Part 2: Hello World

04 FEBRUARY 2017

This is the second tutorial in our Golang tutorial series. Please read Golang tutorial part 1: Introduction and installation to know about what is golang and how to install golang.

There is no better way to learn a programming language than getting our hands dirty with code. Lets go ahead and write our first go program.

I would personally recommend using Visual Studio Code with the go extension as the IDE. It has autocomplete, code styling and a host of other features.

Setting up the go workspace

Before beginning to write code, we have to setup the go workspace.

In the case of **Mac or Linux**, the go workspace should be located in $HOME/go. So lets go ahead and create a directory **go** inside **$HOME**.

In the case of **Windows**, the workspace should be located in **C:\Users\YourName\go**. So lets create the **go** directory inside **C:\Users\YourName**.

It is possible to use a different directory as the workspace by setting the GOPATH environment variable. But for now lets use the above location for simplicity.

All the source files for go should be located in a directory named **src** inside the workspace. So lets create directory **src** inside the **go** directory we created above.

Every go project should in turn have its own subdirectory inside src. Lets create a directory **hello** inside src to hold the hello world project.

The directory structure should look like the one below after creating the above directories.

```
go
  src
    hello
```

Save the following program as **helloworld.go** in the hello directory we just created.

```
package main

import "fmt"

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

Heres what the directory structure will look like after creating the above program

```
go
  src
    hello
      helloworld.go
```

Running a go program

There are a couple of different ways to run a go program. Lets look at them one by one.

1) Using **go run** command - Type `go run workspacepath/src/hello/helloworld.go` in the command prompt.

workspacepath in the above command should be replaced by the path of your work space (**C:/Users/YourName/go** in windows and **$HOME/go** in linux or Mac)

You should see the output `Hello World` in the console.

2) Using **go install** command - Run `go install hello` command followed by `workspacepath/bin/hello` to run the program.

workspacepath in the above command should be replaced by the path of your work space (**C:/Users/YourName/go** in windows and **$HOME/go** in linux or Mac). You should see the same `Hello World` output in the command line.

When you type **go install hello**, the go tool searches for the hello package (hello is called as package, we will look into packages in more detail later) inside the workspace. Then it creates a binary named `hello(hello.exe` in the case of windows) inside the bin directory of the workspace. The directory structure should like below after running go install hello

```
go
  bin
    hello
  src
    hello
      helloworld.go
```

3) The third cool way of running the program is using the go playground. Although this has its restrictions, this method comes in handy when we want to run simple programs. I have created a playground for the hello world program. Click here to run the program online.

You can use the go playground to share your source code with others.

A short explanation of the hello world program

Here is the hello world program we just wrote

```
package main //1

import "fmt" //2

func main() { //3
    fmt.Println("Hello World") //4
}
```

We will see what each line of the program does in brief here. We will dwell deep into each section in the upcoming tutorials.

package main - **Every go file must start with the** `package name` **statement.** Packages are used to provide code compartmentalisation and reusability. Here the package name used is `main`

import "fmt" - The fmt package is imported and it will be used inside the main function to print text to the standard output.

func main() - The main is a special function. The program execution starts from the main function. **The main function should always reside in the main package**. The **{** and **}** indicate the start and end of the main function.

fmt.Println("Hello World") - The **Println** function of the **fmt** package is used to write text to the standard output.

The code is available for download at github.

You can now move on to Golang tutorial part 3: Variables to learn about variables in golang.

Please post your feedback and queries in the comments section. Thank you.

Naveen Ramanathan

iOS developer at dietco.de and Golang enthusiast.