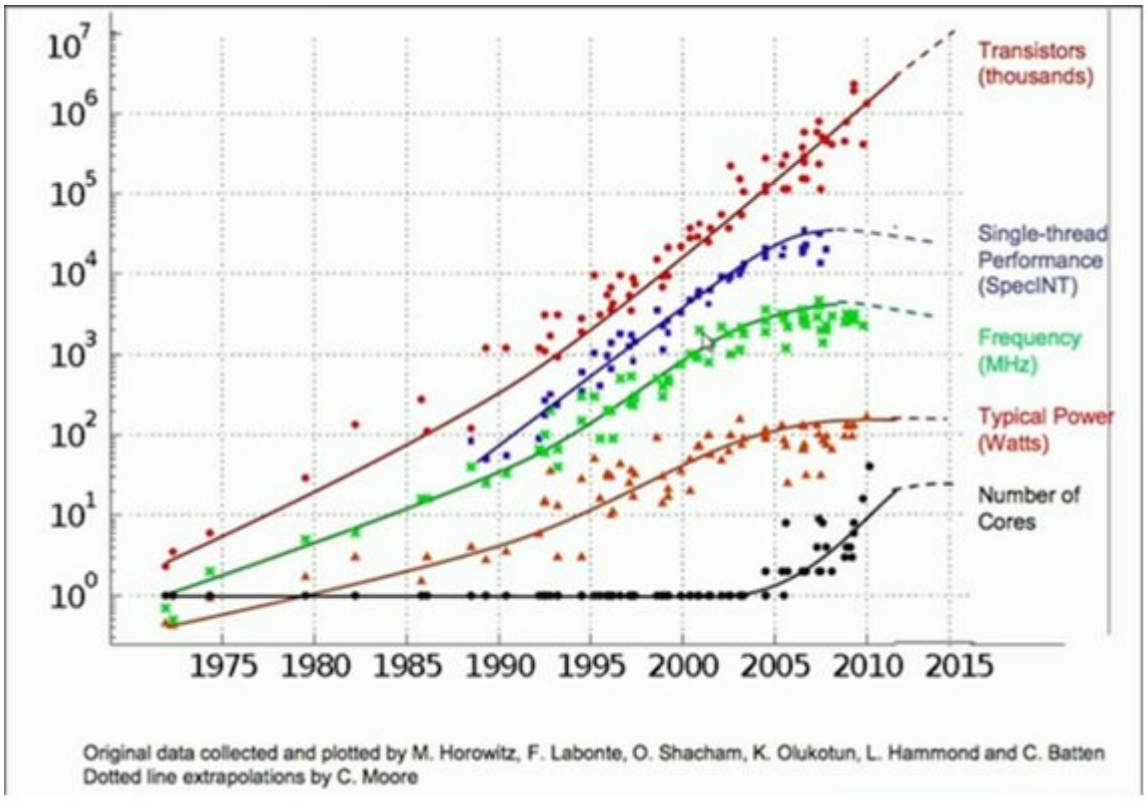
# ۷ دلیل قانع کننده برای یادگیری زبان برنامه‌نویسی Go


در سال‌های اخیر، زبان برنامه‌نویسی تقریبا جدیدی به نام Go یا همان GoLang توانسته است جایگاه ویژه‌ای میان توسعه‌دهندگان پیدا کند. برنامه‌نویسان مجرب همواره مشتاق هستند زبان جدیدی را یاد بگیرند، زیرا زبان‌های جدید قابلیت‌های کاربردی قدرتمندی متناسب با تغییرات دنیای فناوری در اختیار توسعه‌دهندگان قرار می‌دهند و برخی از مشکلات رایج را برطرف می‌کنند. مجله شبکه با استناد به دلایلی که ممکن است کمتر در مورد آن‌ها خوانده یا شنیده باشید به شما خواهد گفت چرا یادگیری زبان برنامه‌نویسی Go انتخاب درستی است.

## محدودیت‌های سخت‌افزاری

قانون مور در حال نزدیک شدن به روزهای پایانی حیات خود است. اولین پردازنده پنتیوم با فرکانس کاری 3.0 گیگاهرتز در سال 2004 میلادی از سوی اینتل منتشر شد. نزدیک به دو سال پیش اپل مک‌بوک‌های قدرتمندی با فرکانس کاری 2.9 گیگاهرتز عرضه کرد که این موضوع نشان می‌دهد در طول این سال‌ها توان پردازشی سرعت چشم‌گیری نداشته است. شکل زیر به خوبی نشان می‌دهد در طول این سال‌ها توان پردازشی تا چه اندازه پیشرفت داشته است.

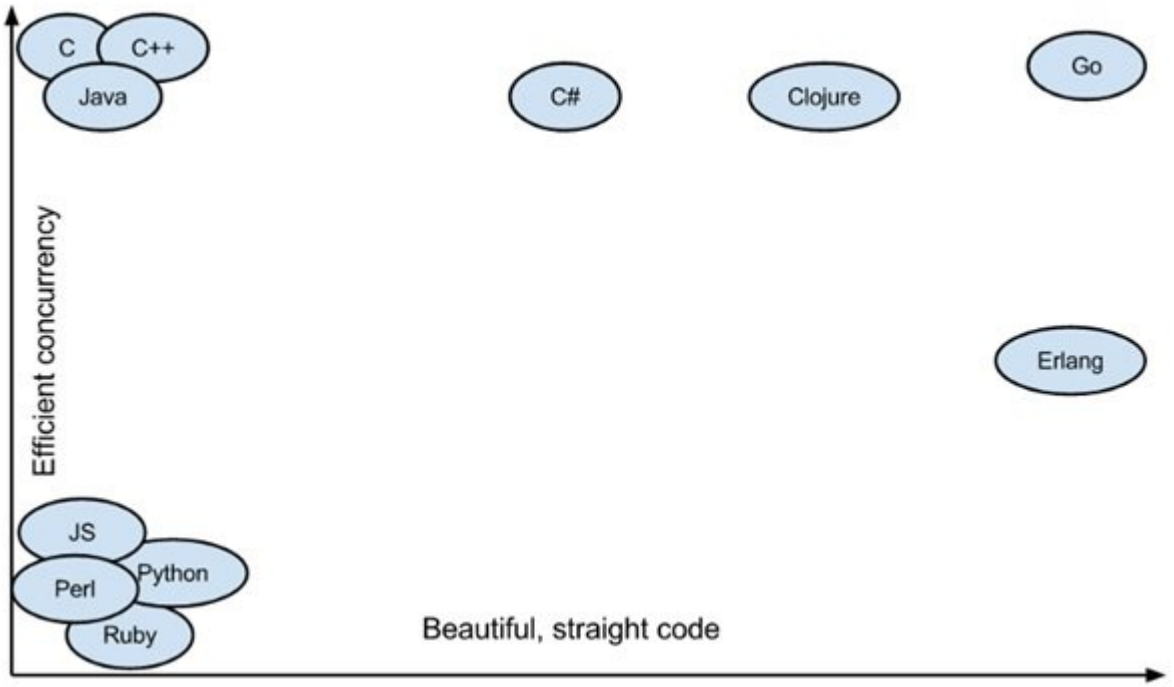



نوار آبی رنگ این نمودار به وضوح نشان می‌دهد عملکرد تک ریسمانی و فرکانس پردازنده‌ها در یک دهه گذشته تغییر قابل محسوسی نداشته و تقریبا ثابت بوده است. اگر تصور می‌کنید افزایش تعداد ترانزیستورها کلید حل مشکل است، بهتر است زود قضاوت نکنید. هر چه مقیاس کوچک‌تر می‌شود برخی خصوصیات کوانتومی پدیدار می‌شوند و در نتیجه تعدد ترانزیستورها هزینه‌های بیشتری به وجود آورده و در عمل این امکان وجود ندارد با افزایش تعداد ترانزیستورها تحول بزرگی به وجود آورد. شرکت‌ها برای حل این مشکل به سراغ راهکارهایی همچون افزایش تعداد هسته‌های تراشه‌ها (چهار، هشت)، Hyper-threading و افزایش کش بیشتر پردازنده‌ها با هدف بهبود عملکرد رفته‌اند. این راهکارها مشکلات خاص خود را دارند. به‌طور مثال، برای بهبود راندمان پردازنده این امکان وجود ندارد تا کش بیشتری را اضافه کرد، زیرا کش محدودیت‌های فیزیکی داشته و هرچه بزرگ‌تر شود سرعت آن کمتر می‌شود. افزایش تعداد هسته‌ها نیز دردسر خاص خود را دارد و شرکت‌ها نمی‌توانند از کنار مسئله مقیاس بی تفاوت عبور کنند. افزایش تعداد هسته‌ها نیز مشکل همزمانی را به وجود می‌آورد که در ادامه به آن اشاره کوتاهی خواهیم کرد. پس اگر در حوزه سخت‌افزار با مشکلاتی روبرو هستیم، باید روی بهبود عملکرد نرم‌افزارها حساب باز کنیم. بیشتر زبان‌های برنامه‌نویسی جدید آن‌گونه که باید بهینه طراحی نشده‌اند و به خوبی نمی‌توانند از پردازنده‌ها به بهترین شکل استفاده کنند.

## Go همراه با (رویه‌ها) Goroutine است

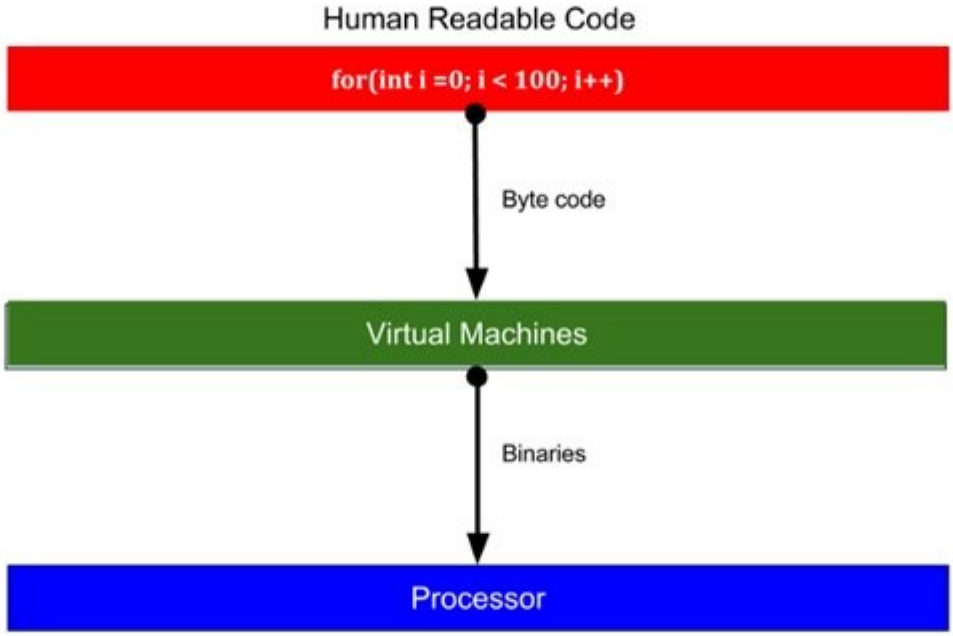
نیل روزافزون **مراکز داده** به هسته‌های بیشتر باعث شده در سال‌های پیش رو شاهد ورود پردازنده‌های جدیدی به بازار باشیم که هسته‌های بیشتری خواهند داشت. نرم‌افزارهایی که روی سرورها اجرا می‌شوند از **میکروسرویس‌های** چندگانه برای حفظ ارتباط با **بانک‌اطلاعاتی**، صف‌های پیام و نگه‌داری کش استفاده می‌کنند. به همین دلیل برنامه‌های کاربردی و زبان‌های برنامه‌نویسی نه تنها باید از قابلیت همزمانی پشتیبانی کنند، بلکه باید همزمان با تعداد هسته‌ها به ویژگی گسترش‌پذیری تجهیز شده باشند. زبان‌های برنامه‌نویسی محبوب همچون **پایتون**، **جاوا** و... در محیط دهه 90 میلادی ساخته شدند که مفهومی به غیر از تک ریسمانی مطرح نبود. درست است که این زبان‌ها از برنامه‌نویسی چندریسمانی استفاده می‌کنند اما عملکرد آن‌ها در ارتباط با اجرای همزمان، قفل ریسمان، شرایط رقابت‌پذیری و بن‌بست‌ها یکسان نیست و هر یک عملکرد متفاوتی دارند. به‌طور مثال، کار جالبی نیست یک ریسمان جدید در جاوا ایجاد کنید، زیرا هر ریسمان جدید 1 مگابایت حافظه Heap مصرف کرده و اگر به‌طور مثال 1000 ریسمان اجرا کنید در عمل یک سامانه کامپیوتری با مشکل کمبود حافظه روبرو می‌شود. از طرفی به سختی می‌توانید میان دو یا چند ریسمان ارتباطی برقرار کنید. **زبان Go** بر پایه برنامه‌نویسی چند ریسمانی ساخته شده است و مجهز به قابلیت است که به جای ریسمان از Goroutineها یا همان رویه‌های **Go**

استفاده کرده که این رویه‌ها 2 کیلوبایت از حافظه Heap را مصرف می‌کنند. در نتیجه یک برنامه می‌تواند از میلیون‌ها رویه به شکل همزمان استفاده کند. از دیگر مزایای این تکنیک می‌توان به اجرای سریع‌تر رویه‌ها نسبت به ریسمان‌ها، حل مشکل قفل متقابل در زمان به اشتراک‌گذاری ساختمان‌های داده‌ای، ارتباط یک به چند رویه‌ها و ریسمان‌های سیستم‌عامل که یک رویه تکی می‌تواند روی ریسمان‌های چندگانه اجرا شوند، یک مکانیسم اولیه برای به وجود آوردن ارتباطی ایمن میان سایر رویه‌ها و... اشاره کرد. در نتیجه باید بگوییم **زبان برنامه‌نویسی Go** در حوزه مدیریت همزمانی‌ها عملکردی بهتر از زبان‌های سنتی همچون سی، سی پلاس پلاس و جاوا دارد.

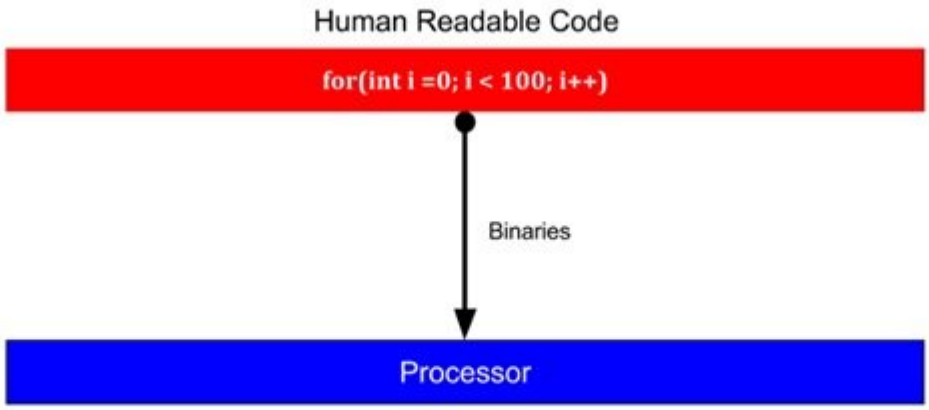



## زبان برنامه‌نویسی Go به شکل مستقیم روی سخت‌افزار اجرا می‌شود

زبان‌هایی شبیه به سی یا سی پلاس‌پلاس در مقایسه با زبان‌های محبوبی همچون جاوا، پایتون به لحاظ عملکرد در وضعیت بهتری قرار دارند، زیرا این دو زبان کامپایل می‌شوند و در نتیجه زبان‌های تفسیری نیستند. زمانی که یک برنامه کاربردی با جاوا یا سایر زبان‌های مبتنی بر ماشین مجازی جاوا نوشته می‌شود، در هنگام کامپایل پروژه، کدهای قابل فهم انسانی به بایت‌کدها تبدیل می‌شوند تا ماشین مجازی بتواند آن‌ها را درک کند، ماشین مجازی جاوا در زمان اجری بایت‌کدها را تفسیر کرده و در ادامه آن‌ها را به فایل‌های باینری قابل فهم برای پردازنده تبدیل می‌کند. اما برنامه‌های نوشته شده با سی یا سی پلاس پلاس روی ماشین مجازی اجرا نشده و در نتیجه چرخه کمتری برای اجرا دارند که همین مسئله بهبود عملکرد را به همراه دارد. به عبارت دقیق‌تر، کدهای قابل فهم انسانی به کدهای باینری تبدیل می‌شوند.



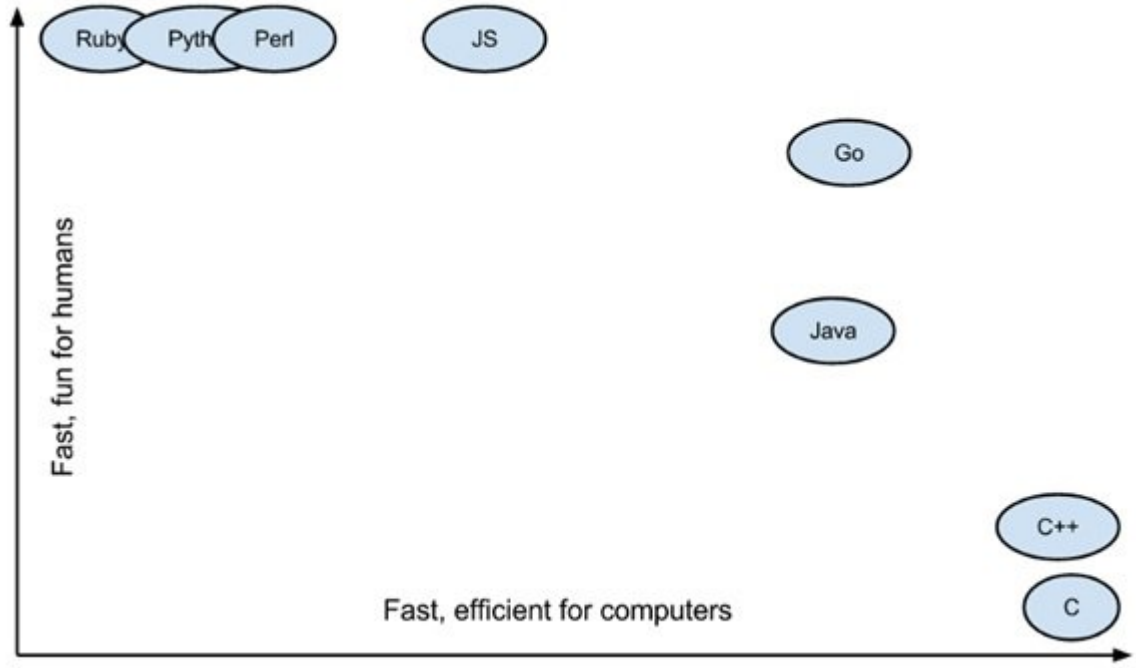

در مقابل این قابلیت ما با مشکل آزاد کرد متغیرهای تخصیص داده شده در این زبان‌ها روبرو هستیم. بیشتر زبان‌های برنامه‌نویسی برای حل این مشکل از تکنیک Garbage Collector و الگوریتم‌های Reference Counting استفاده می‌کنند.**زبان برنامه‌نویسی Go** هر دو مزیتی که به آن اشاره شد را دارد، زیر همانند کدهای سی پلاس پلاس و سی کامپایل شده و هم از تکنیک garbage collection برای آزادسازی متغیرها استفاده می‌کند.




## درGo کدها به آسانی مدیریت و نگه‌داری می‌شوند

ترکیب نحوی **زبان برنامه‌نویسی Go** ساده است. توسعه‌دهندگان **go** بیشتر ویژگی‌های موجود در زبان‌های شی‌گرا را برای خواناتر بودن کدها کنار گذاشته‌اند. در **زبان Go** شما به جای کلاس‌ها از بسته‌ها استفاده کرده و کلاس‌ها جای خود را به ساختار struct داده‌اند. **زبان برنامه‌نویسی Go** از ارث‌بری پشتیبانی نمی‌کند، در نتیجه کدها به سادگی ویرایش می‌شوند. در دنیای وراثت اگر کلاس ABC از کلاس XZY ارث‌بر کند . تغییراتی در XYZ به وجود آید، این احتمال وجود دارد تا ناخواسته یکسری پیامدهای دردسرآفرین برای کلاس‌های مشتق شده به وجود آید. عدم وجود

سازنده، Exception، Generics و Annotation برنامه‌نویسی در این زبان را بیش از پیش ساده کرده است.




## گوگل از زبان Go پشتیبانی قدرتمندی می‌کند

ممکن است پشتیبانی گوگل از **زبان Go** یک مزیت مستقیم نباشد، اما **go** از سوی تیم توسعه‌دهندگان گوگل طراحی شده و حمایت می‌شود. گوگل **زبان Go** را با هدف حل مشکلاتی در ارتباط با پشتیبانی از گسترش‌پذیری و عملکرد طراحی کرد. دو فاکتور فوق درست همان مشکلاتی هستند که کارشناسان در زمان ساخت سرورهای شخصی با آن دست به گریبان هستند.

## وجه مشترک Go با زبان‌های دیگر زیاد است

درست است که **زبان Go** تفاوت‌های محسوسی با سایر زبان‌های محبوب دارد، اما وجه اشتراک زیادی نیز دارد. به‌طور مثال، **Go** همانند سی و سی پلاس‌پلاس عملکرد بالایی داشته و همانند جاوا در زمینه مدیریت همزمانی عالی کار می‌کند. کدهای نوشته شده با **go** همانند پایتون شفاف و روشن هستند.

## زبان Go برای حل مشکل محدودیت‌های سخت‌افزارها ایده‌آل است

موقعیت دنیای نرم‌افزار و سخت‌افزار دقیقا متفاوت از چند دهه قبل شده است. زمانی که سخت‌افزارها قیمت پایینی داشتند، اما هزینه تولید نرم‌افزارها به دلیل فقدان یک رویه منسجم بسیار بالا بود و اکنون قضیه برعکس شده است. به‌طور مثال، نسخه DataCenter ویندوز سرور 2019 به قیمت 6155 هزار دلار به فروش می‌رسد، اما پردازنده Core i7-8700k به قیمت 369 دلار به فروش می‌رسد! (به شکل تقریبی 60 میلیون تومان در برابر 6 میلیون تومان.). محدودیت‌های سخت‌افزار توسعه‌دهندگان را متقاعد کرده است که به سراغ **زبان Go** بروند. توسعه‌دهندگان برای آن‌که بتوانند نرم‌افزاری بهینه برای تجهیزات اینترنت اشیا تولید کنند، باید به دنبال زبانی ایده‌آل باشند که بهترین تجربه کاربری را ارائه کند. **زبان برنامه‌نویسی Go** گزینه ایده‌آلی در این زمینه است.

**تاریخ انتشار:**
04 دی 1397

---

**نشانی منبع:**

https://www.shabakeh-mag.com/workshop/14273/%DB%B7-%D8%AF%D9%84%DB%8C%D9%84-%D9%82%D8%A7%D9%86%D8%B9-%DA%A9%D9%86%D9%86%D8%AF%D9%87-%D8%A8%D8%B1%D8%A7%DB%8C-%DB%8C%D8%A7%D8%AF%DA%AF%DB%8C%D8%B1%DB%8C-%D8%B2%D8%A8%D8%A7%D9%86-%D8%A8%D8%B1%D9%86%D8%A7%D9%85%D9%87%E2%80%8C%D9%86%D9%88%DB%8C%D8%B3%DB%8C-go