پردیس علوم

دانشکده ریاضی، آمار و علوم کامپیوتر

بررسی زبان برنامه‌نویسی Go

نگارنده

سارا ابریشمی

استاد راهنما:

مهندس ابراهیم نقیب‌زاده مشایخ

پایان‌نامه برای دریافت درجه‌ی کارشناسی

در رشته‌ی علوم کامپیوتر

۱۳۹۶

مقدمه

از زمان پیدایش اولین زبان‌های برنامه‌نویسی تا کنون زبان‌های بسیاری ساخته و معرفی شده‌اند. اما تعداد اندکی از آن‌ها توانسته‌اند با گذار زمان محبوبیت خود را حفظ کنند. آن‌چه عموماً اتفاق افتاده از یاد رفتن و جایگزین شدن با زبان‌های رایج‌تر است. در این مطالعه ما به بررسی یکی از زبان‌های برنامه‌نویسی جدید می‌پردازیم. زبان برنامه‌نویسی Go در سال ۲۰۰۹ توسط شرکت گوگل[1] معرفی شده است. تلاش در این مقاله آشنایی با این زبان، بررسی ویژگی‌های آن و مقایسه‌اش با دیگر زبان‌هاست.

کلمات کلیدی: زبان برنامه‌نویسی - Go

[1] Google

پیش‌گفتار

زبان برنامه‌نویسی یک زبان رسمی با مجموعه‌ای از دستورهاست که می‌تواند خروجی‌های متفاوتی تولید کند. این دستورها برای کامپیوتر است تا بتواند الگوریتم خاصی را اجرا کند و امکان نوشتن برنامه جهت تولید نرم‌افزارهای جدید به وجود می‌آورد.

ساخت اولین زبان برنامه‌نویسی به قبل از اختراع کامپیوتر بازمی‌گردد، و برای هدایت رفتار ماشین‌هایی مانند دستگاه‌های نساجی اتوماتیک و نوازنده‌های پیانو به کار می‌رفت. هزاران زبان برنامه‌نویسی تا به حال به وجود آمده که عموماً در حوزه‌ی کامپیوترند و همچنان هم هرساله زبان‌های جدیدی به وجود می‌آیند.

تعریف هر برنامه به دو قسمت دستوری و معناشناسی تقسیم می‌شود. بعضی از زبان‌ها بر اساس یک سند مشخصات به وجود می‌آیند. به طور مثال زبان C بر اساس مشخصات استاندارد ISO به وجود آمده در حالی که برخی دیگر دارای پیاده‌سازی غالبی هستند که به عنوان مرجع مورد استفاده قرار می‌گیرد مانند زبان Pearl و برخی از زبان‌ها هر دوی این‌ها را دارا بوده و زبان پایه‌شان بر اساس یک استاندارد بوده و باقی موارد اضافه‌شده دارای پیاده‌سازی غالبی مشترکی است.

معمولاً هر زبان برنامه‌نویسی دارای یک محیط نرم‌افزاری برای وارد کردن متن برنامه، اجرا، کامپایلر و رفع اشکال آن هستند.

عموماً زبان‌های برنامه‌نویسی را به پنج نسل تقسیم می‌کنند:

نسل اول زبان ماشین همان زبان صفر و یک.

نسل دوم زبان‌هایی مانند اسمبلی[۲] که برای انسان قابل فهم‌تر بودند.

[2] Assembly

نسل سوم زبان‌هایی مانند کوبول[۳] و پی ال وان[۴] و ... که دستورهایشان برای انسان قابل فهم‌تر بوده و همچنین برای اجرا به کامپایلرها نیاز داشته‌اند.

نسل چهارم مثل زبان‌های اوراکل[۵] و فاکس پرو[۶] و اس‌کیوال[۷]ها که بسیار به زبان محاوره‌ای انسانی نزدیک‌ترند.

نسل پنج زبان‌هایی مانند پرولوگ[۸] ، اُپی‌اس۵[۹] که تمرکزشان بر حل مسئله با توجه به محدودیت‌های موجود به جای استفاده از الگوریتم‌های نوشته‌شده توسط برنامه‌نویس است.

زبان‌های برنامه‌نویسی را می‌توان از چهار دیدگاه متفاوت مورد بررسی قرار داده و تقسیم‌بندی کرد:

الف) روش‌های برنامه‌نویسی: ۱. زیرروالی ۲. ساخت‌یافته ۳. ماژولار ۴. شی‌گرا.

ب) نزدیکی به زبان ماشین: ۱. سطح پایین ۲. سطح میانی ۳. سطح بالا.

ج) نوع ترجمه: ۱. مفسری ۲. کامپایلری.

د) رابط برنامه‌نویسی: ۱. مبتنی بر متن ۲. مبتنی بر گرافیک (ویژوال).

بر اساس این تقسیم‌بندی‌ها زبان برنامه‌نویسی Go یک زبان رویه‌ای، سطح بالا، کامپایلری و مبتی بر متن است.

زبان برنامه‌نویسی Go یک زبان متن‌باز نسل چهارم است که در سال ۲۰۰۷ توسط توسط رابرت گریزمر[۱۰]، راب پایک[۱۱] و کن تامپسون[۱۲] مهندسان شرکت گوگل ساخته و در سال ۲۰۰۹ معرفی شد.

این زبان در اصل یک آزمایش توسط شرکت گوگل بود برای طراحی زبان برنامه‌نویسی جدیدی که با وجود تصحیح مشکلات اصلی موجود در زبان‌های برنامه‌نویسی موجود خصوصیات مثبت آن‌ها را هم حفظ کند.

این زبان باید چند ویژگی مهم داشته باشد:

۱. به صورت استاتیک نوشته شده باشد و بتواند مانند Java و C++ با سیستم‌های بزرگ متناسب شود.

۲. به محیط‌های توسعه‌ی یکپارچه نیازمند[۱۳] نباشد اما از آن‌ها پشتیبانی کند.

---

[3] Cobol
[4] PL1
[5] Oracle
[6] Fox Pro
[7] SQL
[8] Prolog
[9] Ops5
[10] Robert Griesemer
[11] Rob Pike
[12] Ken Thompson
[13] integrated development environments

۳. از شبکه[۱۴] و چندپردازی[۱۵] پشتیبانی کند.

جالب است بدانیم یکی از انگیزه‌های هر سه طراح این زبان مشکلشان با پیچیدگی‌های زبان C++ بوده است.

[14] Networking

[15] Multiprocessing

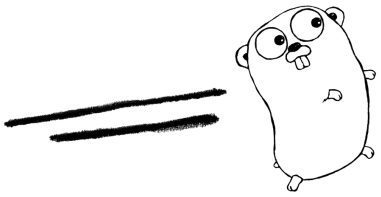
فهرست مطالب

فهرست تصاویر

فصل ۱

معرفی زبان Go

۱.۱ هدف از ایجاد زبان



شکل ۱.۱: لوگوی زبان Go

ساخت زبان برنامه‌نویسی Go اواخر سال ۲۰۰۷ آغاز می‌شود تا جوابی باشد برای مشکلاتی که برای توسعه‌ی زیرساخت‌ها نرم‌افزار در شرکت گوگل موجود بود. این روزه می‌توان گفت چشم‌انداز محاسباتی مرتبط با محیطی که زبان در آن استفاده می‌شود نیست و بیشتر برای زبان‌های C++، Java و پایتون[1] هستند. مشکلاتی که توسط پردازنده‌های چندهسته‌ای، سیستم‌های شبکه‌ای، خوشه‌های عظیم محاسباتی و برنامه‌نویسی تحت وب ایجاد شده دور زده شده‌اند و مستقیم به حل آن‌ها پرداخته نشده است.

مقیاس تغییر کرده است: سرورهای برنامه‌های امروزی شامل میلیون‌ها خط کد می‌شوند که صدها یا هزاران برنامه‌نویس روی آن‌ها کار کرده‌اند و روزانه به‌روزرسانی می‌شوند. و در این شرایط زمان ساخت حتا برای مجموعه خوشه‌های بزرگ تا دقایق یا ساعت‌ها طول می‌کشد.

زبان Go طراحی و تولید شد تا بتواند کار در این محیط‌ها را سازنده‌تر کند. جدای از وجهه‌های شناخته‌شده آن مانند همروندی داخلی و زباله‌جمع‌کن، طراحی Go مدیریت شدید وابستگی‌ها، وفق‌پذیری معماری نرم‌افزار

[1] Python

با رشد سیستم و نیرومندی در مرز بین قسمت‌ها نیز در ای زبان در نظر گرفته شده است.
Go ساخته شد تا به گوگل کمک کند مشکلاتش را حل کند و گوگل مشکلات بسیاری داشت:
سخت‌افزار و نرم‌افراز هر دو بسیار بزرگ‌اند و میلیون‌ها خط کد در نرم‌افراز موجود است و سرورها بیشتر به زبان C++ و بسیاری از قسمت‌های دیگر Java و پایتون هستند. هزاران مهندس به عنوان سرپرست یک بخش کلی شامل تمام نرم‌افزارها روی کدها کار می‌کنند و از روزی به روز دیگر تغییرات محسوسی در سطوح مختلف این بخش به وجود می‌اید. یک سیستم توزیع‌شده با طراحی مخصوص می‌تواند این پیشرفت‌ها را امکان‌پذیر کند اما همچنان این سیستم بسیار بزرگ است.
به طور خلاصه توسعه در گوگل زیاد است. گاهی کند و بیشتر مواقع زمخت اما همیشه تاثیرگذار است.
هدف پروژه‌ی Go کاهش این کندی‌ها و زمختی‌ها از توسعه‌ی نرم‌افزاری شرکت گوگل بود تا بتواند این پروسه را سازنده‌تر و مقیاس‌پذیر کند. این زبان توسط و برای کسانی نوشته شده که در کار نوشتن - خواندن و مشکل‌زدایی از - سیستم‌های بزرگ نرم‌افزاری هستند.
پس هدف این زبان تحقیق درباره‌ی طراحی زبان‌های برنامه‌نویسی نیست بلکه برای اصلاح و بهتر کردن محیط کاری برای طراحان آن و همکارانشان است. Go بیشتر مربوط به مهندسی نرم‌افزار است تا تحقیق زبان برنامه‌نویسی. یا بهتر است بگوییم راجع به طراحی زبان در خدمت مهندسی نرم‌افزار است.
زمانی که Go ارایه شد برخی ادعا کردند که بعضی امکانات و متدولوژی‌های خاص زبان‌های جدید که به آن‌ها سختگیری[۲] را کم دارد و چگونه ممکن است این زبان با نداشتن این خصوصیات ارزشمند باشد؟ جواب پدیدآورندگان Go به این سوال این است که خصوصیاتی که این زبان دارد باعث می‌شود بتواند مشکلاتی را که توسعه‌ی نرم‌افزار در مقیاس بزرگ دارد حل کند. این مشکلات عبارت‌اند از:
کندی در ساخت؛
وابستگی‌های کنترل‌نشده؛
این که هر برنامه‌نویس از یک زیرمجموعه از زبان استفاده می‌کند؛
مشکل در فهم کد (کدها سخت خوانده می‌شوند. بد ثبت می‌شوند و...
تکثر تلاش؛
قیمت به‌روزرسانی؛

[2] de rigueur

نسخه‌ی انحرافی؛

سختی نوشتن ابزار اتوماتیک؛

ساخت‌های بین‌زبانی.

امکانات مجزای یک زبان امکان حل این مشکلات را ندارد. دید بهتری از مهندسی نرم‌افزار مورد نیاز است و در طراحی Go تلاش شده است تا بر حل این مشکلات تمرکز شود.

به طور مثال در مورد چگونگی نمایش ساختار برنامه: برخی منتقدان به رویکرد Go در این زمینه و استفاده از بلوک‌ها و براکت‌ها مانند زبان C و عدم استفاده از فاصله برای نشان دادن دندانه‌ها مانند پایتون و هسکل[۳] معترض بودند. اما در این زمینه هم بررسی‌های زیادی صورت گرفته و چه بسا دیده شده برنامه‌هایی که از روی یک اشتباه در زمان تغییر برنامه با مشکلات بزرگی روبه‌رو شده‌اند. اگرچه فاصله برای برنامه‌های کوچک بسیار زیباست اما در مقیاس بزرگ نتیجه‌بخش نیست و هر چه اندازه‌ی برنامه بزرگ‌تر باشد مشکلات بیشتری به وجود می‌آورد. به همین دلیل ترجیح داده شده که برای امنیت بیشتر ساختارها این‌گونه باشند.

هنگامی که سرعت کامپایل کم باشد زمان برای فکر کردن هست. افسانه‌ی Go این است که در زمان یکی از همین ساخت‌های ۴۵ دقیقه‌ای ایده‌ی ساختش به ذهن آمده. باور بر این بودخ که تلاش برای طراحی یک زبان جدید که مناسب نوشتن برنامه‌های بزرگ گوگل مثل سرور وب باشد ارزشش را دارد. با استفاده از مهندسی نرم‌افزار که می‌تواند اوضاع برنامه‌نویسان گوگل را هم بهبود ببخشد. مهم‌ترین ملاحظات برای ساخت زبانی با خصوصیات گفته‌شده این‌هاست:

باید در مقیاس بزرگ کار کند؛ برای برنامه‌های بزرگ با مقدار زیاد وابستگی‌ها و تیم‌های زیادی که روی آن کار می‌کنند.

باید ظاهری آشنا داشته باشد. اندکی شبیه C. برنامه‌نویسانی که در گوگل کار می‌کنند عموماً سال‌های اولیه‌ی اشتغال خود را سپری می‌کنند و بیشتر با زبان‌های رویه‌ای[۴] مانند خانواده‌ی C آشنا هستند.به دلیل این که نیاز است که برنامه‌نویسان بتوانند سریعاً کار با زبان را شروع کرده و پربار باشند زبان نمی‌تواند خیلی رادیکال باشد.

باید جدید باشد. C ، C++ و برخی قسمت‌های Java بسیار قدیمی هستند و طراحی آن‌ها پیش از ظهور

[3] Haskell

[4] procedural

ماشین‌های چندهسته‌ای، شبکه و توسعه‌ی برنامه‌های تحت وب است. خصوصیاتی در دنیای مدرن هست که با رویکردهای جدیدتر بهتر می‌توان به آن‌ها پرداخت. مانند همروندی داخلی.

اگر بخواهیم تمام اهداف ایجاد این زبان را تقسیم‌بندی کنیم:

کارایی زبان‌های کامپایل‌شده دارای سامانه‌ی ایستا،

آسانی برنامه‌نویسی زبان‌های پویا،

امنیت گونه‌ها، و امنیت حافظه،

پشتیبانی برای همروندی‌ها و ارتباط،

زباله‌جمع‌کن کارا و بدون تاخیر و

زمان کامپایل کوتاه.

۲.۱ برنامه‌نویسی با Go

خیلی‌ها فکر می‌کنند بزرگ‌ترین نقطه‌ی قوت Go امکانات همروندی آن است، در حالی که از نظر خود طراحان زبان، پیاده‌سازی منحصربه‌فرد رابط‌ها یکی از بهترین دستاوردهای آنان است. رابط‌ها در زبان‌های دیگری هم حضور دارند، اما هیچ‌کدام مانند Go پیاده‌سازی نشده‌اند. رابط‌های Go شیوه‌ی duck-typing را طوری به این زبان مترجمی و استاتیک وارد کرده‌اند که این قابلیت به ستون طراحی APIها در این زبان بدل گشته است.

Go یک زبان رویه‌ای مدرن است. امکاناتی برای شما مهیا است که بتوانید بدون دردسرهای موجود در زبانی مثل C، با توابع معمولی کدنویسی کنید. بدون پیچیدگی‌های موجود در Java و C++ کدهای شی‌گرا طراحی کنید. و بدون درگیر شدن با دستورات نحوی غریبه‌ی زبان‌های تابع‌گرا، به بسیاری از قابلیت‌های موجود در این زبان‌ها دسترسی داشته باشید.

یکی از روش‌های کاهش مشکلات یک محصول، حذف قسمت‌هایی است که تجربه ثابت کرده باعث بروز بیش‌ترین مشکلات خواهند شد. در همین راستا خیلی از قابلیت‌هایی که در زبان‌های دیگر مشکل‌زا بودند و یا به درستی پیاده‌سازی نشده بودند، کاملا از Go حذف شده اند. Go معتقد است نبود خیلی از قابلیت‌ها، بهتر از بودشان است. موارد دیگری هم هستند که شاید خیلی به چشم نیایند، اما در عمل بسیار مهم‌اند. مثلاً در صورت وجود مشکلی در زمان کامپایل، مترجم پیغام‌های خطایی را تولید می‌کند که برنامه‌نویس را واقعاً

متوجه مشکل می‌کند، و مانند خیلی از زبان‌ها خطوط درهم و برهمی را چاپ نمی‌کند که فهمیدن آن خودش بخشی از زمان برنامه‌نویس را هدر می‌دهد. یا فرضاً مستندات زبان، که واقعا در بین تمام زبان‌های برنامه‌نویسی منحصربه‌فرد است. مستندات Go شیوه‌های پویایی را برای نمایش و تعامل بهتر با خوانندگانش ابداع کرده‌اند که کم‌کم در حال ورود به بقیه‌ی زبان‌ها نیز می‌باشد. یا جامعه‌ی کاربری Go که به دلیل ذات سیستمی این زبان، علاوه بر برنامه‌نویسان خوش ذوق سطح بالا، شامل برنامه‌نویسان محتاط سطح پایین هم می‌شود و به شما اطمینان می‌دهد که از این اجتماع، فقط ابزارهایی بیرون خواهد آمد که بتواند نظر هر دو گروه را جلب کرده باشد و ده‌ها مدل از این موارد کوچک که بدون این‌که زیاد متوجه آن باشید در حال بالابردن بهره‌وری شما می‌باشند.

Go مشخصاً بر پایه‌ی سنت‌های C بنا شده است ولی تغییراتی هم برای افزایش اختصار،سادگی و امنیت داشته است. Go شامل این قسمت‌هاست:

۱. یک نحو[۵] و یک محیط که الگوهایش بسیار مشابه زبان‌های دینامیک است.

۱.۱. شیوه‌ی اختصاری اعلام و مقداردهی اولیه‌ی متغیرها توسط گونه‌ی استنتاجی:

استفاده از $x := 0$ به جای $\text{var x} = 0;$ یا $\text{int x} = 0;$

۲.۱. سرعت بالای تولید

۳.۱. مدیریت بسته‌ها از راه دور (go get) و مستندسازی بسته‌ها به صورت برخط.

۲. رویکردهای متمایز در مقابل مشکلات خاص:

۱.۲. ابتکارات داخلی در مورد همزمانی‌ها: روند سبک[۶] (goroutines)، کانال‌ها و حکم select.

۲.۲. یک سیستم رابطی به جای ارث‌بری مجازی و یک قرار دادن یک گونه ب جای ارث‌بری غیرمجازی.

۳.۲. یک زنجیره‌ی دستوری که به طور پیش‌فرض لینک‌های استاتیک محلی باینری بدون وابستگی بیرونی تولید می‌کند.

۳. میل به ساده نگه داشتن توضیحات؛ به قدری ساده که در ذهن برنامه‌نویس بماند با حذف قسمت‌هایی از توانایی‌های زبان‌های مشابه.

---

[5] syntax

[۶] light-weight process(LWP) به معنای دست یافتن به چندوظیفه‌ای است

فصل ۲

خصوصیات زبان

۱.۲ ویژگی‌ها

زبانی از طرف برنامه‌نویسان، برای برنامه‌نویسان

تاریخ برنامه‌نویسی اثبات کرده زبان‌هایی که از طرف برنامه‌نویسان طراحی شده‌اند - برنامه‌نویسانی که روی پروژه‌ها و سیستم‌های واقعی مشغول کار بودند - بسیار کاربردی‌تر و محبوب‌تر از زبان‌هایی هستند که از طرف محققین رایانهای و با نگرشی ایده‌آل‌گرا به وجود آمدند. چه کسانی Go را خلق کردند؟ کن تامسون، راب پایک، و رابرت گریسمر. بدون وارد شدن به جزییات، بد نیست نگاهی به ویژگی‌ها و خدمات قبلی این افراد داشته باشیم:

خلق یونیکس[1]، زبان C، سیستم Plan9، کدینگ معروف UTF-8، توسعه‌ی عبارات با قاعده، مشارکت در طراحی Java HotSpot، مشارکت در طراحیV8 (کروم[2] و Node.js و ...)، یک جایزه تورینگ ، یک مدال ملی فناوری ایالات متحده.

[1] Unix

[2] Chrome

سادگی و سرعت یادگیری

یکی از ویژگی‌های اساسی و منحصربه‌فرد Go، سادگی بسیار زیاد زبان و سرعت یادگیری بالای این زبان توسط تازه‌کاران است. این یکی از اهداف اولیه‌ی طراحی توسط طراحان این زبان برنامه‌نویسی بوده و دلیلش هم این واقعیت بود که همیشه امکان خارج شدن عده‌ای از تیم اصلی و اضافه شدن افراد جدید به تیم‌های نرم‌افزاری وجود دارد و زبان جدید باید آن قدر ساده باشد که افراد جدید به راحتی آن را فرا گیرند و به تیم اضافه شوند و همزمان آنقدر ساده باشد که افراد دیگر کد افراد جدید را به سادگی درک کنند و یک برنامه‌نویس تازه‌کار بتواند در کنار یک برنامه‌نویس حرفه‌ای کار کند.
زبان Go تنها ۲۵ کلمه‌ی کلیدی دارد.

کتابخانه‌ی قدرتمند زبان

تقریبا هیچ زبانی نمی‌توانید پیدا کنید که کتابخانه‌ای به قدرت کتابخانه‌ی همراه با زبان برنامه‌نویسی Go ارایه کرده باشد. این کتابخانه‌ها توسط بهترین برنامه‌نویسان جهان که تجربه‌ی کارهای بسیار بزرگ در مورد آن بخش از کتابخانه داشته‌اند طراحی شده است. به همین دلیل در اکثر اوقات برای ساختن یک برنامه‌ی تازه نیاز به استفاده از کتابخانه‌های بیرونی نخواهید داشت. از بسته‌هایی برای ارتباط با سیستم‌عامل تا اس‌کیوال تا برنامه‌نویسی شبکه و کار با جی‌سان[۳] و حتا یک سرور تحت وب قدرتمند داخلی، همه و همه بدون نصب هیچ ابزار اضافی همراه زبان در دستان شما برای ساخت برنامه‌هایتان خواهد بود.

استاتیک

زبان‌های برنامه نویسی استاتیک، زبان‌هایی هستند که در زمان کامپایل نوع داده‌ها باید مشخص باشد. Go یک زبان استاتیک است و وجود این ویژگی در زبان باعث شده بسیاری از مشکلات کدها قبل از زمان اجرا، یعنی در زمان کامپایل تشخیص داده شود و این یک مزیت بزرگ نسبت به زبان‌هاییست که داینامیک هستند.

[3] Json

ابزارهای خط فرمان

Go به همراه یک بسته‌ی کامل از ابزارهای خط فرمان عرضه می‌شود. ابزارهای تصحیح کد، کامپایل و نصب برنامه‌ها، مدیریت بسته‌ها برای استفاده از ابزارها، کدهایی که دیگران نوشته‌اند و سایر ابزارهای همراه بسته‌ی نرم‌افزاری به برنامه‌نویسان در توسعه‌ی سریع‌تر، حرفه‌ای‌تر و ساده‌تر کدها کمک می‌کند.

مدیریت بسته‌ها

Go شما را مجبور می‌کند که از ساختار فایل‌ها و بسته‌های مختص خودش استفاده کنید. این کار باعث می‌شود همه‌ی پروژه‌ها ساختار یکسانی داشته باشند و برنامه‌نویسان پروژه‌های مختلف به راحتی با پروژه و ساختار دایرکتوری‌ها و فایل‌های آن ارتباط برقرار کنند.

توسعه‌ی مداوم

تیم توسعه‌ی Go متشکل از جمعی از بهترین برنامه‌نویسان حال حاضر دنیاست و به صورت روزانه تغییرات، بهبودها و بحث‌های زیادی در مورد مشکلات و انتشار نسخه‌های آتی زبان صورت می‌گیرد.

سرعت کامپایل بالا

یکی از ویژگی‌های برتر Go نسبت به بسیاری از زبان‌های کامپایلری از جمله C++ و C، سرعت بسیار بالای کامپایل کدهای آن است. این تفاوت آن قدر نسبت به زبان‌های بیان‌شده زیاد است که می‌توان کدهای زبان Go را با دستور go run کامپایل و اجرا کرد و این عمل طوری دیده می‌شود که گویی یک فرمان از یک زبان خطی[۴] را اجرا کرده‌اید.

ابزار تست درونی

نوشتن تست واحد و تست رفتار در بین برنامه‌نویسان، این روزها امری متداول و در بسیاری از تیم‌ها اجباریست. زبان‌ها و چارچوب‌های حال حاضر عمدتاً از بسته‌های نرم‌افزاری جدا از زبان برای تست‌ها استفاده می‌کنند.

[4] script

Go در کتابخانه‌ی استانداردش، یک بسته برای نوشتن تست‌ها دارد و با ابزار go test هم این تست‌ها را می‌توان اجرا کرد.

همروند و موازی

Go ذاتاً یک زبان همروند است. زبان همروند زبانی است که طراحی آن به طور مشخص حول محور همروند بودن شکل گرفته است.

در حال حاضر زبان‌های همروند معروف کدام‌اند؟ ارلنگ[5]، هسکل، Go و...

با این‌که سیستم JVM در حال حاضر نمی‌تواند از نظر همروندی امکاناتی را در سطح سه زبان نام برده‌شده ارایه دهد، اما نمی‌توان اسم کلوژور[6] و اسکالا[7] را در لیست بالا قرار نداد. با نگاهی به این لیست می‌توان متوجه شد که Go تنها زبان غیرتابع‌گرا در این لیست است. (البته اسکالا هم فقط تابع‌گرا نیست) یکی از مهم‌ترین عوامل جذب زبان‌های تابع‌گرا امکانات همروندی آن‌ها بود. اما با معرفی Go دیگر نیازی به استفاده از زبان‌های تابع‌گرا - زبان‌هایی که سینتکس و مدل برنامه‌نویسی در آن‌ها ممکن است برای خیلی‌ها خوشایند نباشد - برای طراحی سیستم‌های همروند نیست. وقتی برنامه‌ها را به صورت همروند توسعه داده شوند، علاوه بر این‌که طراحی برنامه بهتر و قابل درک‌تر خواهد شد، می‌توان اطمینان داشت که در صورت وجود امکانات سخت‌افزاری مناسب، برنامه می‌تواند به طور موازی اجرا شده و سرعت بالاتری را به ارمغان بیاورد.

۲.۲ برنامه‌هایی که از Go استفاده می‌کنند

در نوامبر سال ۲۰۰۹، اولین نسخه‌ی آزمایشی زبان با پشتیبانی گوگل به صورت متن‌باز به عموم برنامه‌نویسان عرضه شد. از آن زمان تا کنون بیش از ۳۰۰ نفر از برنامه‌نویسان داوطلب در توسعه این پروژه شرکت داشته‌اند. از اولین ماه انتشار این زبان، شرکت‌های نوپای زیادی استفاده از آن را شروع کردند. با اینکه زبان در نسخه آزمایشی به سر می‌برد، اما برای خیلی‌ها وجود نام کن تامپسون و راب پایک کافی بود تا از کیفیت زبان اطمینان حاصل کنند.

[5] Erlang
[6] Closure
[7] Scala

اما شرکت‌های بزرگ‌تر، منتظر نسخه‌ی پایدار ماندند. تا این که در ماه مارس سال ۲۰۱۲، نسخه‌ی ۱.۰ از زبان برنامه نویسی Go به صورت پایدار منتشر شد.

در زیر لیست تعدادی از شرکت‌های استفاده کننده از این زبان را می‌بینید:

گوگل؛

یوتیوب[8]: تیم یوتیوب برنامه‌ی متن‌باز زیرساخت مقیاس‌پذیر مای‌اس‌کیوال را با این زبان منتشر کرده است.

بی‌بی‌سی[9]؛

نتفلیکس[10]: برای دو قسمت از معماری سرور خود از Go استفاده می‌کند.

کنونیکال[11]: زیرساخت back-end خود را با این زبان توسعه می‌دهند.

شبکه‌ی نوکیا زیمنس[12]: آن‌ها از Go برای خودکار کردن تایید آزمایش سخت‌افزار باندپهن[13] و برد RF استفاده می‌کند.

بیتلی[14]؛

هیرکو[15]: انبار داده‌های توزیع‌شده توسط Go پیاده‌سازی شده است.

کلاودفلیر[16]: یکی از نرم‌افزارهای خود را با Go ساخته‌اند.

اسماگ‌ماگ[17]؛

فیدبوکس[18]: از Go و mgo استفاده می‌کند تا بتواند روزانه بیش از یک میلیون جلد کتاب را عرضه کند.

آیرون.آی‌او[19]: یکی از برنامه‌های خود برای مقیاس‌پذیر صف کارها و خدمات دیگرش را با Go پیاده‌سازی کند.

موو وب[20]؛ تمام نرم‌افزارهای داخلی‌اش را با Go دوباره‌نویسی کرده است.

---

[8] Youtube
[9] BBC
[10] Netflix
[11] Canonical
[12] Nokia Siemens
[13] baseband
[14] Bitly
[15] Heroku
[16] CloudFlare
[17] SmugMug
[18] Feedbooks
[19] Iron.io
[20] Moovweb

ایربریک[۲۱]: از روبی به Go مهاجرت کرد.

سویرل.یواس[۲۲]؛

ساوند کلاود[۲۳]؛

تقریباً در تمام اسامی بالا، از Go برای طراحی سیستم‌های back-end استفاده شده است. جایی که زبان‌های خطی قدرت مناسب را ندارند و معمولا از JVM استفاده می‌شود؛ در خیلی از مواقع هم برای رسیدن به سرعت بالاتر، پای کدهای C و C++ به میان می‌آید.

همچنین برخی از برنامه‌های متن‌باز قابل توجه در زبان Go به شرح زیرند:

کاکروچ‌دی‌دبی[۲۴]: یک پایگاه داده متن‌باز، قابل انعطاف و سازگار است.

سینک‌تینگ[۲۵]: یک نرم افزار کلاینت - سرور برای هماهنگ‌سازی فایل متن‌باز است.

اسنپی[۲۶]: یک مدیریت بسته برای اوبونتو تاچ[۲۷] است که توسط کنونیکال توسعه یافته است.

اپن‌شیفت[۲۸]: یک پلتفرم محاسبات ابری.

همچنین برخی از چارچوب‌های متن‌باز قابل توجه در زبان Go عبارت‌اند از:

گوریلا[۲۹]: یک ابزار وب برای Go.

مارتینی[۳۰]: بسته‌ی کاربردی - خدماتی برای برنامه‌های وب.

بیگو[۳۱]: یک چارچوب وب با کارایی بالا در Go، برای برنامه های وب و سرویس‌های back-end مورد استفاده قرار می‌گیرد.

با وجود آن که چندسالی بیشتر از عمر این زبان نمی‌گذرد اما لیست شرکت‌ها و برنامه‌هایی که از Go استفاده می‌کنند همین طور ادامه دارد و این نشان می‌دهند این زبان توانسته مخاطبان را به خودش جذب کند.

[21] Air Brake
[22] swirl.us
[23] SoundCloud
[24] CockroachDB
[25] Syncthing
[26] Snappy
[27] Ubuntu Touch
[28] OpenShift
[29] Gorilla
[30] Martini
[31] Beego

۳.۲ رتبه‌بندی‌ها

از زمان عرضه‌ی زبان Go چیزی بیش از ۷ سال می‌گذرد. در این سال‌ها این زبان رشد خوبی در رتبه‌بندی‌ها داشته که به چند مورد اشاره خواهد شد:

PYPL

یکی از معیارها برای رتبه‌بندی زبان‌ها معیار PYPL است به معنی شهرت زبان برنامه‌نویسی[۳۲] است.
آن‌ها معتقدند میزان جست‌وجو برای آموزش زبان معیار بسیار خوبی برای رتبه‌بندی زبان‌ها بر اساس شهرت است و می‌تواند به افراد کمک کند تا زبان مناسب برای یادگیری یا استفاده در پروژه‌هایشان را انتخاب کنند.
این معیار به صورت سالانه منتشر می‌شود.
شکل روبه‌رو مربوط به جولای ۲۰۱۷ است.
همان طور که در تصویر هم دیده می‌شود زبان Go با دو رتبه صعود در جایگاه ۱۸ام قرار دارد.

Worldwide, Jul 2017 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Java	22.6 %	-1.1 %
2		Python	16.4 %	+4.0 %
3		PHP	9.1 %	-1.0 %
4		C#	8.2 %	-0.6 %
5		Javascript	8.0 %	+0.6 %
6		C++	6.6 %	-0.3 %
7		C	6.5 %	-0.3 %
8	↑	R	3.7 %	+0.5 %
9	↓	Objective-C	3.6 %	-1.1 %
10		Swift	2.8 %	-0.3 %
11		Matlab	2.5 %	-0.1 %
12		Ruby	1.8 %	-0.5 %
13	↑	VBA	1.4 %	-0.1 %
14	↓	Visual Basic	1.3 %	-0.3 %
15	↑	Scala	1.2 %	+0.1 %
16	↑	TypeScript	1.2 %	+0.5 %
17	↓↓	Perl	0.8 %	-0.3 %
18	↑↑	Go	0.5 %	+0.1 %
19	↓	lua	0.4 %	-0.1 %
20	↑↑↑	Kotlin	0.4 %	+0.2 %
21	↓↓	Delphi	0.3 %	-0.1 %
22		Rust	0.3 %	+0.0 %
23	↓↓	Haskell	0.3 %	+0.0 %

© Pierre Carbonnelle, 2016

شکل ۱.۲: شاخص PYPL

[32] PopularitY of Program Language

TIOBE

معیار دیگر معیار TIOBE است. که ماهانه منتشر می‌شود.

این معیار بر اساس جست‌وجوی نام زبان در موتورهای جست‌وجو است و ماهانه منتشر می‌کند. البته باید توجه داشت تا آگوست ۲۰۱۶ معیار TIOBE نام زبان Go را محدود به استفاده از نام گوگل کرده بود و تمامی نتایج برای جست‌وجوی زبان برنامه‌نویسی Go گوگل است. اما در آگوست ۲۰۱۶ این محدودیت برداشته شد و بر اساس اطلاعیه‌ی سایت خود معیار با این حذف زبان Go از رتبه‌ی ۵۵ به رتبه‌ی ۲۰ ارتقا یافت.

در شکل روبه‌رو رتبه‌ی زبان‌ها در ژوئن ۲۰۱۷ با همان ماه در سال قبل مقایسه شده. که در آن Go در رتبه‌ی ۱۵ام است.

Jun 2017	Jun 2016	Change	Programming Language	Ratings	Change
1	1		Java	14.493%	-6.30%
2	2		C	6.848%	-5.53%
3	3		C++	5.723%	-0.48%
4	4		Python	4.333%	+0.43%
5	5		C#	3.530%	-0.26%
6	9	⌃	Visual Basic .NET	3.111%	+0.76%
7	7		JavaScript	3.025%	+0.44%
8	6	⌄	PHP	2.774%	-0.45%
9	8	⌄	Perl	2.309%	-0.09%
10	12	⌃	Assembly language	2.252%	+0.13%
11	10	⌄	Ruby	2.222%	-0.11%
12	14	⌃	Swift	2.209%	+0.38%
13	13		Delphi/Object Pascal	2.158%	+0.22%
14	16	⌃	R	2.150%	+0.61%
15	48	⌃⌃	Go	2.044%	+1.83%
16	11	⌄⌄	Visual Basic	2.011%	-0.24%
17	17		MATLAB	1.996%	+0.55%
18	15	⌄	Objective-C	1.957%	+0.25%
19	22	⌃	Scratch	1.710%	+0.76%
20	18	⌄	PL/SQL	1.566%	+0.22%

شکل ۲.۲: شاخص TIOBE

Redmonk

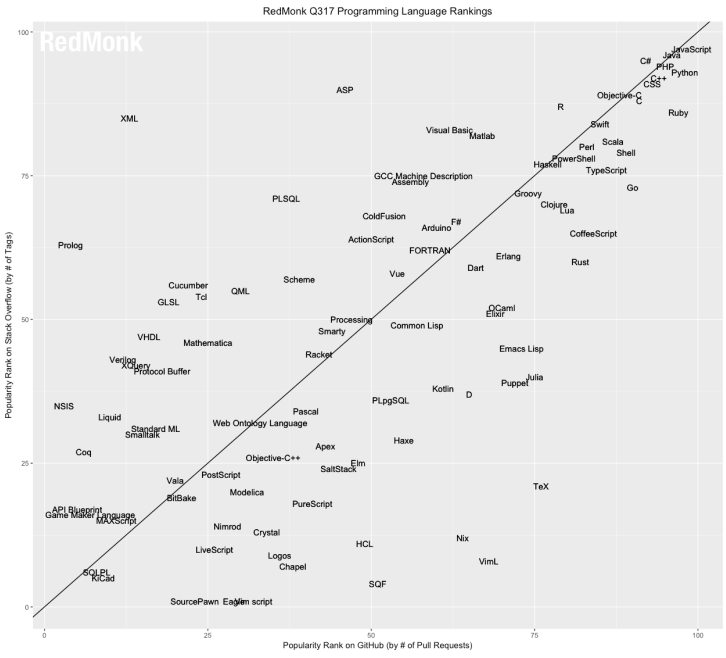
معیار Redmonk هر ۶ ماه منتشر می‌شود.

این معیار بر اساس خط‌های خام کد در گیت‌هاب[۳۳] و برچسب‌های زبانی در استک اورفلو[۳۴] است.

در این رتبه‌بندی Go در جایگاه ۱۳ام است.

[۳۳] GitHub؛ یک سرویس میزبانی وب برای پروژه‌های برنامه‌نویسان در سراسر دنیا.

[۳۴] StackOverflow؛ یک سایت پرسش و پاسخ درباره‌ی حوزه‌ی وسیعی از مباحث برنامه‌نویسی و زبان‌های مختلف برنامه‌نویسی.



شکل ۳.۲: شاخص Redmonk

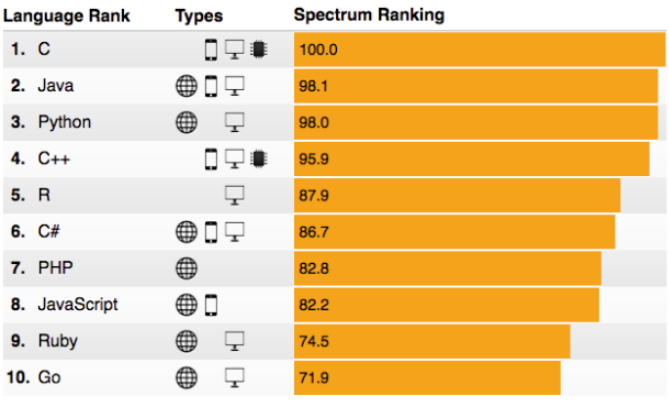
IEEE Spectrum

معیار IEEE Spectrum سالانه اعلام می‌شود.

رتبه‌بندی در این معیار بر اساس ۱۰ منبع انجام می‌شود:

جست‌وجوی نام برنامه در گوگل، گرایش‌ها در سایت‌های گوگل، توییتر[۳۵]، گیت‌هاب، استک اورفلو، ردیت[۳۶]، هکر نیوز[۳۷]، کریر بیلدر[۳۸]، دایس[۳۹] و کتابخانه‌ی دیجیتال IEEE XPlore.

Go در این معیار ۱۰ام است.



شکل ۴.۲: شاخص IEEE Spectrum

Eng Language

معیار Eng Language توسعه‌یافته‌ی PYPL است و برای یافتن اسم زبان‌ها از سیستم متفاوتی استفاده می‌کند. اما معیارش همان جست‌وجوی آموزش زبان است.

تفاوت محسوس در این معیار قرار گرفتن زبان Go در جایگاه ۱۸ام است.

---

[۳۵] Twitter؛ یک شبکه‌ی اجتماعی و سرویس ارایه‌دهنده‌ی میکروبلاگ است که به کاربران اجازه می‌دهد تا ۱۴۰ حرف، پیام متنی را که توییت نامیده می‌شود، ارسال کنند.

[۳۶] Reddit؛ یک سایت برای جمع‌آوری اخبار اجتماعی که در آن کاربران ثبت‌نام کرده قادرند اخبار را در قالب لینک یا متن ارسال کنند و آن را با دیگران به اشتراک بگذارند.

[۳۷] Hacker News؛ سایت خبری با تمرکز بر روی اخبار علوم کامپیوتر و برنامه‌های نوپا.

[۳۸] CareerBuilder؛ سایت اشتغال برخط.

[۳۹] Dice؛ سایت شغلی با تمرکز بر تکنولوژی و علوم مهندسی.

فصل ۳

مقایسه‌ی زبان

۱.۳ نواقص زبان Go

نقطه ضعف‌های Go از دو زاویه قابل بررسی است. در زاویه‌ی اول نقطه ضعف‌هایی هستند که سازندگان زبان کاملا به آن‌ها آگاهی داشته و قبول دارند که باید بیشتر روی آن‌ها کار شود. در زاویه‌ی دوم نقطه ضعف‌هایی هستند که در جامعه‌ی کاربران Go به عنوان نقاط قوت به آن نگریسته می‌شود! اما برنامه‌نویسانی که قصد مهاجرت از زبان‌های دیگر را دارند، ممکن است با آن‌ها مشکل پیدا کنند.

نقص اول تفاوت سرعت Go با زبان C است! Go به علت داشتن سیستم زمان اجرا و سیستم زباله‌جمع‌کن حداقل به صورت تئوری هیچ وقت توانایی برابری با C را نخواهد داشت. این که چرا C تا این اندازه سریع است، خود نیاز به بحث جداگانه‌ای دارد.

در حال حاضر کامپایلر Go فقط قادر است کدهای مناسب تولید کند، نه کدهای سریع! به این معنی که کامپایلر Go در نسخه فعلی بهینه‌سازی خاصی را روی کدهای نهایی انجام نمی‌دهد. البته که تولید کد ماشین روی چندین پلتفرم مختلف برای یک زبان Native مانند Go اصلاً کار راحتی نیست. در این زمینه باید کمی صبر کرد و فرصت بیشتری به توسعه دهنده‌های زبان داد.

Go در حال حاضر هم یک زبان سریع محسوب می‌شود. اگر بخواهید Go را با زبان‌هایی مانند پایتون یا روبی[1]

[1] Ruby

یا پی‌اچ‌پی[2] و یا امثال آن‌ها مقایسه کنید، تجربه ثابت کرده است که Go بین ۲۰ تا ۵۰ برابر سریع تر عمل می‌کند.

البته موارد زیادی وجود دارد که دیده شده Go به نسبت این زبان‌ها کندتر عمل کند. در چنین حالتی، مطمئن باشید که آن عملیات در پشت صحنه توسط کدهای C اجرا شده است. در چنین زبان‌هایی برای حل مشکل سرعت، بسیار از ماژول‌ها را در C توسعه می‌دهند.

اگر بخواهیم Go را با زبان‌هایی مثل Java یا C مقایسه کنیم، نمی‌توانیم با قطعیت نظری را مطرح نماییم. مایکروسافت[3] و اوراکل (همان سان[4] سابق) میزان زیادی از وقت و میلیون‌ها دلار از سرمایه‌ی خود را برای توسعه‌ی ماشین مجازی این دو زبان صرف کرده‌اند.

حتی Java که همیشه به شوخی به عنوان یک زبان کند معرفی می‌شود، در واقع یکی ازسریع ترین پلتفرم‌های موجود در دنیای برنامه‌نویسی است. در خیلی از موارد Go سریع‌تر از این دو پلتفرم بوده، و در خیلی از موارد هم کندتر از آن ها. واقعاً نمی‌توان به صورت مطلق درباره سرعت آن‌ها ابراز نظر کرد. اما با جرئت می‌توان گفت که از نظر میزان حافظه‌ی مصرفی، Go ثابت کرده که ممکن است حتا تا ده برابر بهینه‌تر از این پلتفرم‌ها عمل کند.

مطلب مهم این است که Go به شدت در حال توسعه است و تقریبا هر روز بهینه‌سازی‌های زیادی در آن اعمال می‌شود. حالا که Go به وضعیت ثبات رسیده است، تازه تیم توسعه کار خود را در زمینه بهینه‌سازی زبان شروع کرده است. برای مثال تنها در یکی از به‌روزرسانی‌ها کدهای مربوط به همروندی‌ها تا ۵۰ درصد عملکرد بهتری داشته اند.

از طرفی، سرعت یک معیار مطلق نیست. خیلی از مسائل ممکن است بر سرعت برنامه تاثیرگذار باشد، که یکی از مهم‌ترین آن‌ها خود برنامه‌نویس و روش‌ها و الگوریتم‌هایی است که انتخاب می‌کند. Go زبانی است با سرعتی بسیار مناسب که روزبه‌روز نیز در حال پیشرفت در این زمینه است؛ فقط باید کمی صبر داشت و به تیم Go فرصت داد تا کارشان را انجام دهند.

نقص دیگر، سیستم زباله‌جمع‌کن است. Go در ابتدا برای معماری x64 طراحی شده بود (۶۴ بیت). دلیل آن هم این بود که چون پلتفرم‌های ۳۲ بیتی کم‌کم در حال جایگزین شدن با پلتفرم‌های ۶۴ بیتی هستند، تیم

[2] PHP
[3] Microsoft
[4] SUN

توسعه تمام توجه خود را به پلتفرم‌های ۶۴ بیتی معطوف کرده بود.

متأسفانه در حال حاضر سیستم زباله‌جمع‌کن روی معماری ۳۲ بیت (x86) با یک نقص فنی روبه‌رو است که در موارد معدودی می‌تواند باعث کرش کردن برنامه شود. البته این مشکل فقط برای معماری x86 است و اگر برنامه‌ها را روی معماری x64 اجرا شود مشکلی رخ نخواهد داد. از همین رو توصیه شده است تا زمان برطرف کردن این مشکل، برنامه‌ها فقط روی سیستم‌های ۶۴ بیتی اجرا شوند.

نکته‌ی بعدی این است که در حال حاضر سیستم زمان اجرا روی پلتفرم‌های ۶۴ بیتی، فقط توانایی استفاده از ۱۶ گیگابایت حافظه دارد (به طور متوالی). خیلی نادرند برنامه‌هایی که به صورت پیوسته به ۱۶ گیگابایت حافظه نیاز داشته باشند، اما در صورت نیاز، باید برنامه را به قسمت‌های کوچکتری تقسیم کرد تا این محدودیت دور زده شود.

در مخزن توسعه Go، اصلاحیه‌هایی برای رفع مشکل GC در سیستم‌های ۳۲ بیتی وجود دارد، و میزان کسترش حافظه برای سیستم زمان اجرا نیز به ۱۲۸ گیگابایت افزایش داده شده است.

مشکل دیگری که در مورد این زبان وجود دارد نام این زبان است. اگر نام Go را در اینترنت جست‌وجو کنید شاید به نتیجه‌ی دلخواه خود نرسید. به همین دلیل طبق قانونی که بین کاربران وجود دارد این زبان را نام Golang صدا می‌کنند، و تلفظ آن نیز مانند ارلنگ است.

نقص بعدی در رابطه با یادگیری Go است. Go یک زبان سیستمی است. در این حالت با همان وضعیتی روبه‌روایم که در زبان C نیز با آن روبه‌رو خواهیم بود. یعنی برنامه‌نویسی با Go به سطح قابل قبولی از تجربه و تخصص در علوم کامپیوتری نیاز دارد.

در زبان‌هایی مثل Go یا C، تا زمانی که کاملاً با طرز کار سی‌پی‌یو، حافظه، ورودی‌خروجی‌ها و مباحث سطح پایین مربوط به هر کدام آشنا نباشیم، نمی‌توانیم به راحتی با زبان ارتباط برقرار کنید. شاید بتوان به سادگی زبان آن را یاد گرفت، اما آگاهی از زبان همیشه در حد پایینی قرار خواهد داشت. در حالی که یک برنامه‌نویس روبی ممکن است هیچ وقت نیاز نداشته باشد تا از ثبات‌ها سر در بیاورد.

خصوصاً برای استفاده از قابلیت‌های همروندی باید پیش‌زمینه‌ی مناسبی برخوردار بود. این طور نیست که به صرف برنامه‌نویسی در این زبان، اطمینان داشت که برنامه روی یک سی‌پی‌یو ۱۶ هسته‌ای ۱۶ برابر سریع‌تر اجرا خواهد شد! برنامه نویسی همروند نیازمند مطالعه‌ی زیاد و تجربه‌ی کافی است. Go برای شما معجزه نمی‌کند. مباحث دیگری هم هستند که به اجتماع کاربران Go مربوط می‌شوند. تفکراتی در این اجتماع وجود دارد که

ممکن است برای برنامه نویسانی که از زبان‌های دیگر می‌آیند کمی عجیب باشد.
برای مثال، تیم توسعه Go و اجتماع کاربری آن همیشه عنوان کرده‌اند که استفاده از کتابخانه‌های ساده، از اولیت بسیار بالاتری نسبت به چارچوب‌ها برخوردار است؛ و این که چارچوب‌ها یکی از عوامل اصلی در پیچیده شدن پروژه می‌باشند و هر چه کمتر از آن‌ها استفاده شود بهتر است! و یا این که Andrew Gerrand یکی از اعضای اصلی تیم توسعه در اکانت تویتر خود اعلام کرد چقدر از این که GO چارجوب‌هایی مثل Django یا Rails ندارد خوشحال است و امیدوار است که هیچ وقت هم نداشته باشد!
مسلماً برای خیلی از برنامه‌نویسان زبان‌های دیگر که به کار با چارچوب‌ها عادت کرده‌اند چنین نظراتی کمی عجیب و غریب است! در Go فلسفه‌ی Unix حکم‌فرماست. جای تعجبی هم ندارد، چون این افراد خودشان خالق Unix هستند!
در فلسفه‌ی Unix باید از ابزارهای کوچکی استفاده کرد که هر کدام کار مشخصی انجام می‌دهند؛ در صورت نیاز می‌توان با کنار هم قرار دادن این قسمت‌های کوچک، ابزار بزرگ‌تری را ایجاد کرد. چارچوب‌ها به عنوان سیستمی که ابزارهای یکپارچه و غیرقابل تجزیه را توصیه می‌کنند، عملاً در جبهه‌ی مخالف فلسفه‌ی Unix قرار دارند.
خیلی از برنامه‌نویسان تازه‌وارد متوجه نمی‌شوند که چرا Go کار با Type و توابع را جایگزین کلاس‌ها و متدها کرده... چرا وراثت را از زبان حذف کرده... چرا این زبان استثنا ندارد... چرا از سربارگذاری توابع پشتیبانی نمی‌کند... و ده‌ها سوال دیگر...
اما اکثر آن‌ها بعد از مدتی کار با Go از این که زبان پیشینشان مانند Go پیاده‌سازی نشده بوده ابراز ناراحتی کرده‌اند! باید گفت که در Go باید به شیوه‌ی Go برنامه‌نویسی کرد.

۲.۳ مقایسه با C++ و C

در این قسمت زبان‌های C و Go از چندین منظر با هم مقایسه می‌شوند:
سیستم‌عامل‌ها با زبان C و C++ ساخته شده‌اند و بسیاری از کتابخانه‌ها رابط‌هایی برای C دارند؛ یعنی می‌توانند تابع‌هایC را با کمترین سربار صدا بزنند. در حالی که Go با این که اجازه‌ی صدا زدن توابع C را می‌دهد اما سربار زیادی در زمینه‌ی اجرا دارد.
C و C++ دارای مدیریت حافظه‌ی دستی هستند. در مقابل Go زباله‌جمع‌کن نسلی دارد که بالاخره زمانی

برای احیای حافظه می‌ایستد و مهندسان Go آن را برای وقفه‌های کوتاه بهینه‌سازی کرده‌اند.
کامپیوترهای ما هسته‌های زیادی دارند به همین علت برنامه‌نویسی موازی بسیار مهم است. لازم است تا زبان‌ها از برنامه‌نویسی چندهسته‌ای پشتیبانی کنند. در این زمینه Go دست بالا را دارد.
برای کارایی بالا یک زبان برنامه‌نویسی خوب باید دلیل خوبی برای کارایی‌اش داشته باشد. یعنی زمانی که به برنامه‌ای نگاه شود بتوان تخمین زد که چه مقدار سریع اجرا خواهد شد. این کار با گذر زمان سخت‌تر شده و بیشتر بر معیارهای سنجش تکیه کرد. با این حال در زبان‌های C و Go تشخیص این مسئله راحت‌تر است.
دهه‌ها گذشته ولی هنوز هیچ راه‌حل جهانی‌ای برای حل مشکلات وابستگی‌ها در C پیدا نشده. هیچ مکانیسم ساخت استانداردی هم برای آن وجود ندارد. ممکن است محیط توسعه‌ی یکپارچه بتواند کار با وابستگی‌ها را راحت‌تر کند اما ولی اگر چیزی داخل خود زبان باشد کار بسیار آسان‌تر خواهد بود. در زبان Go راه استانداردی وجود دارد تا بتوان برنامه‌ها را ساخت؛ آزمایش کرد و وابستگی‌ها را افزود.

۳.۳ مقایسه با Java

Go در مقابل Java؛ می‌توان گفت این دو حریف مناسبی برای یک دیگر نیستند. یکی از آن‌ها بزرگ و سنگین‌وزن است که صنعت را سال‌ها تحت سلطه داشته و دیگری تازه‌وارد سبک‌وزن ترسناکی است که جوانی و امید فراوانی را از خود نشان می‌دهد.
Go و Java هدف‌های متفاوتی هم دارند. یکی بخش سرور برنامه‌های تخت وب را نشانه رفته که دیگری زمانی گزینه‌ی ارشد آن بوده است. دیگری حالا انتخاب مشهوری برای وسایل است.
ولی همه Java را برای برنامه‌نویسی بخش سرور برنامه‌های تحت وب - همان محدوده‌ای که Go به آن حمله کرده و دارد پایگاه Java را از بین می‌برد. - ترک نکرده‌اند. و البته این تغییر جهش بزرگی نخواهد بود چون هر دو زبان در بسیاری جنبه‌ها شبیه به هم هستند.
هر دو ادای احترامی به C هستند؛ اگر نه در باطن حداقل در ظاهر - جایی که برنامه‌نویسان زندگیشان را صرف سروکله زدن با دستورات نحوی می‌کنند. هر دو سرراست و دستوری هستند و آن قدر شباهت ساختاری دارند که تبدیل یکی به دیگری کار سختی نباشد.

سابقه‌ی طولانی Java تأثیرات شبکه‌ای دارد که به همه کمک می‌کند

Java از سال ۱۹۹۵ مورد استفاده بوده است و هر ساله افراد زیادی را جذب کرده. از پردازنده‌های کوچک جاسازی‌شده تا چیپ‌های بزرگ سرورها Java سریع و کارا اجرا می‌کنند. آن هم به خاطر ماشین مجازی چابک و دقیق - سر - وقت آن. اندروید[5] به عنوان محبوب‌ترین پلتفرم تلفن همراه در دنیا مزیتی برای Java شده است. به همین دلایل هم Java همچنان در صدر رتبه‌بندی‌ها می‌ماند. این پذیرش وسیع به این معناست که کدهای بسیاری برای استفاده هست و بسیاریشان هم متن‌باز هستند تا زندگی را آسان کنند.

سابقه‌ی کم Go باعث مناسب بودنش می‌شود

امکان استفاده از کدهای قدیمی ممکن است به نظر یک هدیه باشد اما بیشتر مواقع به زحمتش نمی‌ارزد. Go تاریخچه‌ی کوتاهی دارد. یعنی برای استانداردهای امروزه‌ی وب ساخته شده. هیچ ایده‌ی فراموش‌شده‌ای در آن نیست. به طور ساده جدید و مهندسی‌شده است برای کسانی که امروزه مشغول ساخت وب هستند.

Java اجازه‌ی دست‌درازی به باقی زبان‌ها را می‌دهد

JVM بنیادی است بر پایه‌ی هزاران زبان هیجان‌انگیز که برای اجرا از Java استفاده می‌کنند. هر کدام را بتوان به راحتی به برنامه وصل کرد. Java این آزادی را می‌دهد که برای هر قسمت برنامه از زبان مخصوصش استفاده کرده و همچنان آن را در همان JVM اجرا کرد. نیازی به استفاده از آن نیست اما قابلیتی است که همواره پیش روی ماست.

Go هماهنگی را ترویج می‌کند

بله ممکن است شما آن قدر توانا باشید که بتوانید با استفاده از زبان‌های متفاوت و JVM چنین کدی را اجرا کنید اما چه کسی می‌تواند آن را بخواند یا اصلاح کند؟ استفاده از کتابخانه‌های زبان‌های متفاوت زمان طراحی یک کد خوب برنامه‌ی چندان مناسبی نیست. این امکانات ممکن است گاهی لازم باشند اما هیچ وقت به خوبی اجرایی نشده.

هماهنگی و ثبات زندگی را برای همه راحت‌تر می‌کنند. این آن چیزی است که Go ارایه می‌کند.

[5] Android

Java هر ساختی که خواسته شود دارد

در گذر سال‌ها جامعه‌ی Java قابلیت‌های بسیاری را از Java خواسته است. برخی از مواقع آرزویشان برآورده شده. قابلیت‌های زیادی به زبان افزوده شده. اگر ایده‌ی جدیدی در مورد زبان برنامه‌نویسی هست احتمال زیادی دارد کسی آن را در دنیای Java اعلام کرده باشد. ایدئال نیست اما هر ایده‌ای که برای نوشتن کد در ذهن دارید توسط مغزهای پشت Java امکان‌پذیر شده است.

Go از ساخت‌هایی که باعث گیجی شوند دوری می‌کند

ساخت Go ساده است. این سادگی ساخت است که باعث شده یک برنامه‌نویس خوب بتواند ضرف چند ساعت این زبان را یاد بگیرد. ایده‌های بسیار جذاب پشت هزاران سند مختلف نیست. این ممکن است زمان نوشتن کد محدودکننده باشد ولی هنگام خواندن کد دیگران آرامشبخش است. هر کسی از خصوصیات هسته‌ای کد یکسانی استفاده می‌کند. و این خصوصیتی است که برای کارایی به Go اضافه نشده!

Java بالغ شده است

گذر سال‌ها باعث بلوغ، خرد و ثبات می‌شود - همه‌ی این‌ها دلیل محکمی است برای انتخاب پایگاه کد وسیع و مهندسی‌شده که بیش از دو دهه عمق دارد. افراد هم اکنون نیز در ابتدای یادگیری علوم کامپیوتر با زبان Java آشنا می‌شوند. برترین پلتفرم -اندروید هم بر اساس Java نوشته شده. مگر زمانی که دلیل محکمی برای تغییر داشته باشید بهتر است با بهترین بمانید.

Go سابقه‌ی بدی در کارنامه‌اش ندارد

گاهی بهتر است گذشته را پشت سر گذاشت. هر چه باشد پیشرفت گاهی به معنی شروع تازه است. Go این امکان را می‌دهد تا با یک ابزار تمیز، محکم و تازه که برای کارهای امروزه بهینه شده است کار کنید. این امکان را می‌دهد تا از سادگی و آزادی پشت سر گذاشتن گذشته لذت برد.

و سادگی به این خاطر که گوگل ساخت Go را شروع کرد تا نوشتن کد برای سرورهای بی‌پایانش را ساده کند. البته این دلیل آن نمی‌شود بیش از نیازش رشد نکرده باشد. هم اکنون هم بسیاری از این زبان برای نوشتن برنامه برای ربات‌ها و ... استفاده می‌کنند.

فصل ۴

مثال‌هایی از زبان Go

در این قسمت به عنوان مثال ۲ برنامه‌ی نوشته‌شده با Go نشان داده می‌شوند:

۱.۴ برنامه‌ی اول

برنامه‌ی اول چاپ ده جمله‌ی اول دنباله‌ی فیبوناچی است.

این برنامه با یک تابع main یک تابع fibonacci و به صورت بازگشتی نوشته شده است.

آن‌چه ممکن است جلب توجه کند شیوه‌ی معرفی و مقدار اولیه دادن است که به صورت استاسیک انجام می‌شود.

لازم است گفته شود بسته‌ی fmt بسته‌ای شامل توابع ورودی‌خروجی است که مشابه دستورات printf و scanf زبان C هستند.

```
package main
import "fmt"
func fibonacci() func() int {
    first, second := 0,0
    return func() int{
        if(first == 0) {
         first, second = 1,1
        }else {
            first, second =second, first + second
        }
    return first
    }
}
func main() {
    f := fibonacci()
    for i := 0; i < 10; i++ {
        fmt.Print(f()," ")
    }
}
```

```
 1 1 2 3 5 8 13 21 34 55
```

شکل ۱.۴: مثال ۱: دنباله‌ی فیبوناچی

۲.۴ برنامه‌ی دوم

برنامه‌ی دوم مرتب کردن یک آرایه است که با
استفاده از Selection Sort انجام می‌شود.
این برنامه هم ۲ تابع دارد؛ یک تابع main و یک
تابع selectionsort که با استفاده از الگوریتم آن آرایه‌ی داده‌شده را مرتب می‌کند و سپس آن را قبل و بعد
از مرتب شدن چاپ می‌کند.

```go
package main
import (
        "fmt"
)
func selectionSort(array []int) {

        for i := 0; i < len(array) - 1; i++ {
                min := i
                for j := i + 1; j < len(array) - 1; j++ {
                        if array[j] < array[min] {
                                min = j
                        }
                }
                array[i],array[min] = array[min],array[i]
        }
}

func main() {

        array := []int{8, 6, 2, 5, 9, 0, 4, 3, 7, 1}
        fmt.Println("Unsorted array: ", array)
        selectionSort(array)
        fmt.Println("Sorted array: ", array)
}
```

```
Unsorted array:  [8 6 2 5 9 0 4 3 7 1]
Sorted array:  [0 2 3 4 5 6 7 8 9 1]
```

شکل ۲.۴: مثال ۲: مرتب کردن یک آرایه

نتیجه‌گیری

از کثرت و تنوع زبان‌های برنامه‌نویسی همگان با خبرند. و در انتخاب بهترین زبان برای برنامه‌نویسی همه دوبه‌شک.

انتخاب‌ها زیادند اما شاید بتوان پیش‌بینی کرد زبان Go با توجه به آن‌چه گفته شد از جمله زبان‌هایی باشد که باید برایش وقت گذاشت و با آن کار کرد.

شاید بتوان پیش‌بینی کرد با توجه به جایگاه زبان Go پس از گذشت حدود ۸ سال از عرضه‌اش؛ همان گونه که تا به این‌جا پیش آمده به پیش رفتن و شهرتش ادامه دهد.

واژه‌نامه‌ی فارسی - انگلیسی

استثنا . . . . . exception
برخط . . . . . online
تابع‌گرا . . . . . functional
ثبات . . . . . register
چارچوب . . . . . framework
چندوظیفه‌ای . . . . . multitasking
خط فرمان . . . . . command prompt
رابط . . . . . Interface
زمان اجرا . . . . . runtime
سربارگذاری . . . . . overloading
شرکت نوپا . . . . . StartUp Company
گرایش . . . . . trend
گونه‌ی استنتاجی . . . . . type inference
معیار سنجش . . . . . benchmarking
ورودی‌خروجی . . . . . I\O
همروند . . . . . concurrency

کتاب‌نامه

[۱] سایت زبان برنامه‌نویسی Go؛ https://golang.org/

[۲] سایت ویکی‌پدیا: https://www.wikipedia.org/

[۳] سایت علوم کامپیوتر معیار سنجش بازی‌ها: http://benchmarksgame.alioth.debian.org/

[۴] Donovan A.A., Kernighan Brian W. (2016) The Go Programming Language.Addison-Wesley

[۵] Doxsey Caleb. (2016) Introducing Go: Build Reliable, Scalable Programs. O'Reilly

[۶] Ivo Balbaert. (2012) The Way To Go: A Thorough Introduction To The Go Programming Language. iUniverse

[۷] Kenedy Wiliam, Ketelsen Brian, St. Martin Erik. (2016) Go in Action. Manning

[۸] Vivien Vladimir. (2016) Learning Go Programming. Packt

Abstract

There are many programming languages in the world and so many are still being produced every year.
So many forgotten and some still familiar to our ears.
In this project we look into one of the newest programming languages; Go by Google.
We learn about this language, it's differences from other languages and we see some simple examples of programming with Go language.

Keywords: Go - Programming Language

College of Science

School of Mathematics, Statistics, and Computer Science

A Brief Study Of Go Programming Language

Sarah Abrishami

Supervisor: Ebrahim Naghibzadeh Mashayekhi

A thesis submitted to Graduate Studies Office

in partial fulfillment of the requirements for the degree of

B.Sc. in

Computer Science

2017