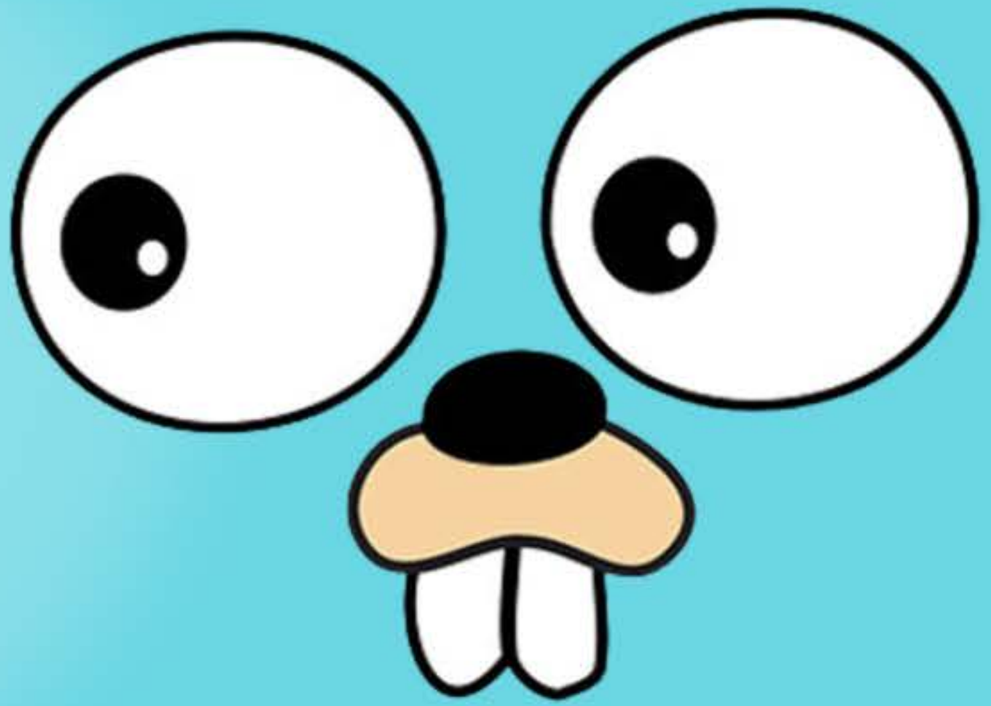
GO (LANG)

Presented by

Morgan Burch
Diana Causevic
David Yasuda
Quinn Hasselgren

WHAT IS GO?

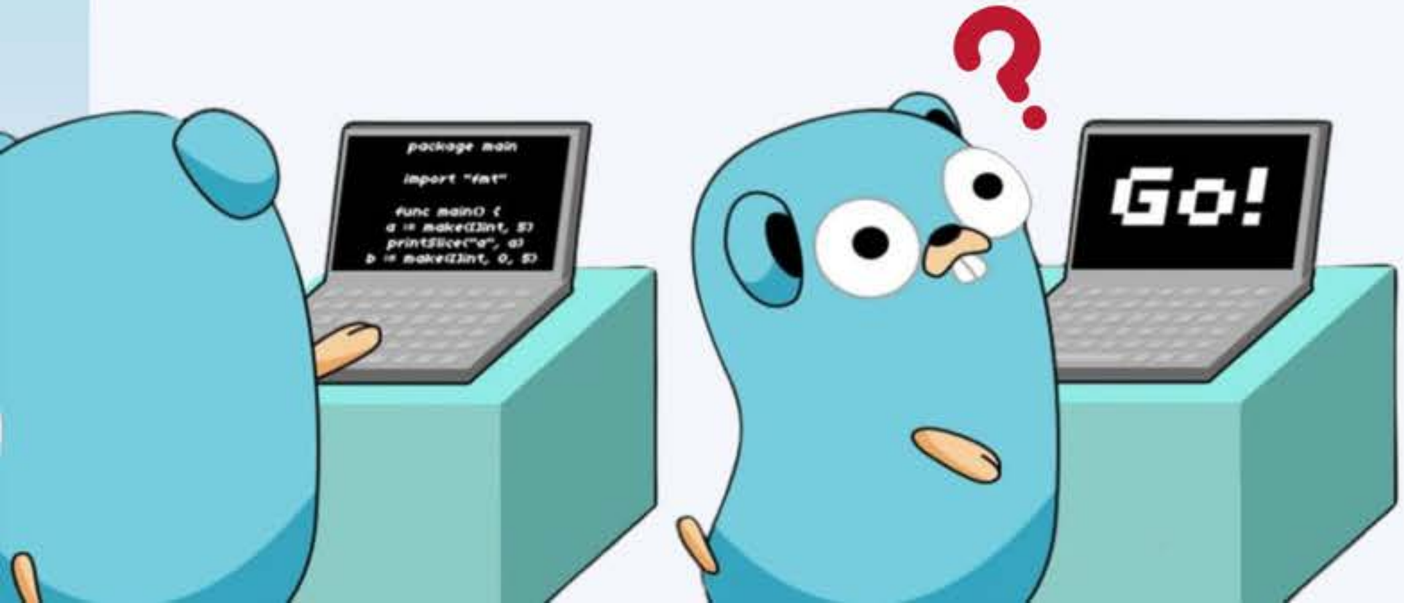
- Go or Golang is an open source programming language designed by Google engineers Robert Griesemer, Rob Pike, and Ken Thompson and was written in C in its early development.
- The GO project began in 2007.
- It was publicly announced by Google in 2009 as an open source project, and released as a beta to the world under an open source license, which allowed contributors to help in its development.
- In 2012, The Go team released GO 1, which marked the languages first stable release.
- Early 2010's, Go became an integral part of Google Cloud Platform's backend
- 2014, the Docker was written in Go, raising its popularity, and garnering attention
- 2015, GO replaced its C compiler to GO itself with the release of GO 1.5
- 2016, GO becomes one of the top 10 most popular programming languages
- 2024 GO 1.23.0 (latest release)



WHY GO WAS CREATED?

GO was created because of a deep hatred of C++. “The three of us got together and decided that we hated C++... we started off with the idea that all three of us had to be talked into every feature of the language” - Ken Thompson

- Google was using C++ mainly for the backend. GOLANG compile times are close to 4 times faster than C++



The creators wanted a language that would:

1. solve the complexities and inefficiencies in other programming languages, mainly Java and C++, which were used in Google at the time. (no inheritance or complex object oriented features.)
2. Increase development speed with faster compilation times(“GO” = fast compilation times)
3. Offer better support for concurrency which was becoming more important for scaling modern software. It was built to take advantage of multi-core processors. It can compile multiple packages at the same time, speeding up the overall process.

“The reason why I was enthusiastic about Go, I read or tried to read C++ 0x proposed standard and that was the convincer for me” - Ken Thompson

GO FEATURES

Simplicity

- Go was designed with readability in mind; it emphasizes simplicity in syntax and structure, making it easy to learn and use.
- Go's simplicity also helps with writability as developers can code quickly without getting caught up with complex syntax and features

Reliability:

- Go is known for its reliability, particularly in handling concurrent operations. Its concurrency model uses goroutines and channels to make reliable and efficient concurrent programs.
- Go also emphasizes explicit error handling, which helps create robust and error-free code.

Fast Compilation

- Go compiles to machine code, allowing for fast execution times with lower resource consumption.

Garbage Collection

- Go garbage collector is a core part of the language runtime, designed to efficiently handle memory deallocation while providing considerable speed and performance.

Standard Libraries

- The language comes with a powerful standard library for tasks like networking, file handling, and web services.
- Go's design principles are clear and consistent, making it easier to learn and use effectively.

WHO USES GO?

Google

- Designed by Google engineers, Go is used internally for various projects, including Google Chrome, Google Earth, YouTube, and Google App Engine.

Uber

- Chooses Go for backend infrastructure to handle high traffic and concurrent requests.
- Benefits include performance, scalability, and ease of maintenance, enabling quick feature releases.

Twitch

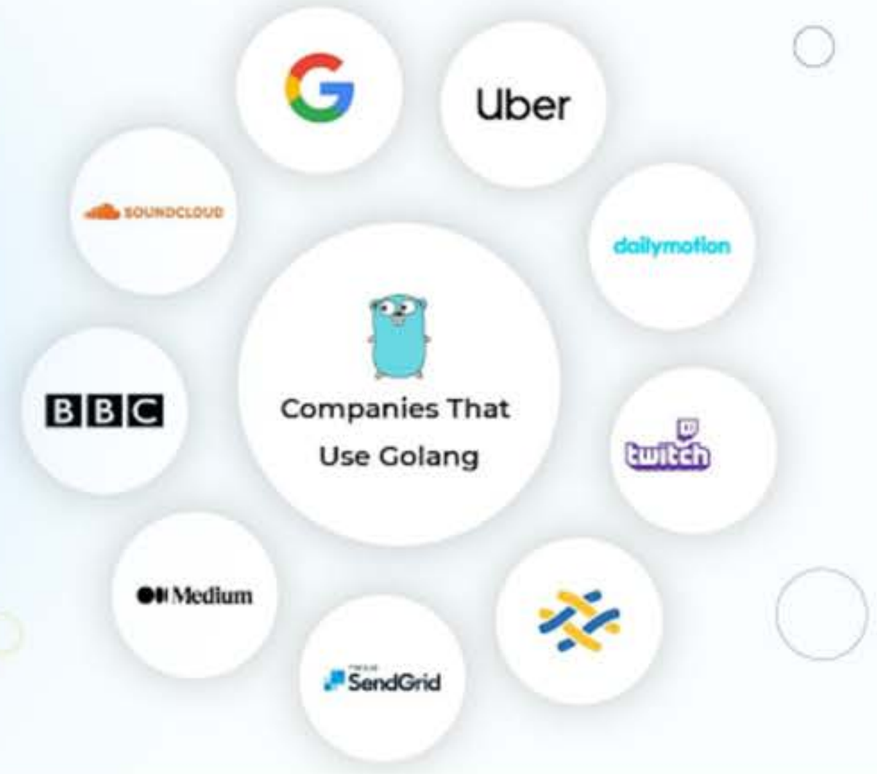
- Utilizes Go for backend services to support multiple live streams and chat rooms simultaneously.
- Offers a seamless user experience with efficient handling of traffic surges.

Dropbox

- Selects Go for its performance and reliability in managing large data volumes.
- Facilitates quick feature development and maintains fast, secure cloud storage services.

SoundCloud

- Implements Go to manage millions of streams and downloads concurrently.
- Enhances performance and scalability, ensuring uninterrupted access to music.



USE CASES OF GO

Distributed Network Services

- Ideal for developing APIs, web servers, and frameworks.
- Leverages goroutines and channels for efficient concurrency.

Cloud-Native Development

- High portability and concurrency make it perfect for cloud computing.
- Used in platforms like Kubernetes and Google Cloud for scalability and performance.

Replacements for Existing Infrastructure

- Modernizes outdated systems by rewriting them in Go.
- Example: New version of the Network Time Protocol (NTP) is built with Go.
    - Networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks.

Utilities and Stand-Alone Tools

- Minimal dependencies make Go great for small, quick-launch tools.
- Easily packaged for redistribution.

Media Platforms

- Employed by YouTube, SoundCloud, and Netflix for efficient data processing.
- Handles significant data distribution seamlessly.

On-Demand Services

- Supports fast and reliable services for everyday users.
- Uber utilizes Go to quickly respond to ride requests.

KEY GO FRAMEWORKS AND TOOLS

Popular Frameworks

1. **Gin**

- Lightweight and fast, focusing on performance and ease of use.
- Features include routing, middleware, error handling, and logging.
- Ideal for quickly building APIs with minimal configuration.

2. **Echo**

- Similar to Gin, known for speed and simplicity.
- Designed for high-performance applications and APIs, with built-in support for WebSockets.
- Features include routing, middleware, and request/response handling.

3. **Revel**

- Full-stack framework for complex web applications.
- Offers extensive built-in features like routing, controllers, and database support.
- Highly modular, allowing developers to customize feature as needed.

Debugging Tools

- GDB: Basic debugging tool supporting file viewing, breakpoints, and code disassembly.
- Delve: Preferred debugger for Go, better understanding of Go runtime and data structures.

-Supports multiple platforms (Linux, OSX, Windows).

-Features include setting breakpoints, viewing goroutines, and evaluating expressions.

Testing Tools

- Standard Testing Library: Basic automated testing framework included with Go.

-Supports unit testing with simple syntax for defining test functions.

-Example test function format: `func TestFunctionName(*testing.T)`.

- Benchmarking: Go supports benchmarking to measure performance and reliability of code execution.

-Benchmark functions use `b.N` to run code multiple times for reliable timing results.

ADVANTAGES OF GO

Simplicity and Readability

- Minimalistic Syntax: Go was designed to be simple and easy to understand.
- -Clean Design: The language avoids complex features found in other languages like inheritance, operator overloading, and generics

Performance

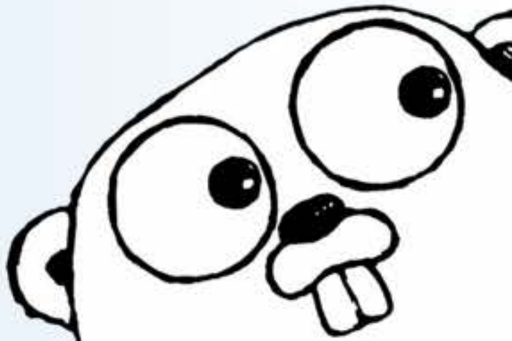
- Compiled Language: Go is a statically typed, compiled language that generates fast machine code.
- Efficient Memory Usage: Go's built-in garbage collector and memory management are optimized for efficiency.

Concurrency Support

- Goroutines: Go has built-in concurrency support via goroutines, lightweight threads that are easy to use.
- Channels: Go also provides channels for safe communication between goroutines, which helps in building scalable, concurrent systems.

Fast Compilation

- Quick Builds: Go compiles quickly compared to many other statically typed languages.



ADVANTAGES OF GO (CONT)

Strong Standard Library:

- Go's standard library is comprehensive and includes packages for web servers, cryptography, I/O, file handling, and more.

Built-in Tools:

- Go tools: Go provides excellent built-in tooling such as go test for testing, go fmt for automatic code formatting, and go vet for static code analysis.
- Dependency Management: Go modules (introduced in Go 1.11) make it easy to manage dependencies.

Strong Ecosystem and Community Support:

- Go has a strong and growing community, especially in the cloud and microservices ecosystems.

Scalability:

- Go is particularly well-suited for building scalable systems, such as web servers, microservices, and distributed applications.

Open Source:

- Go is an open-source project, meaning it has ongoing contributions from the global community and is continually improving.

LIMITATIONS OF GO

Limited Generics Support:

- **Generics Added Recently**: Although generics were introduced in Go 1.18, they are still relatively new and not as robust or flexible as in other languages like Java, C++, or Rust.
- **Lack of Advanced Features**: Go generics are intentionally kept simple, which means it may not support some advanced generic programming patterns found in other languages.

Manual Memory Management:

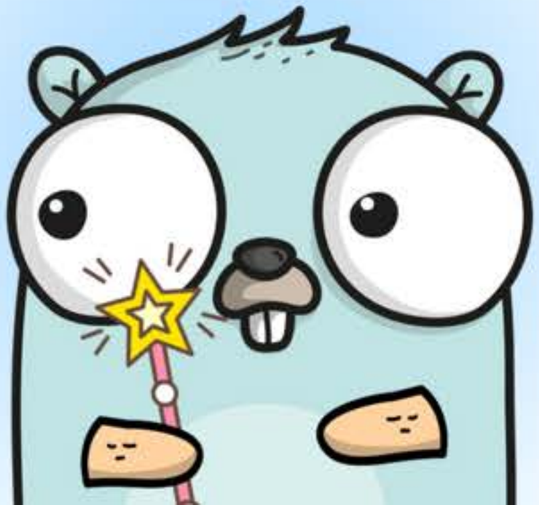
- **Garbage Collection Trade-offs**: Go has a built-in garbage collector, but it can sometimes introduce performance overhead in applications that require real-time processing or have very high memory usage.

Error Handling:

- **Verbose Error Handling**: Go's approach to error handling is explicit but can become verbose, requiring frequent checks like `if err != nil`.
- **No Exception Handling**: Go does not have traditional exception handling (try-catch blocks).

Limited Third-Party Ecosystem:

- **Fewer Libraries Compared to Other Languages:** While Go has a strong standard library, its third-party ecosystem is still growing and might not be as extensive as those of older languages like Python, Java, or JavaScript.



LIMITATIONS OF GO (CONT)

Lack of Immutability by Default:

- **Immutable data structures:** Go does not have native support for immutable data structures, making it harder to write purely functional code or prevent unintended data mutations.
- **Lack of Advanced Features:** Go generics are intentionally kept simple, which means it may not support some advanced generic programming patterns found in other languages.

Basic Dependency Management:

- **Go Modules Are Still Maturing:** Although Go Modules have improved dependency management, they are still not as feature-rich as other dependency management systems like npm (JavaScript), Cargo (Rust), or Maven (Java).

Limited GUI Development:

- **Not Suitable for Desktop Applications:** Go is mainly designed for systems programming, backend services, and command-line tools.
- **No Exception Handling:** Go does not have traditional exception handling (try-catch blocks).

No Built-in Support for Advanced Data Structures

- **No In-built Generic Data Structures:** Go lacks built-in support for common data structures like trees, heaps, or graphs, unlike languages like C++ or Java.

CONCLUSION STUFF

- Demo summary + github link
- Strengths: Go's syntax is clean and easy to read, avoiding complex features like inheritance, method overloading and operator overloading making it easier to understand, code, maintain and debug
- Go is gaining popularity because it makes development easy while maintaining speed, power and efficiency needed for modern software systems.
- Approximately 10-20% of Fortune 500 companies use GO
- 1-2 million GO programmers and counting
- Questions?!?!?!?!

SOURCES

- Go Programming Language: A Comprehensive Overview - DEV Community
- https://itnext.io/10-features-of-go-that-set-it-apart-from-other-languages-89337e5ee551
- https://bwoff.medium.com/understanding-gos-garbage-collection-415a19cc485c
- What is Golang? A Guide to the Go Programming Language (trio.dev)
- https://verpex.com/blog/website-tips/golang-frameworks
- https://trio.dev/what-is-golang-used-for/
- https://www.bairesdev.com/blog/companies-using-golang/
- https://www.techtarget.com/searchnetworking/definition/Network-Time-Protocol
- https://medium.com/@ibrahimpasha.m.d/debugging-and-testing-in-golang-93f4031b00d9#:~:text=Go%20supports%20debugging%20through%20GDB,line%20numbers%20in%20the%20program.&text=This%20command%20lists%20the%20source%20code.
- https://www.scalefocus.com/blog/why-you-should-go-with-go-for-your-next-software-project
- https://medium.com/@julienetienne/why-go-the-benefits-of-golang-6c39ea6cff7e
- https://www.tftus.com/blog/why-opt-for-golang-the-benefits-of-choosing-golang-for-your-project
- https://www.toptal.com/golang/4-go-language-criticisms
- https://chandrakant22.medium.com/what-are-the-advantages-and-disadvantages-of-golang-4c2b1cb77fbc

THANK YOU