JANUARY 2026 SERIES

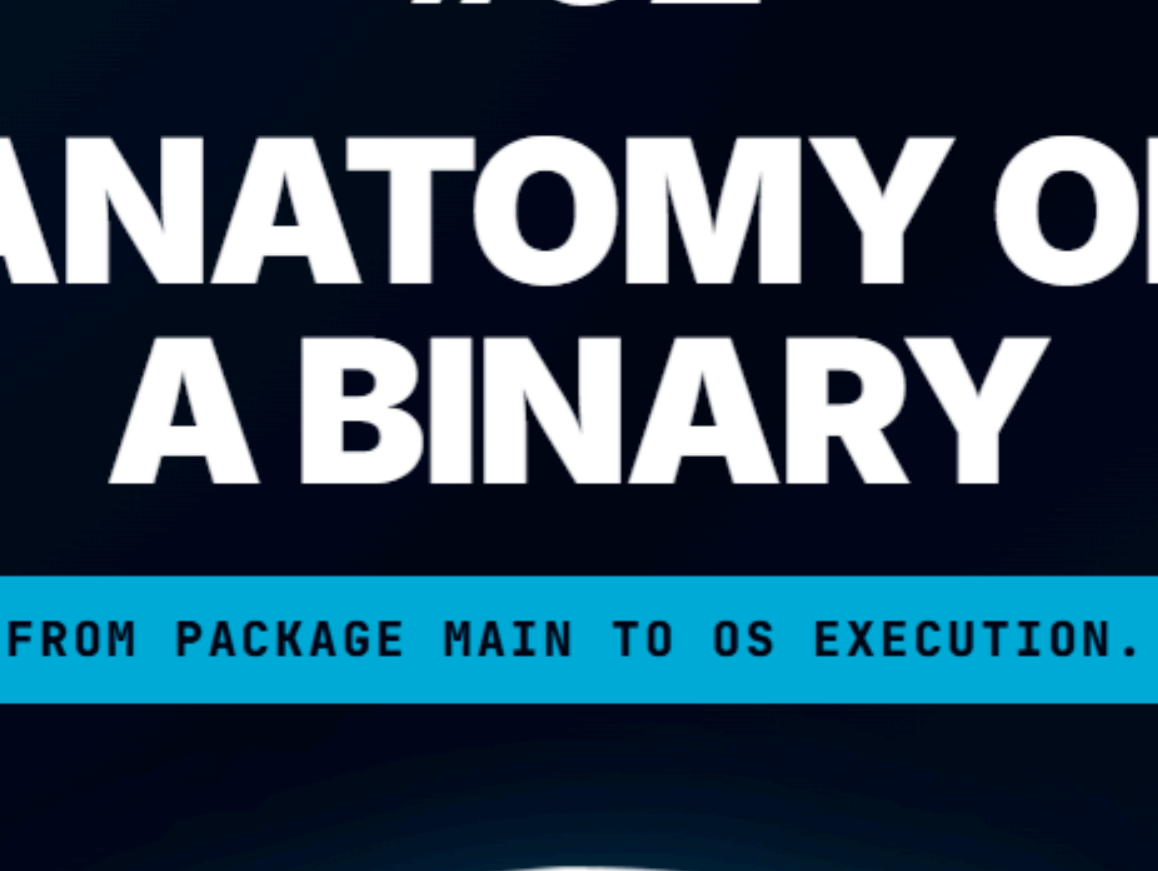
FROM GO BUILD TO GO RUN

GOLANG 2026 - NIV RAVE

#02

ANATOMY OF A BINARY

FROM PACKAGE MAIN TO OS EXECUTION.



Anatomy of a Binary

What actually happens when you run a Go program?

```
C:\Dev\GoLang\HelloWorld> go run cmd/main.go
```

+

The foundation

A simple ‘Hello World!’ program

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello, Go World!")
}
```

4 lines. 0 magic. Total transparency.

package main — The Signal

1. The Entry Point Signal

```
package main // Signal to the compiler: "Build an executable"
```

In Go, the package name main is a special reserved name.
It tells the compiler: "This is not a shared library; build a standalone executable binary."

import "fmt" - No Hidden Magic

2. Explicit Imports

```
import "fmt" // Explicitly pulling from the standard library
```

Go requires every dependency to be explicitly stated.
If you import it and don't use it, the code won't compile – this keeps your binaries lean and imports clean.

func main() - The OS Handshake

3. The Predictable Entry Point

```go
func main() {// Execution starts here after the Go runtime is ready.
    fmt.Println("Hello, Go World!")
}
```

This is where the user defined journey begins.
When you run the binary, the OS looks for this specific entry point.

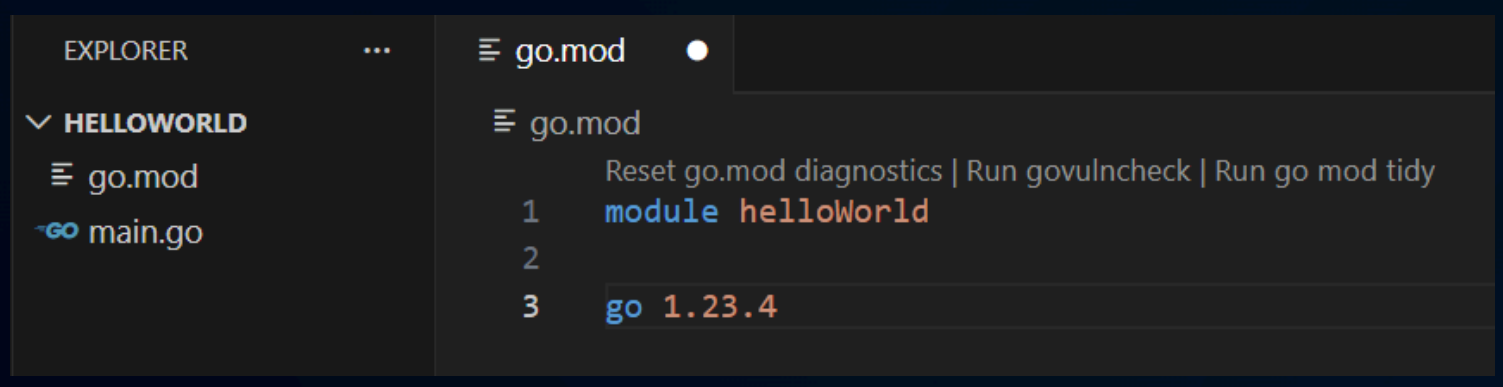
The go.mod Foundation

4. Modern Module Setup

Even for a single-file script, or simple programs, we use Go Modules.

```
C:\Dev\GoLang\HelloWorld> go mod init helloWorld
go: creating new go.mod: module helloWorld
```

This creates a go.mod file to track your Go version and dependencies.



Development vs. Production

5. How to Run & Build

Development: Compile to a temp folder and execute go run main.go

```
C:\Dev\GoLang\HelloWorld> go run main.go
Hello, Go World!
```

Production: Create a standalone, static binary go build main.go

```
C:\Dev\GoLang\HelloWorld> go build main.go
C:\Dev\GoLang\HelloWorld> ls


    Directory: C:\Dev\GoLang\HelloWorld


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         12/31/2025   5:25 PM             29 go.mod
-a----         12/31/2025   5:35 PM        2225152 main.exe
-a----         12/31/2025   5:27 PM             84 main.go
```

go run is for speed during dev. go build is for the artifact you ship to production.

GOLANG /// BA

One binary to rule them all.

Tomorrow we tackle the syntax that trips up most newcomers: Variables, Constants, and the := operator

What was the biggest "deployment hell" issue Go solved for you? Let's discuss!