecas dinámicas y esta es una gran desventaja. El caso es que son más grandes porque los programas externos están integrados en los archivos ejecutables.	✔Las bibliotecas dinámicas compatibles con Swift se cargan directamente en la memoria de una aplicación y optimizan el rendimiento de la aplicación.
Perspectiva a largo plazo	❌Muchos desarrolladores no quieren cambiar de Objective-C porque ya han invertido mucho tiempo en aprenderlo y desarrollar aplicaciones en él. Apple aún no ha proporcionado la fecha presunta para cuando planean dejar de ser compatible con Objective-C, pero podemos suponer que nunca lo harán.	✔Swift es un idioma joven que está creciendo muy rápido. Tiene una comunidad vibrante y fuerte que contribuye activamente a su formación. En esta etapa de desarrollo, Swift no se limita solo al ecosistema de Apple.
Reconocimiento comunitario	✔Muchos desarrolladores todavía usan Objective-C por muchas razones. La más común sería que muchos de ellos son expertos en el idioma. En este caso, los beneficios existentes de cambiar a Swift pueden no ser suficientes para superar los costos de aprender el nuevo idioma.	✔Dado que Swift es de código abierto, obtuvo un notable apoyo de la comunidad, lo que le permite a Apple corregir errores de manera oportuna. Apple puede recopilar los comentarios e implementar las mejoras en función de ellos.

Abraham Castañeda Quintero

Frameworks para iOS.

Framework MapKit	En esta lección vemos el Framework MapKit, con el que insertaremos una vista de mapa en muestras Apps y mostraremos en el mapa la localización del usuario. MapKit es una API de Apple que nos permite trabajar con mapas de una forma muy sencilla, en la clase de hoy veremos como integrar una vista de mapa en una vista.
Quicklook	El Framework Quicklook es un pequeño repositorio de clases y protocolos que nos va a permitir básicamente orquestar todo el funcionamiento de mostrar documentos en una vista especial que Apple ha creado para nosotros.
Uikit Dinamycs	Para aplicar estos efectos vamos a hacer uso de un animador, es decir un objeto de la clase UIDynamicAnimator. A este animador le vamos a ir añadiendo los diferentes efectos y los coordinará para que se pongan en funcionamiento.
Alamofire	Alamofire añade una capa por encima de las clases nativas de Apple para otorgar de una flexibilidad, pero sobre todo de una sencillez inaudita en lo que a realizar llamada a un servidor se refiere. Dentro del Framework tenemos funciones que nos permiten realizar llamadas de una forma muy sencilla