## Packages and Imports

This lesson explains how to import different libraries and packages in GO

we'll cover the following ^
Packages
Import statement examples

## Packages #

Every Go program is made up of packages. Programs start running in package main.



If you are writing an executable code (versus a library), then you need to define a main package and a main() function which will be the entry point to your software.

By convention, the package name is the same as the last element of the import path. For instance, the "math/rand" package comprises files that begin with the statement package rand.

## Import statement examples



## Or grouped:

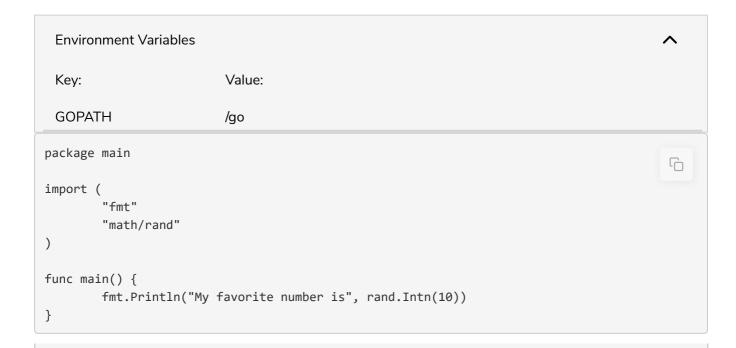


Check out the two links below for more information about *packages* and *imports* in GO.

• Go Tour: Packages

• Go Tour: Imports

Below are two examples showing how to use each of these imports.







Down below is another example to help further understand how imported packages can be used.

```
Environment Variables

Key: Value:

GOPATH /go

package main
import (
    "fmt"
    "math"
)
func main() {
    fmt.Printf("Now you have %g problems.", math.Sqrt(7))
}
```

Usually, nonstandard lib packages are namespaced using a *web url*. For instance, there is some Rails logic ported to GO, including the cryptography code used in *Rails 4*. The source code is hosted on **GitHub** containing a few packages, in the following repository:

http://github.com/mattetti/goRailsYourself

To import the crypto package, you would need to use the following import statement:

Environment Variables			
Key:	Value:		
GOPATH	/go		
<pre>import "github.com/mattetti/goRailsYourself/crypto"</pre>			

The next lesson further explains this in more detail. Read on to find out more!