**Coding convention in GOLANG**

**PACKAGES**

* We should import packages as shown below

`import ( // Standard library packages

// Internal project packages

// Third-party packages

)`

* Sort packages by alphabetical order for each group.

`import ( "net/http" //DON'T "errors" //DON'T

"nimblehq/models" //DON'T

"nimblehq/controllers" //DON'T

)`

-———————————————————————

`import ( "errors" //DO  
"net/http" //DO

"nimblehq/controllers" //DO

"nimblehq/models" //DO

)`

**VARIABLES**

* A name must begin with a letter, and can have any number of additional letters and numbers.

ex :- var num, num1 int

* The variables should have proper/meaning name and it should define why we are using it.
* Use **UpperCamelCase** for public and **lowerCamelCase** for private variables, interfaces, structs and functions.
* A variable name cannot start with a number.

ex :- var 1num int ———(not allowed)

* + If the name of a variable begins with a lower-case letter, it can only be accessed within the current package this is considered as unexported variables.

var num1 int ———(allowed)

* A variable name cannot contain spaces.

ex : var Token Num ———(not allowed)

vartoken numm ———(allowed)

* If the name of a variable begins with a capital letter, it can be accessed from packages outside the current package one this is considered as exported variables.
* If a name consists of multiple words, each word after the first should be capitalized like this: empName, EmpAddress, etc.
* Variable names are case-sensitive (car, Car and CAR are three different variables).

**GLOBAL VARIABLE**

* The first letter of the variable must be in uppercase
* ex :- var Global\_Variable int
* The variables can consists of alphabets, digits(not as first letter) and underscore
* ex := var Num1 int
* The variables should not consists of special characters in them
* ex:= var Num$ int ——(not allowed)
* The variables should not declare in the main package ,keep it in separate module and it can be called by using the package name like ‘ var package\_Name.Global\_Variable ‘

**CONSTANTS**

* Name of constants must follow the same rules as variable names, which means a valid constant name must starts with a letter or underscore, followed by any number of letters, numbers or underscores.
* By convention, constant names are usually written in uppercase letters. This is for their easy identification and differentiation from variables in the source code.
* const (
* PRODUCT = "Mobile"
* QUANTITY = 50
* PRICE = 50.50
* STOCK = true
* )

**How to declare when we have multiple constants,variables and structs?**

**Naming Conventions for GOLANG Functions:**

* Every function name should have meaningful names.
* We should mention why we’re using this particular function by commenting it.
* A name must begin with a letter, and can have any number of additional letters and numbers.

ex :- employees(),Employee5()

* A function name cannot start with a number.

ex :- 1stStudent() ——(not allowed)

* A function name cannot contain spaces.

ex :- Employee Name() ——(not allowed)

* If the functions with names that start with an uppercase letter will be exported to other packages. If the function name starts with a lowercase letter, it won't be exported to other packages, but you can call this function within the same package.
* If a name consists of multiple words, each word after the first should be capitalized like this: empName, EmpAddress, etc.
* Function names are case-sensitive (emp(), Emp() and EMP() are three different functions)
* Prefer one or two letters abbreviation as the function receiver name. Such names should be consistent with the other receiver functions.
* func (registration \*Registration) New() { // DON'T
* ... // DON'T
* ... // DON'T
* }
* func (r \*Registration) New() { // DO
* ... // DO
* ... // DO
* }
* Any go function should not exceed 50 lines and any file/module should not exceed 500 lines.

**Interface names**

By convention, one-method interfaces are named by the method name plus an -er suffix or similar modification to construct an agent noun: Reader,Writer, Formatter,CloseNotifier etc.