

# Variables & inferred typing

This lesson discusses the basics of variable declaration, initialization and inferred typing.

## WE'LL COVER THE FOLLOWING ^

- Variables Declaration
- Variable Initialization

Go is often referred to as a “simple” programming language, a language that can be learned in a few hours if you are familiar with any basic programming language. Go was designed to feel familiar and to stay as simple as possible, [the entire language specification](#) fits in just a few pages.

There are a few concepts we are going to explore before writing our first application.

## Variables Declaration #

The `var` statement declares a list of variables. The name of the variable comes first, and the type of variable is declared after it.

### Environment Variables ^

Key:	Value:
GOPATH	/go

```
var (  
    name    string  
    age     int  
    location string  
)
```



Or even

Environment Variables ^

Key:	Value:
GOPATH	/go

```
var (  
    name, location string  
    age           int  
)
```

Copy

Variables can also be declared one by one:

Environment Variables ^

Key:	Value:
GOPATH	/go

```
var name    string  
var age     int  
var location string
```

Copy

## Variable Initialization #

A `var` declaration can include initializers, one per variable.

Environment Variables ^

Key:	Value:
GOPATH	/go

```
var (  
    name    string = "Prince Oberyn"  
    age     int    = 32  
    location string = "Dorne"  
)
```

Copy

If an initializer is present, the type can be omitted, the variable will take the type of the initializer (inferred typing).

Environment Variables ^

Key:	Value:
GOPATH	/go

```
var (  
    name = "Prince Oberyn"  
    age  = 32  
    location
```

Copy

```
    age      = 32
    location = "Dorne"
)
```

You can also initialize multiple variables at once.

Environment Variables



Key:	Value:
GOPATH	/go

```
var (
    name, location, age = "Prince Oberyn", "Dorne", 32
)
```



Inside a function, the `:=` short assignment statement can be used in place of a `var` declaration with implicit type.

Environment Variables



Key:	Value:
GOPATH	/go

```
package main
import "fmt"

func main() {
    name, location := "Prince Oberyn", "Dorne"
    age := 32
    fmt.Printf("%s age %d from %s ", name, age, location)
}
```



A variable can contain any type, including functions:

Environment Variables



Key:	Value:
GOPATH	/go

```
func main() {
    action := func() { //action is a variable that contains a function
        //doing something
    }
    action()
}
```



Outside a function, every construct begins with a keyword (`var`, `func` and so on) and the `:=` construct is not available.

Use [Go's declaration Syntax](#) to read more about the Go syntax.

Let's take a look at how *constants* are declared in the next chapter.