**Introduction:**

**Go CLI Commands Overview**

**1. go build**

* **Description**: Compiles multiple Go source code files.
* **Usage**: Typically used to compile packages or the entire application without executing it.

**2. go run**

* **Description**: Compiles and executes one or two Go files.
* **Usage**: Ideal for running standalone Go programs quickly, especially during development.

**3. go fmt**

* **Description**: Formats all the code in each file within the current directory.
* **Usage**: Ensures code is formatted according to the standard Go style guidelines, aiding in maintaining code consistency.

**4. go install**

* **Description**: Compiles and "installs" a package.
* **Usage**: Compiles the package along with all its dependencies and moves the binaries to the workspace's bin directory or the path specified by $GOPATH/bin.

**5. go get**

* **Description**: Downloads the raw source code of someone else's package.
* **Usage**: Fetches packages from remote repositories and installs them.

**6. go test**

* **Description**: Runs any tests associated with the current project.
* **Usage**: Executes all unit tests in the package of the current directory, identifying any potential issues before deployment.

**Sample Hello World Program:**

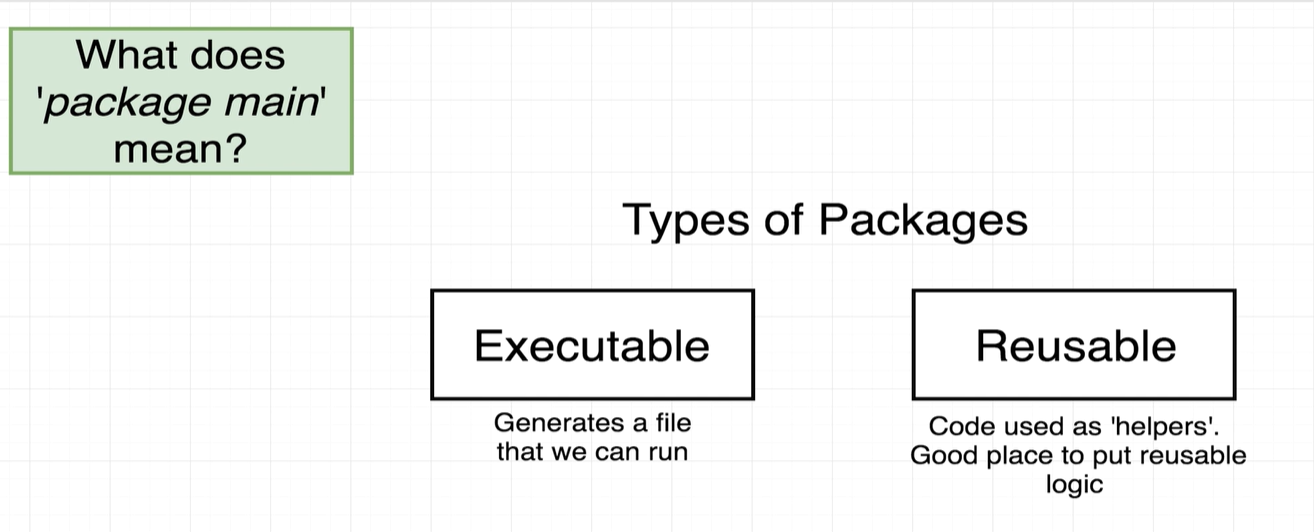
package main

import "fmt"

func main() {

    fmt.Println("Hello World!")

}



In Go, there are two main types of packages:

1. **Executable Packages**:

- These packages are designed to be compiled into executables. They always have a `main` package declaration and must include a `main` function, which serves as the entry point for the program.



2. **Library Packages**:

- These packages are used to organize and reuse code. They do not have a `main` function and are intended to be imported and used by other packages (including executable packages).



Executable packages are specifically for applications, while library packages are for shared functionality that can be reused across different parts of an application or even across different applications.

// Declares the package name. `main` is a special package name in Go that indicates

// this package will be compiled to produce an executable program.

package main

// Imports the `fmt` package, which contains functions for formatted I/O,

// including printing to the console.

import "fmt"

// The `main` function is the entry point of the program. When the executable

// runs, execution starts from this function.

func main() {

    // Calls the `Println` function from the `fmt` package to print

    // the string "Hello World!" to the console, followed by a newline character.

    fmt.Println("Hello World!")

}