

Exported names

This lesson discusses how to refer to exported names and how to use them after importing their package

WE'LL COVER THE FOLLOWING ^

- How to Export

How to Export

After importing a package, you can refer to the names it exports (meaning variables, methods, and functions that are available from outside of the package). In Go, a name is exported if it begins with a capital letter. `Foo` is an exported name, as is `FOO`. The name `foo` is not exported.

See the difference between:





Environment Variables ^

Key:	Value:
GOPATH	/go


```
package main

import (
    "fmt"
    "math"
)

func main() {
    fmt.Println(math.pi) //lowercase name
}
```



As you can see the code above gave an error as lowercase cannot be exported. On the other hand, the code below will run fine as `Pi` is begins with a capital letter so can be exported

Environment Variables 



Key:	Value:
GOPATH	/go

```
package main

import (
    "fmt"
    "math"
)

func main() {
    fmt.Println(math.Pi) //uppercase name
}
```






`Pi` is exported and can be accessed from outside the package, while `pi` isn't available.

Use the provided Go [documentation](#) or godoc.org to find exported names.

Below is another *exported names* example:


Environment Variables 





Key:	Value:
GOPATH	/go

```
package main

import (
    "fmt"
    "net/http"
)

func main() {
    fmt.Printf("HTTP status OK uses code: %d", http.StatusOK)
}
```





The next lesson will cover *functions* and *return* values. Read on to find out more!

