




# GO PROGRAMMING LANGUAGE



Danielle Williams  
CS 330: Fall 2024  
December 5, 2024

# INTRODUCTION

Go is an open-source, multi-paradigm programming language that was created in 2007 by Robert Griesemer, Rob Pike, and Ken Thompson at Google.

*(Pike, 2012; Yadav, 2024)*



*Meet the Golang Gopher!*

Image via Easio

**“Today's server programs  
comprise tens of millions of  
lines of code, are worked on  
by hundreds or even  
thousands of programmers,  
and are updated literally  
every day.”**

**—ROB PIKE,  
CREATOR OF GO**





# LANGUAGE PURPOSE



Go makes the software development process more **productive** and **scalable** by reducing clumsiness and speeding up build times.

## Go solves various problems caused by:

- Networked systems
- Large-scale computing clusters
- Multicore processors
- The Web-based programming framework

*At the time, languages such as **C++**, **Java**, and **Python** were being used to circumvent rather than directly address these problems.*

## Features:

- Clear dependencies, syntax, and semantics
- Prioritizes composition over inheritance
- Garbage collected
- Concurrent
- Simple tooling (including VSCode)

# CLEAR SYNTAX

Go has both implicit and explicit declarations.

Multiple variables can be declared simultaneously.

```
a := 17
b, c := true, false
var a int = 17
var b, c bool = true, false
```

# COMPOSITION OVER INHERITANCE

Go utilizes structs instead of objects. HAS-A relationships are the only viable method of inheritance in the language.

A struct being used to create another struct is called **composition**.

*(vanigupta20024, 2020)*

```
type Student struct{
    person Person
    year int
    major string
}
```

# WHAT IS GO USED FOR?



**Creating web applications**



**Command-line tools**



**Scalable network services**

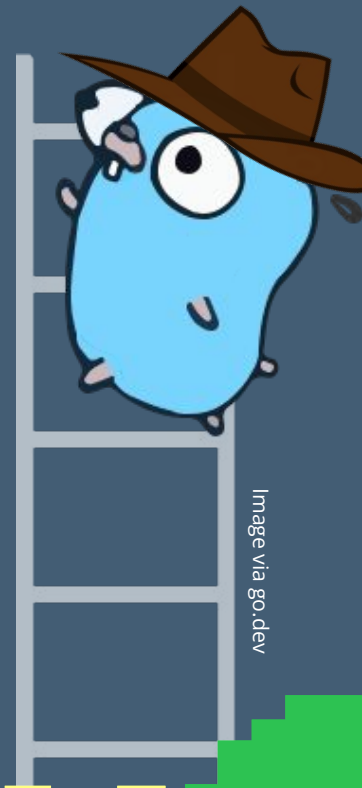


Image via go.dev

# GO PROGRAMMING PARADIGMS

## Concurrent



Computations are simultaneous with multi-threading

## Imperative



Commands show how the computation takes place

## Functional



All functions are pure and every statement evaluates to a value

## Object-Oriented



Isolated objects exchange messages with one another

# HELLO WORLD!

```
package main Package name
```

```
import "fmt" Imported package
```

```
func main() {  
    fmt.Println("Hello, World!")  
} fmt package exported method
```

## Basic Function Formula:

***func** f1(a [type1], b [type2], ...)  
[returnType] {...}*

**main()** does not take any parameters and does not have a return type.

## Syntax:

- **CamelCase** is the language standard.
- Exported methods start with a capital letter.



# FINAL PROJECT GOAL

My goal was to create a text-based RPG with a story, enemy and boss battles, and pointer-based dungeons. Throughout the progression of the game, I wanted the hero to gain experience, level-up, and learn more powerful moves.



Image via  
Ardan Labs

# PROGRAM DESIGN

What's this?  
Press enter to continue →

There seems to be a commotion outside. →

You leave your modest house and see the same troubling site.  
Dry soil, no crops.  
Oh where had the rain gone? →

You can't make a lot of money from farming these days.  
The months of drought led you to a new occupation. →

VILLAGER:  
Hey Mercenary! --  
Um, what's your name again? →

Please enter your name: Danielle

Please select a direction.  
Options: R U L (visited) r

Danielle encountered a Young Slime

Name: Young Slime  
Role: Slime  
Level: 8  
HP: 41/41

Name: Danielle  
Role: Mercenary  
Level: 10  
HP: 75/75  
Experience: 0/100

FIGHT (F) INVENTORY (I) RUN (R) f

## EnemyFactory

Attributes:  
EnemyFile string

Methods:  
CreateEnemyList() []Enemy

## Enemy

Attributes:  
baseChar Character  
level int64  
maxHP int64  
hp int64  
experience int64  
moveset []Move  
prefix string

Methods:  
Getters & setters  
isEmpty() bool  
PrintEncounterHeight() Hero  
Defeated() Hero  
doDamage(damage int64, h \*Hero)  
Academy \*Move h \*Hero  
HealMove() \*Move  
Struggle() \*Hero  
FullPP(Heal)

## Hero

Attributes:  
baseChar Character  
level int64  
maxHP int64  
hp int64  
moveset []Move  
learnableMoves map[int64]\*Move  
experience int64  
inventoryItem []Item  
inventoryHealingItem []HealingItem

Methods:  
Getters & setters  
EmptyInventoryItem() calcMaxHp() int64  
updateMaxHp() increaseExperience(exp int64)  
levelUp() learnMove() isLearnableMove() bool  
pickUpItem(item) pickUpHealingItem(hi HealingItem)  
hasRevised() int  
FullPP(Heal) RemoveHealingItem(hi HealingItem)  
isActive() bool doDamage(damage int64, e \*Enemy)  
Attack() \*Move, e \*Enemy  
HealthMove() \*Move Struggle() \*Enemy  
Run() \*Enemy bool

## MoveFactory

Attributes:  
MoveFile string

Methods:  
CreateMoveList() []Move

## Move

Attributes:  
name string  
category string  
damage int64  
maxPP int64  
pp int64  
accuracy int64  
levelLearned int64

Methods:  
Getters & setters  
DecreasePP() RestorePP() isHit() bool isHitOrMiss() bool

## Character

Attributes:  
name string  
role string

Methods:  
getters & setters

## Item

Attributes:  
name string  
category string  
desc string

Methods:  
Getters & setters  
isEmpty() bool  
FindItem(h \*Hero)

## ItemFactory

Attributes:  
ItemFile string

Methods:  
CreateItemList() []Item

## HealingItem

Attributes:  
baseItem Item  
hp int64  
rarity int64  
prefix string

Methods:  
Getters & setters  
isEmpty() bool  
ItemHealth \*Hero  
FindHealingItem(h \*Hero)

## HealingItemFactory

Attributes:  
HealingItemFile string

Methods:  
CreateHealingItemList() []HealingItem

## Room

Attributes:  
num int  
enemy []Enemy  
item []Item  
healthItem HealingItem  
up \*Room  
down \*Room  
left \*Room  
right \*Room  
boss bool  
locked bool  
visited bool

Methods:  
Getters & setters  
HasEnemy() bool  
HasItem() bool  
RemoveItem() HasHealingItem() bool  
AssignEnemyGroup([]Enemy)  
AssignHealingItem(hi HealingItem)  
pickDirection(h \*Hero) \*Room  
Interact \*Hero, eg []Enemy  
hi (HealingItem)

## Dungeon

Attributes:  
entrance \*Room  
num int  
name string  
enemyGroup []Enemy

Methods:  
DungeonTraverse(h \*Hero)  
hi (HealingItem) bool