Como implementar Go en el desarrollo computacional.

Es de saber que a la fecha existen múltiples lenguajes que compiten diariamente por ser el mas usado, Go no es la excepción. Basta con hacer una búsqueda en el crecimiento de búsquedas de los lenguajes de programación actuales para entender que esta poco a poco teniendo un crecimiento. Y son muchas las empresas de renombre y lideres en sus propios mercados las que están empezando eventualmente a migrar sus tecnologías por implementaciones y paquetes de GoLang sin embargo existen decenas de herramientas en el mercado. Son millones de personas las que al día de hoy hacen uso de herramientas computacionales para desarrollar software y ser competitivo en un mercado que está constantemente cambiando es un reto el cual los desarrolladores de lenguajes deben afrontar. En mi experiencia personal no me he inclinado por un solo lenguaje, ya que cada herramienta provee soluciones diferentes a situaciones diferentes. Lenguajes de propósito general y lenguajes cuyo uso se ve orientado a un paradigma o a la resolución de un problema específico.

C y C++ como lenguajes de programación, los he usado previamente con el desarrollo orientado a hardware y a sistemas embebidos, Javascript y Typescrit orientados a desarrollo web y herramientas de entornos de trabajo o Frameworks de front-end.  
  
Por otra parte C#, GoLang y mencionando nuevamente Typescript los he usado como lenguajes de back-end o de desarrollo en sistemas que procesan data de y para una o varias bases de datos que conectan con algún entorno de front-end, he incluso he escuchado hablar de sistemas existentes mucho más eficientes que operan con C++.   
  
Cada una de estas herramientas permite lograr a través de un conjunto de paradigmas y de prácticas de programación lograr una lógica computacional para la resolución de problemas de la vida real, ya sean entretenimiento, ventas, publicidad o redes sociales. Sistemas que por su complejidad son capaces de generar un conjunto de necesidades y soluciones a los llamados usuarios, que son al final del día aquellos a los que se les venderá y/o expondrá el producto final (software).   
  
Ahora partiendo de la anterior introducción genero la siguiente pregunta problema:

¿Es Golang una herramienta apropiada en el desarrollo computacional?

En este escrito se expone, identifica y se entra a fondo en los conceptos principales de Go como lenguaje de programación y como herramienta de desarrollo computacional para así entender como la abstracción de problemas reales y las posibilidades que Go ofrece y permite desarrollar y crear “Software” que hace parte del diario vivir de cada uno de nosotros.

¿Porque Go? ¿Y por qué no?

La generación de binarios en sistemas computacionales de cualquier tipo es una característica nativa del lenguaje. Generar el ejecutable o el binario para que este se pueda correr en sistemas basados en Unix o en Windows es una gran ventaja sobre muchos otros lenguajes. También en sistemas como Google App Engine, un servicio de PAAS el sistema en si funciona excesivamente rápido dado a que dependencias que usa el lenguaje facilitan su extracción por este medio. (Steven Ellis) en su blog “10 Características de Go que lo separan de otros lenguajes” expone que esta característica en especifico facilita la distribución de paquetes en este lenguaje dada a la facilidad que tiene para ser ejecutado y generar un request. En comparación a códigos generados con Python o Node que en comparación demuestran un gran margen en velocidades.

Como Go es un lenguaje con dependencias no centralizadas el acceso es mucho más fácil ya que integra el uso del lenguaje con el acceso con comandos como “ Go install” facilitan enormemente este proceso. El uso de Git como herramienta de distribución de paquetes de manera publica permite una descentralización del manejo de dependencias. Es decir que los mismos usuarios determinan su usabilidad y escalabilidad con cada actualización del lenguaje. Hay que tener en cuenta que esta actualización solo se encuentra disponible en la versión 1.10 de Golang con Go Modules la cual prescinde del uso de GOPATH el cual generaba problemas al alocar paquetes necesarios en el código. Migrar de una metodología a la otra está a una búsqueda en Google de distancia.

Go es un lenguaje que no usa los Objetos como tipo de implementación de una clase, sino hace uso de las estructuras o Struct, estas a diferencia de lenguajes como Java, Python o Javascript usan los valores por defectos pasados por valor y no por referencia como los últimos tres mencionados.  
  
Eso significa que cualquier cambio hecho al objecto en la función que recibe esta reflejado en el cambio en el objecto original. En Go por el contrario las Estructuras o Struct son pasadas por valor, con la opción de pasar un puntero el cual accede al valor por referencia. Esto es un concepto clave en el desarrollo de software y uno el cual es aprendido tanto por practica y con el estudio de la teoría.

Como dice MIT “La mutabilidad hace difícil entender lo que tu programa hace, y mucho más difícil el reforzar contratos” .

Incluye la palabra reservada Defer que a mi visión es “Lo último en guarachas” como se dice coloquialmente en Colombia, o lo último en programación que permite la ejecución de una sección de código una vez se hayan ejecutado las funciones circundantes a esta. Esta característica facilita la limpieza o tear-down en un proyecto.

Adopta los paradigmas de POO además de hacer uso de funciones anónimas que retornan un valor inmediato o funciones declaradas en otras funciones que permiten el encapsulamiento que cambia la visibilidad de alguna variable o propiedad.

El lenguaje hace uso de interfaces de manera implícita. Esta propiedad permite el hecho de que si un método que debe recibir un tipo concreto de dato que contiene todas las propiedades puestas, Go infiere que la Estructura u Objecto implementa la interfaz.

También posee un manejo de errores nativo del lenguaje. Facilitando así el desarrollo. Este promueve la codificación de pruebas y su integración con el lenguaje mejorando su uso.

Es de entender que todas estas son características que diferencian a Go de lenguajes como Python, o Javascript, los cuales tienen sus propiedades y beneficios a la hora de usar. Y tendrá similitudes con C# e inclusive con Typescript el entender cada una de estas facilita que lenguaje usar en que situación.

La solución de problemas no depende solamente de las propiedades de un lenguaje de programación sino como cada una de estas puedan llegar a ser usadas en un sistema. Con todas las propiedades previamente mencionadas es probable que para la continuidad del curso se nos pida ir usando cada una de estas propiedades en los programas venideros. Sin embargo el aprovechamiento de esta poderosa herramienta depende estricta y únicamente de cada desarrollador.

Concluyendo la pregunta basada en el planteamiento. Sí, Golang es una herramienta apropiada en el desarrollo computacional. No por sus características innatas sino porque hereda muchas características de lenguajes como C++, C#, Java y Typescript pero implementadas de otras formas. Las características que lo difieren la hacen una herramienta mas robusta y con mas posibilidades.

A su vez GoLang carece de características que otros lenguajes si usan, como Generics o Genericos para poder acceder a los tipados propios del lenguaje, sobrecarga de métodos. No posee herencia realmente directa, posee la característica Composition que es lo mismo que tener un campo de una estructura tipo T, pero con los métodos de T estando disponibles. Esto no debería ser tan confuso como la herencia pero que es un punto en contra.   
La buena noticia es que constantemente se están sacando nuevas actualizaciones de este lenguaje. Creando nuevas características que facilitan desde algo tan básico como la sintaxis hasta algo tan complejo como es el manejo de memoria. En mi opinión personal pasa el test de lenguajes que deberías conocer o al menos lenguajes a los que valen la pena hacerles un “Hola Mundo!”.

Interpretación y recopilación.

Tesis

Argumentos

Conclusiones