

**INSTITUTO SUPERIOR TECNOLÓGICO
DEL SUR**

CARRERA

DISEÑO Y PROGRAMACIÓN WEB

LENGUAJE WEB III

**“Lenguajes de Programación del lado del
servidor”**

PROFESOR(A): Amado Cerpa Juan Andrés

ALUMNO: Vilca Apaza Christian

SEMESTRE : V

19/05/2021

Índice

Dart.....	3
¿Dónde puedo utilizar Dart?	3
Flutter Framework de Dart	4
Ventajas	4
Desventajas.....	4
Sintaxis de Dart	5
Go(Golang o Google Go)	6
Características de Go	6
Ventajas de Go	6
Desventajas de Go	7
Sintaxis de Go	7
Ruby	8
Características.....	8
Ventajas	8
Desventajas.....	8
Sintaxis.....	9
Python	10
Características.....	10
Ventajas de Python	10
Desventajas de Python.....	11
Sintaxis.....	12
Javascript.....	13
Nodejs	13
Ventajas	13
Desventajas.....	14
Sintaxis.....	15
Conclusión	16
Bibliografía.....	17

Dart

El lenguaje de programación Dart fue desarrollado principalmente por Google. Dart es un estándar Ecma, la organización europea para la estandarización de sistemas informáticos y de comunicación y productos electrónicos.

La programación Dart es una alternativa interesante a JavaScript en los navegadores web actuales. De acuerdo con los desarrolladores de Dart, ya no es posible subsanar las deficiencias de JavaScript mediante el desarrollo del lenguaje.

El lenguaje Dart de Google empezó a desarrollarse en 2010 y se presentó un año después. Como los navegadores no podían, ni pueden, trabajar con este lenguaje de forma natural, y JavaScript puede ejecutarse en todos los navegadores actuales, existe el compilador Dart2js, es decir, "Dart para JavaScript". El lenguaje Dart se asemeja a los ya establecidos lenguajes de programación orientados a objetos, entre los que se encuentran Swift, C# o Java, que se subordinan a determinados paradigmas de programación. Las reglas para combinar caracteres definidos, es decir la sintaxis, son similares al lenguaje C. Esta semejanza facilita enormemente el aprendizaje, de manera que es posible iniciarse en él sin tener que enfrentarse a grandes problemas de lenguaje.

DartEditor es el primer editor lanzado por Google (noviembre de 2011) para escribir aplicaciones Dart. Es un editor ligero de código abierto que incluye todas las herramientas necesarias para desarrollar, analizar y depurar las aplicaciones. Permite crear y editar los ficheros y gestionar los directorios de los proyectos y soporta resaltado de sintaxis y auto-completado de código. Además es posible navegar y buscar cualquier elemento que necesites del API de Dart, así como establecer puntos de control y depurar (hacer debug).

Dos años después (noviembre de 2013) el equipo de desarrolladores anuncia que está trabajando en Spark, un nuevo IDE basado en el browser (es una Chrome app) para construir Chrome apps. Ha sido desarrollado con Dart y utiliza el framework Polymer

¿Dónde puedo utilizar Dart?

Dart es un lenguaje de propósito general, y lo puedes utilizar casi para cualquier cosa:

En aplicaciones web, utilizando la librería de arte: html y el transpilador para transformar el código en Dart en JavaScript, o utilizando frameworks como AngularDart.

En servidores, utilizando las librerías de arte: http y arte: io. También hay varios frameworks que se pueden utilizar, como por ejemplo Aqueduct.

En aplicaciones de consola.

En aplicaciones móviles gracias a Flutter.

Flutter Framework de Dart

Flutter es un framework de Dart para crear aplicaciones multiplataforma con un único código. A diferencia de otros frameworks multiplataforma como por ejemplo Ionic, el código de una aplicación de Flutter se compila a código nativo, por lo que el rendimiento alcanzado es superior a aplicaciones basadas en web-views. Además, a diferencia de React Native, Flutter no utiliza componentes nativos, sino que viene con sus propios componentes, llamados widgets, por lo que la misma aplicación se verá igual en cualquier dispositivo, independientemente de su sistema operativo o la versión. Gracias a ello, el desarrollador no tiene que preocuparse por que el diseño de su aplicación se vea mal en dispositivos antiguos.

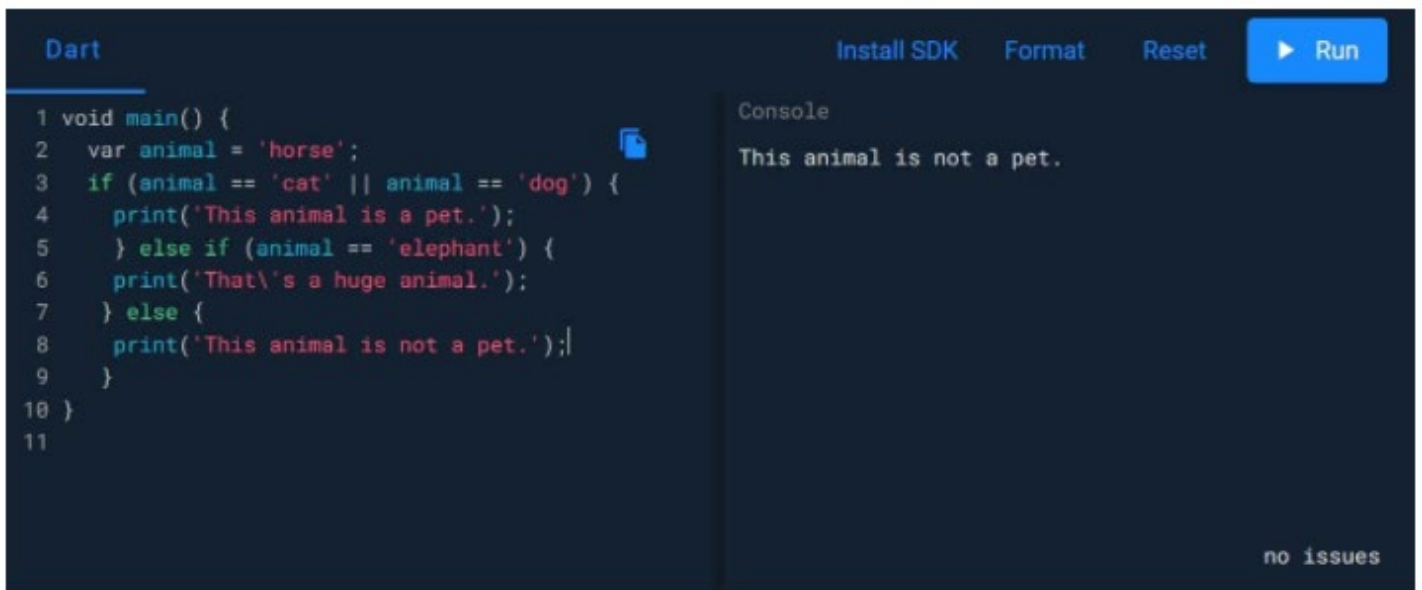
Ventajas

Dart es un lenguaje de código abierto y, por tanto, de acceso gratuito para cualquier persona. Detrás de la programación Dart se encuentra el gigante Google, lo que ofrece perspectivas a largo plazo para el desarrollo del lenguaje. Debido a su sintaxis, este lenguaje es fácil de aprender para los programadores debido a que los desarrolladores han simplificado muchas características complicadas de otros lenguajes y las han combinado de manera inteligente. Quien ya conozca el lenguaje C# no tardará en familiarizarse con Dart. El lenguaje de programación ha sido desarrollado para la web y mediante traducción directa (y muy rápida) a JavaScript, Dart funciona en todos los navegadores móviles y de escritorio actuales. Para las tareas de desarrollo, basta con un editor de texto sencillo, aunque para estas son necesarios conocimientos más exhaustivos del lenguaje. De última, Dart de Google hace más sencillo el trabajo con editores especiales como Android Studio (Google) o Visual Studio Code (Microsoft).

Desventajas

El lenguaje de programación Dart es relativamente nuevo, lo que implica una comunidad de soporte aún bastante reducida y una disponibilidad de materiales de aprendizaje inferior a JavaScript. No obstante, es de esperar que crecerá y mejorará a corto plazo. Aunque la instalación inicial de un editor en un ordenador y sus detalles técnicos estén bien documentados, también está lleno de obstáculos. Asimismo, muchos críticos juzgan negativamente la existencia de un nuevo lenguaje en el mercado en lugar del perfeccionamiento de los ya existentes.

Sintaxis de Dart



The screenshot shows a Dart IDE interface. The top bar contains the text 'Dart' on the left, and 'Install SDK', 'Format', 'Reset', and a blue 'Run' button with a play icon on the right. The code editor on the left contains the following Dart code:

```
1 void main() {  
2   var animal = 'horse';  
3   if (animal == 'cat' || animal == 'dog') {  
4     print('This animal is a pet.');5   } else if (animal == 'elephant') {  
6     print('That\'s a huge animal.');7   } else {  
8     print('This animal is not a pet.');9   }  
10 }  
11
```

The console on the right, titled 'Console', displays the output: 'This animal is not a pet.'. At the bottom right of the console area, it says 'no issues'.

Go(Golang o Google Go)

Go, también conocido como Golang o Google Go, nace en el año 2007 y fue desarrollado por los ingenieros Rob Pike, Ken Thompson y Robert Griesemer, quienes buscaban que este fuera un lenguaje escalable como lo es C++ y Java, sin embargo su lanzamiento fue llevado a cabo por Google en el año 2009. Go, es un lenguaje de programación concurrente, compilado de código abierto (open source) relativamente nuevo ya que data de once años de antigüedad hablando históricamente, donde su trayectoria ha permitido ir mejorando, puliendo y madurando todos sus detalles. Se encuentra inspirado en la sintaxis de lenguaje C y Algol, con tipado estático.

Características de Go

Siguiendo el patrón y objetivo de que este nuevo lenguaje lograra resolver problemas comunes en otros lenguajes de programación, el equipo desarrollador de Go decidió conservar ciertas características, y con el resurgimiento de nuevas mejoras que le permitirían apalancarse en el mercado. A continuación, te nombraremos algunas de las características más relevantes de Go:

- Soporta redes.
- Es multiprocesador.
- Cuenta con un garbage recollector (Recolector de Basura).
- Es multiparadigma, lo que le permite llevar a cabo programación de forma estructurada, orientada a objetos, etc.
- Es concurrente.
- Curva de aprendizaje aplanada.
- Sintaxis concisa y clara, es similar a la sintaxis de lenguaje C.
- Es opensource.

Adicionalmente a las múltiples características que presenta Go, este cuenta con las siguientes herramientas:

- Godoc: Permite mostrar la documentación por medio del HTTP.
- Go Vet: Herramienta encargada de buscar posibles errores dentro del código.
- Go Get: Con ella es posible instalar y/o recuperar paquetes.
- Go Build: Utilizando la data de origen permite generar binarios.
- Go fmt: Herramienta para dar formato al código.

Ventajas de Go

Gracias a que el equipo desarrollador de Go tenía como uno de sus objetivos que este lenguaje de programación fuese de fácil aprendizaje y de simple codificación, sin dejar de ser robusto, lograron obtener un producto de gran eficiencia en cuanto al procesamiento de grandes escalas de datos, siendo este punto una de las grandes ventajas con las que cuenta Golang.

En este mismo orden de ideas, gracias a su potente núcleo de librerías podemos llevar a cabo el manejo y procesamiento a gran escala de datos gracias a el Big data, es posible hacer encriptados y web api lo que hará que solo utilicemos las capas necesarias y que nuestro proyecto sea más rápido y

ligero, sin la necesidad de utilizar librerías de terceros o un framework. Es importante destacar que Go cuenta con las siguientes librerías:

- http.
- encryp.
- sql.
- pprof.
- mail.
- trace.
- elf.
- macho.
- encoding.

Desventajas de Go

Al hablar de las desventajas de Go, podemos mencionar que este lenguaje de programación no cuenta con documentación extensa en cuanto a manuales, procedimientos, libros e informes, sin embargo la documentación que podemos ubicar en la Web oficial de Go es de mucha ayuda a la hora de llevar a cabo un proyecto con él. Por otra parte, no es posible implementar herencias si no instanciamentos de un type. Los tipos de datos genéricos sólo es posible utilizarlos con su biblioteca estándar, mediante funciones en conjunto con las interfaces.

Por ser un lenguaje de programación que no cuenta con tanta antigüedad como Java, PHP, C#, entre otros., su comunidad es pequeña y se encuentra en crecimiento. Por ende, en el mercado hay carencia de programadores expertos en este lenguaje.

Sintaxis de Go

```
1 import (  
2     "fmt"  
3     "log"  
4     "net/http"  
5 )  
6  
7 func holaFunc(w http.ResponseWriter, r *http.Request) {  
8     fmt.Fprintf(w, "Hola mundo!")  
9 }  
10  
11 func main() {  
12     http.HandleFunc("/", holaFunc)  
13     log.Fatal(http.ListenAndServe(":8080", nil))  
14 }
```

Ruby

Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos, creado por el programador japonés Yukihiro "Matz" Matsumoto, quien comenzó a trabajar en Ruby en 1993, y lo presentó públicamente en 1995. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos similares a Smalltalk. Comparte también funcionalidad con otros lenguajes de programación como Lisp, Lua, Dylan y CLU. Ruby es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre.

Características

- Orientado a objetos
- Cuatro niveles de ámbito de variable: global, clase, instancia y local.
- Manejo de excepciones
- iteradores y clausuras o closures (pasando bloques de código)
- expresiones regulares nativas similares a las de Perl a nivel del lenguaje
- Posibilidad de redefinir los operadores (sobrecarga de operadores)
- recolección de basura automática
- Altamente portable
- Hilos de ejecución simultáneos en todas las plataformas usando hilos verdes, o no gestionados por el sistema operativo.
- Carga dinámica de DLL/bibliotecas compartidas en la mayoría de las plataformas
- Introspección, reflexión y metaprogramación
- Amplia librería estándar
- Soporta inyección de dependencias
- Soporta alteración de objetos en tiempo de ejecución continuaciones y generadores

Ventajas

- Cuenta con código libre.
- Tiene una extensa comunidad detrás de él que la respalda.
- Favorece en el ahorro de líneas de código.
- Tiene una forma más fácil de interactuar con

Desventajas

- No es un lenguaje muy conocido.

Sintaxis

```
$ irb
irb(main):001:0> puts "Hola mundo"
Hola mundo
=> nil
irb(main):002:0> 1+2
=> 3
```

Python

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma. Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License.

Características

Las características del lenguaje de programación Python se resumen a continuación:

Es un lenguaje interpretado, no compilado, usa tipado dinámico, fuertemente tipado.

Es multiplataforma, lo cual es ventajoso para hacer ejecutable su código fuente entre varios sistemas operativos.

Es un lenguaje de programación multiparadigma, el cual soporta varios paradigma de programación como orientación a objetos, estructurada, programación imperativa y, en menor medida, programación funcional.

En Python, el formato del código (p. ej., la indentación) es estructural.

Ventajas de Python

Hablemos primero de las características de Python que suponen grandes ventajas:

- **Propósito general**
Es un lenguaje de propósito general, es decir, permite crear programas de cualquier tipo: desde aplicaciones de Data Science hasta páginas web completas.
- **Open source**
Es open source, es decir, es gratuito y puedes utilizarlo de forma libre en cualquier sistema operativo. Además, el hecho de ser open source ha favorecido la creación de una enorme comunidad detrás de este lenguaje, gracias a lo cual el lenguaje nunca queda obsoleto y va recibiendo actualizaciones periódicas.
- **Portable**
Puede ser utilizado en todos los sistemas operativos: Windows, Linux, etc.
- **Simple**
Es muy sencillo y fácil de aprender, lo que hace posible que cualquier principiante pueda empezar a desarrollar programas de manera muy rápida.
- **Sintaxis clara**

Python es un lenguaje ordenado y limpio, y tiene una sintaxis clara, obligando a que se respete una correcta indentación en todo el código. Esto hace que todos los programas escritos en Python tengan una estructura unificada y fácil de entender.

- **Alto nivel**
Es un lenguaje de alto nivel, lo que se traduce en una mayor facilidad de uso. Con un lenguaje de bajo nivel tendrías que preocuparse de gestionar algunos asuntos más complejos, como por ejemplo la memoria del programa, mientras que con Python esto no es necesario.
- **Orientado a objetos**
Es un lenguaje orientado a objetos, es decir, está basado en objetos que agregan tanto datos, como funcionalidades. Normalmente, se considera que la programación orientada a objetos favorece la reusabilidad, el mantenimiento y la fiabilidad del código.
- **Multiparadigma**
Además, no solo ofrece programación orientada a objetos, sino también programación estructurada, imperativa y funcional. Por ello, se le considera un lenguaje multiparadigma.
- **Interpretado**
Es un lenguaje interpretado, es decir, no hace falta compilarlo. Esto se traduce en un desarrollo mucho más rápido.
- **Incrustable**
Es fácilmente incrustable, gracias a lo cual es posible incorporar programas escritos en Python a otros programas escritos en lenguajes diferentes, como C y C++.
- **Fuerte tipado dinámico**
Es de tipado dinámico, lo que significa que el valor que tienen asociado las variables puede cambiar en pleno tiempo de ejecución. Además, está fuertemente tipado, y esto implica que el tipo de valor no cambia repentinamente, es decir, una cadena de texto que tiene almacenado un único número entero no se convertirá en un número sin haber realizado antes una conversión de tipo (paso de string a tipo int, en este caso).
- **Librerías**
Hay un montón de librerías disponibles que incorporan muchísimas funcionalidades extras.

Desventajas de Python

- **Ejecución más lenta que en los lenguajes no interpretados**
- **Aprendizaje para el desarrollo web**
A pesar de que es perfectamente posible desarrollar webs completas en Python, la cosa se complica cuando lo que queremos es construir webs con funcionalidades un poco más específicas. Para intentar solucionar este inconveniente, existen frameworks como Django, que fomentan el desarrollo mucho más rápido al proporcionar módulos y guías ya preparados para el formato web en Python.
- **Hosting**

No todos los servicios de hosting están preparados para soportar aplicaciones en Python.

Sintaxis

```
def calculo(op,a,b):  
    return {  
        'sum': lambda: a + b,  
        'rest': lambda: a - b,  
        'mult': lambda: a * b,  
        'div': lambda: a/b  
    }.get(op, lambda: None)()  
  
print(calculo('sum',3,4))
```

Javascript

Javascript es un lenguaje poderoso, capaz de aportar soluciones eficaces en la mayoría de los ámbitos de la tecnología.

Es especialmente importante porque es el único lenguaje de programación que entienden los navegadores, con el que se desarrolla la parte de la funcionalidad frontend en sitios web y aplicaciones web modernas. Pero también es fundamental en muchos otros tipos de desarrollos. Sus usos más importantes son los siguientes:

- Desarrollo de sitios web del lado del cliente (frontend, en el navegador)
- Desarrollo de todo tipo de aplicaciones gracias a la plataforma NodeJS
- Desarrollo de aplicaciones para dispositivos móviles, híbridas o que compilan a nativo
- Desarrollo de aplicaciones de escritorio para sistemas Windows, Linux y Mac, pudiendo escribir un código compatible con todas las plataformas.
- Por tanto, podemos considerar a Javascript el lenguaje universal, pues es el que más tipos de aplicaciones y usos que puede abarcar en la actualidad. Es por ello que resulta un lenguaje muy recomendable para aprender, ya que nos ofrece capacidades para usarlo en todo tipo de proyectos, siendo que algunos de ellos son parcela exclusiva de Javascript.

Nodejs

Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web.⁴ Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent, que además tiene contratado a Dahl en plantilla.

Ventajas

- ¿Está considerando crear aplicaciones de juegos o visitar aplicaciones? Para fabricar tales aplicaciones continuas, Node JS es la mejor alternativa. Las empresas más importantes en todo el mundo, como eBay, PayPal, LinkedIn, etc., son los resultados fructíferos del desarrollo con Node JS.
- Google pensó en el motor V8. Tiene una velocidad de ejecución sorprendente y Node JS la utiliza. Además, hace que el código en ejecución (compuesto por los especialistas al crear el programa) se forme más rápido. Los diseñadores solo necesitan concentrarse en componer código legítimo y esta etapa utilizará la velocidad del sistema.
- Se Enriquece el Intercambio.
- La creación de soluciones web efectivas utilizando esta plataforma se vuelve notablemente más sencilla, ya que incluye NPM (Node Package

Manager) con el repositorio de casi 50.000 paquetes. Los desarrolladores pueden compartir, actualizar o reutilizar fácilmente el código con la ayuda de este NPM incorporado.

- Base de Código Única y Potente.
- Otra razón por la que Node JS funciona como una tecnología innovadora en el campo del desarrollo web es que los desarrolladores pueden trabajar sin esfuerzo tanto en el lado del servidor como en el lado del cliente con codificación JavaScript. Esta característica no solo ayuda a ahorrar un tiempo valioso para los desarrolladores, sino que también ayuda a sincronizar diferentes datos automáticamente entre el lado del cliente y el lado del servidor.
- Ofrece Servicios como un Servidor Proxy.
- Si una aplicación web del lado del servidor se comunica de manera dedicada con diferentes recursos de terceros para recopilar datos, para almacenar imágenes, esta plataforma puede actuar como servidor proxy cuando una empresa no tiene suficiente infraestructura de proxy profesional.
- Secuencia de Datos Suave.
- Si desea acceder a un archivo en particular mientras lo carga, Node JS le permitiría hacerlo. Esta característica es extremadamente beneficiosa para los desarrolladores cuando trabajan con codificación de audio o video en tiempo real. Se asegura de disminuir todo el tiempo de procesamiento. Nada puede realmente superar la eficiencia de esta plataforma en el caso de la transmisión de datos de diversas fuentes.

Desventajas

- Seguridad Client-Side- Desde que el código en JavaScript es ejecutado en el client-side, bugs y descuidos pueden ser explotados algunas veces para malos propósitos. Por esto, algunas personas deciden desactivar JavaScript por completo.
- Soporte del navegador- Mientras server-side script siempre produce el mismo resultado, algunas veces diferentes navegadores interpretan el código JavaScript de manera distinta. Estos días las diferencias son mínimas, y no deberías tener que preocuparte mientras compruebes tu código en la mayoría de los navegadores.
- API Inestable
La API de Node tiene la mala costumbre de cambiar en formas que rompen la compatibilidad hacia atrás de versión en versión, lo que requiere que apliques cambios frecuentes en tu código para mantener todo funcionando en las versiones mas actuales. Aún así, se supone que será más estable desde la versión 0.2.0.
- Muchas Formas de Programar
La falta inherente de organización de código se puede considerar una gran desventaja. Se nota su efecto claramente cuando el equipo de desarrollo no está muy familiarizado con la programación asíncrona o los patrones de diseño estándar. Simplemente hay demasiadas formas de programar y de obtener código desparejo y difícil de mantener.

Sintaxis

```
const http = require('http');

const hostname = '127.0.0.1';
const port = 1337;

http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Hello World\n');
}).listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Conclusión

Estos lenguajes de programación son casi nuevos algunos que ofrecen un mejor ambiente para programar en el lado del servidor, el mas nuevo es dart que se quiere coronar como un lenguaje multiusos para todo prácticamente le quieres hacer competencia a JavaScript, Python y otros.

Cada uno tienes sus ventajas y desventajas, pero yo creo que como desarrollador se debería conocer un poco de todo y siempre estar al tanto de lo nuevo que sale, bueno o también dedicarse solo a JavaScript por que en ese lenguaje de programación cada hora creo q sale una nueva librería o framework o algo nuevo.

Bibliografía

<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/lenguaje-de-programacion-dart-de-google/>
<https://inlab.fib.upc.edu/es/blog/que-es-el-lenguaje-de-programacion-dart>
<https://es.wikipedia.org/wiki/Dart>
<https://openwebinars.net/blog/que-es-go/>
[https://es.wikipedia.org/wiki/Go_\(lenguaje_de_programaci%C3%B3n\)](https://es.wikipedia.org/wiki/Go_(lenguaje_de_programaci%C3%B3n))
<https://es.wikipedia.org/wiki/Ruby>
<https://jditic92.wordpress.com/2015/05/11/caracteristicas-ventajas-y-desventajas-de-las-tecnologias-de-desarrollo-ruby-on-rails-y-php/>
<https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion1/caracteristicas.html>
<https://es.wikipedia.org/wiki/Python>
<https://www.discoder.tech/python-ventajas-desventajas/>
<http://www.softwero.com/2014/08/las-desventajas-de-usar-nodejs-en-el.html>
<https://es.wikipedia.org/wiki/JavaScript>
<https://es.wikipedia.org/wiki/Node.js>
<https://www.openinnova.es/node-js-ventajas-y-desventajas-desarrollo/>