



## Interfaces in Go

### What are interfaces in Go ?

- These are like contracts, which fulfill the job of the contractee
- Interfaces are collection of method signatures in Go.
- Useful for creating mocks in unit testing

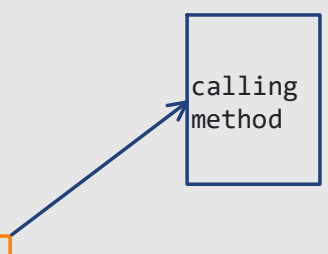
Syntax:

```
type <Name_Of_Interface> interface {  
  
    method1() float64  
    method2() float64  
    .  
    .  
    .  
    methodN() int  
}
```

Example:

```
package main  
import "fmt"  
  
func main() {  
    fmt.Printf("Interfaces in GO")  
  
    gen:= genre {  
        Name: "anime",  
    }  
    result := movie.Genre(gen)  
    fmt.Println(result) // output: anime  
}  
  
// interface  
type movie interface {  
    Genre() string  
}  
  
// struct genre  
type genre struct {  
    Name string  
}
```

calling  
method



```
// receiver function, which takes genre struct as an input
func (g genre) Genre() string {
    return g.Name
}
```

- Interfaces helps achieving the **polymorphism** feature, where in multiple object will use same method for different functionality

Example: canPlay method below, will be used by different types for different data

```
type Item interface {
    canPlay() string
}

// struct 1
type Game struct {
    Name string
}

// struct 2
type Player struct {
    Name string
}

// receiver function
func (g Game) canPlay() string {
    return g.Name
}

// receiver function
func (p Player) canPlay() string {
    return p.Name
}
```