Golang Session

- Harsh Dusane

Topic : Golang Interface

What is interface?

- An Interface is an abstract type.
- Interface describes all the methods of a method set and provides the signatures for each method.
- To create interface use interface keyword, followed by curly braces containing a list of method names, along with any parameters or return values the methods are expected to have

Defining Interface

```
type Employee interface {
        PrintName() string                // Method with string return type
        PrintAddress(id int)              // Method with int parameter
        PrintSalary(b int, t int) float64 // Method with parameters and return type
}
```

Define Type that Satisfies an Interface

```
package main
import "fmt"
// Employee is an interface for printing employee details
type Employee interface {
        PrintName(name string)
        PrintSalary(basic int, tax int) int
}
// Emp user-defined type
type Emp int
// PrintName method to print employee name
func (e Emp) PrintName(name string) {
        fmt.Println("Employee Id:\t", e)
        fmt.Println("Employee Name:\t", name)
}
// PrintSalary method to calculate employee salary
func (e Emp) PrintSalary(basic int, tax int) int {
        var salary = (basic * tax) / 100
        return basic - salary
}
func main() {
        var e1 Employee
        e1 = Emp(1)
        e1.PrintName("John Doe")
        fmt.Println("Employee Salary:", e1.PrintSalary(25000, 5))
}
```

Define Type that Satisfies Multiple Interfaces

```
package main
import "fmt"
type Polygons interface {
        Perimeter()
}
type Object interface {
        NumberOfSide()
}
type Pentagon int
func (p Pentagon) Perimeter() {
        fmt.Println("Perimeter of Pentagon", 5*p)
}
func (p Pentagon) NumberOfSide() {
        fmt.Println("Pentagon has 5 sides")
}
func main() {
        var p Polygons = Pentagon(50)
        p.Perimeter()
        var o Pentagon = p.(Pentagon)
        o.NumberOfSide()
        var obj Object = Pentagon(50)
        obj.NumberOfSide()
        var pent Pentagon = obj.(Pentagon)
        pent.Perimeter()
}
```

Interfaces with common Method

```
package main
import "fmt"
type Vehicle interface {
        Structure() []string // Common Method
        Speed() string
}
type Human interface {
        Structure() []string // Common Method
        Performance() string
}
type Car string
func (c Car) Structure() []string {
        var parts = []string{"ECU", "Engine", "Air Filters", "Wipers", "Gas Task"}
        return parts
}
func (c Car) Speed() string {
        return "200 Km/Hrs"
}
type Man string
func (m Man) Structure() []string {
        var parts = []string{"Brain", "Heart", "Nose", "Eyelashes", "Stomach"}
        return parts
}
func (m Man) Performance() string {
        return "8 Hrs/Day"
}
```

Interface Accepting Address of the Variable

```
package main
import "fmt"
type Book struct {
        author, title string
}
type Magazine struct {
        title string
        issue int
}
func (b *Book) Assign(n, t string) {
        b.author = n
        b.title = t
}
func (b *Book) Print() {
        fmt.Printf("Author: %s, Title: %s\n", b.author, b.title)
}
func (m *Magazine) Assign(t string, i int) {
        m.title = t
        m.issue = i
}
func (m *Magazine) Print() {
        fmt.Printf("Title: %s, Issue: %d\n", m.title, m.issue)
}
type Printer interface {
        Print()
}
```

Empty Interface Type

```
package main
import "fmt"
func printType(i interface{}) {
        fmt.Println(i)
}
func main() {
        var manyType interface{}
        manyType = 100
        fmt.Println(manyType)
        manyType = 200.50
        fmt.Println(manyType)
        manyType = "Germany"
        fmt.Println(manyType)
        printType("Go programming language")
        var countries = []string{"india", "japan", "canada", "australia", "russia"}
        printType(countries)
        var employee = map[string]int{"Mark": 10, "Sandy": 20}
        printType(employee)
        country := [3]string{"Japan", "Australia", "Germany"}
        printType(country)
}
```

Polymorphism

```
package main
import (
	"fmt"
)
// Geometry is an interface that defines Geometrical Calculation
type Geometry interface {
	Edges() int
}
// Pentagon defines a geometrical object
type Pentagon struct{}
// Hexagon defines a geometrical object
type Hexagon struct{}
// Octagon defines a geometrical object
type Octagon struct{}
// Decagon defines a geometrical object
type Decagon struct{}
// Edges implements the Geometry interface
func (p Pentagon) Edges() int { return 5 }
// Edges implements the Geometry interface
func (h Hexagon) Edges() int { return 6 }
// Edges implements the Geometry interface
func (o Octagon) Edges() int { return 8 }
// Edges implements the Geometry interface
func (d Decagon) Edges() int { return 10 }
```

Interface Embedding – ad hoc Polymorphism

```
package main
import "fmt"
type Geometry interface {
        Edges() int
}
type Polygons interface {
        Geometry // Interface embedding another interface
}
type Pentagon int
type Hexagon int
type Octagon int
type Decagon int
func (p Pentagon) Edges() int { return 5 }
func (h Hexagon) Edges() int  { return 6 }
func (o Octagon) Edges() int  { return 8 }
func (d Decagon) Edges() int  { return 10 }
```