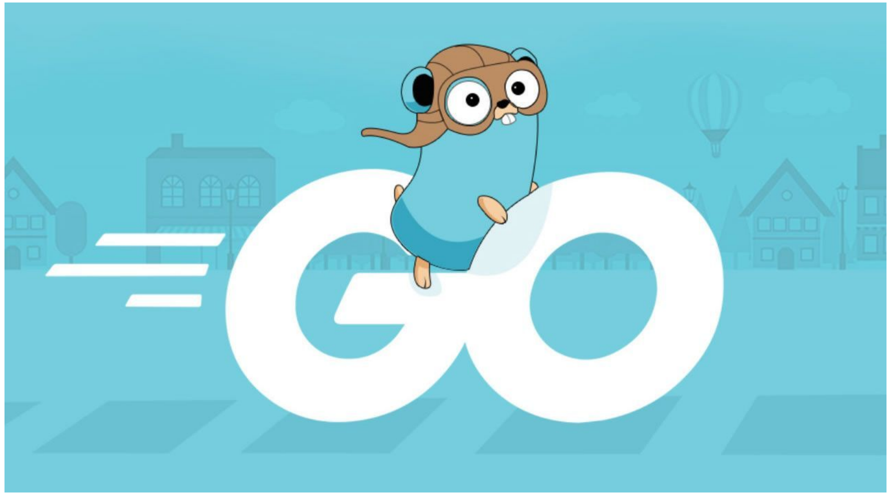
Learn by Doing

Greg Horie

Overview

- Quick Background - Go Design
- Learn By Doing Format
- Go Code Examples
- Summary
- Feedback



Go Design Inspirations

- Go is an open-source programming language developed by Google in 2007
- Designed as a next-generation C
- Borrows some syntax from C, Pascal, Modula, and Oberon
- Great for building system tools, network services, and large-scale systems

Projects Using Go

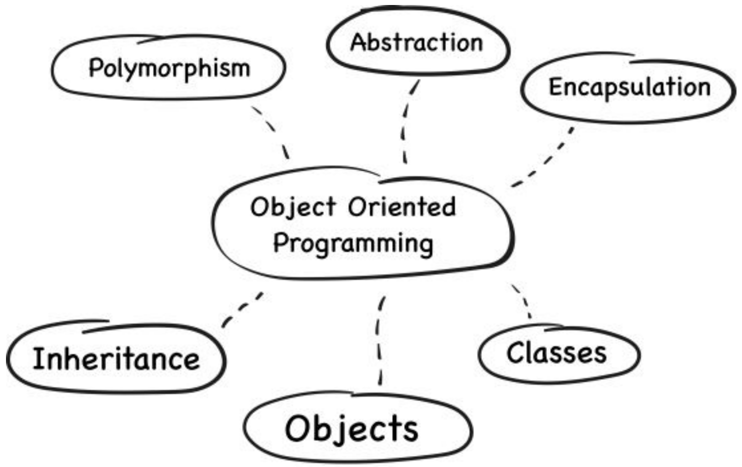
- **Kubernetes** - open-source container orchestration
- **Prometheus** - open-source monitoring
- **Docker** - container runtime and tooling
- **Terraform** - open-source infrastructure as code
- **Hugo** - static site generator

Go In A Nutshell

- Compiled, statically typed, open-source language with an active community
- Focus on readability and maintainability
- Compiled executables are operating system specific
- Compiled app contains a statically-linked run-time (no virtual machine)
- Provides the illusion of an interpreted language
- Garbage collection is a feature
- Supports built-in concurrency
- Comprehensive standard library

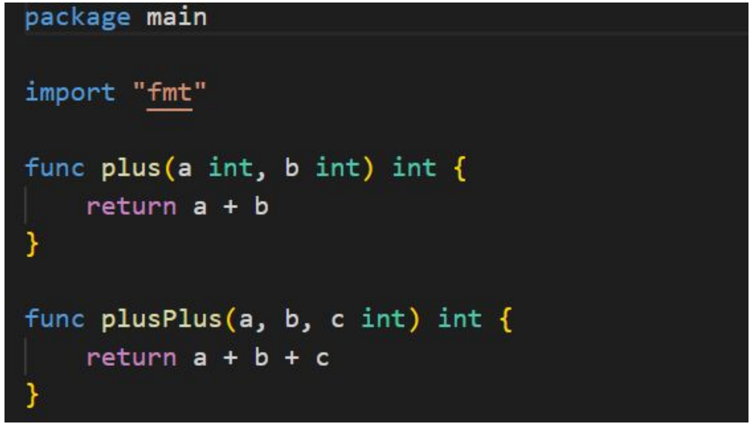
Is Go Object-Oriented?

- Has some OOP features
- Can define custom interfaces
- Can define struct types with data fields
- Can "attach" member functions (behaviour) to these structs



Some Syntax Rules

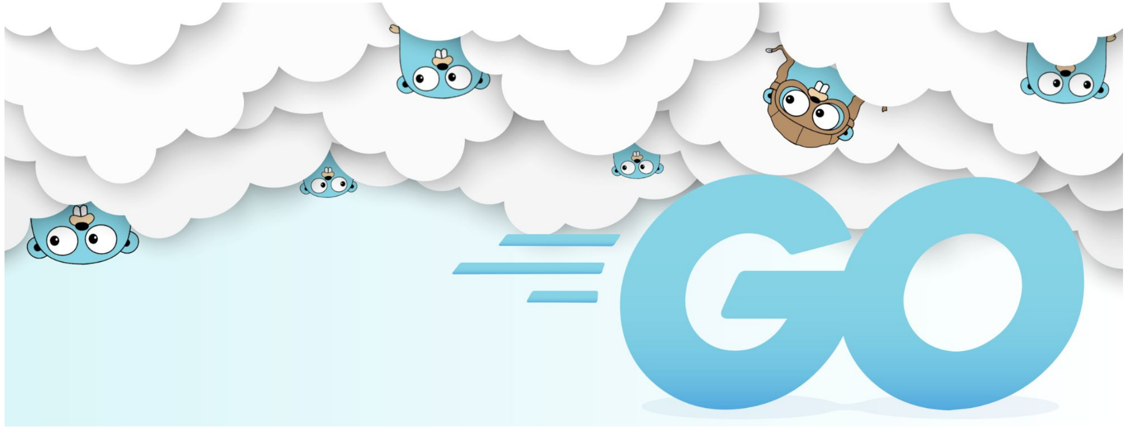
- Go is case sensitive
- Variables and package names are in lowercase and mixed case
- Initial character in public field names are uppercase
- Initial uppercase character means symbol is exported
- No semicolons required, but you can use them



```go
package main

import "fmt"

func plus(a int, b int) int {
    return a + b
}

func plusPlus(a, b, c int) int {
    return a + b + c
}
```

Learn By Doing Format

- Try a new experiment
- Code example plus discussion
  - Repeat
- I'll ask for feedback at the end of the presentation



Go Code Examples

- Check out GitHub for the examples:
    - https://github.com/netserf/vicpimakers-presentation-go-learn-by-doing

Summary

- Go is built for developers that appreciate simplicity
  - It's easy to learn even for a beginner
- It compiles down to a single executable
  - Great for publishing code
  - Removes many dependency challenges
- Comes with a rich standard library
  - Encourages building your own code over external frameworks and libraries
- If you want a modern language for building server-side components without the challenges of memory management, then try out Go.

Possible Future Discussions

- Go Deep - Learn by Doing 2
- GitHub Actions
- Diagrams as Code
- Python Dev Tools
- Kubernetes - Learn by Doing
- Google App Engine

Feedback

- Anonymous Feedback:
  - https://forms.gle/EmtLgnfQWQb9q5v86

VicPiMakers and Others Slack

- Please let us know if you want an invite to this Slack group

Backup Slides

Not Supported in Go

- No type inheritance
- No method or operator overloading
- No structured exception handling
- No implicit numeric conversions

Package vs Module

Go Package

- A directory of .go files.
- Basic building block of a Go program.
- Help to organize code into reusable components.

Go Module

- Collection of packages.
- Includes built-in dependencies and versioning.
- Out of scope for this presentation.

Syntax Rules - Braces

- Code blocks are wrapped with braces
- Starting brace MUST BE on the same line as preceding statement

```
for i := 0; i < 10; i++ {

    fmt.Println(i)

}
```

Built-In Functions

- **Link**: https://golang.org/pkg/builtin
- Go compiler assumes builtin package is always imported
- **Examples:**
- len(string) - return string length
- panic(error) - stops execution and displays error message
- recover() - manages behavior of a panicking go routine

Golang.org

- **Link**: https://golang.org
- Try the Go language playground on the homepage
- Also try the full page version on https://play.golang.org
    - See code samples listed
- **Downloads:** https://golang.org/dl/