Язык GO

Структуры и методы

Структуры

Структура представляет собой тип данных, объединяющий несколько именованных значений произвольных типов в единое целое.

Каждое значение называется **полем**.

Все эти поля собраны в единую сущ-ность, которая может копироваться как единое целое, передаваться функции и возвращаться ею, храниться в массивах и т.д.

Структуры. Пример

```
type Point struct {
    X int
    Y int
}

type Man struct {
    Name string
    Age int
}
```

Объекты

Объект - экземпляр структуры.

Доступ к отдельным полям объекта осуществляется с помощью записи с точкой объект.поле

Пример из жизни

Например

Есть структура для описания стула. По этому описанию мы можем изготовить стул. У стула должны быть количество ножек и цвет.

```
type chair struct{
    Legs int
    Color string
}
```

И мы получаем настоящий стул, который можно потрогать, перекрасить, добавить/удалить ножки.

Объекты. Инициализация

Объекты создаются как обычные переменные:

1) var center Point - создать объект структуры Point. У объекта будут поля X и Y.

2) center := Point {} - аналогична первой записи, но короче.

3) center := Point {20, 20} - создать объект и инициализировать поля значениями.

Объекты. Обращение к полям

```
center := Point {20, 20}

center.X += 10

center.Y += 30



now_X := center.X

now_Y := center.Y
```

Объекты. Вывод

Чтобы вывести все поля объекта, используется метод fmt.Printf("%#v\n", center)

- подробный вывод.

Можно вывести только одно поле

fmt.Println(center.X)

Объекты. Передача в функцию

```
type Rect {
    A, B int
}
func area(r Rect) int {
    return r.A * r.B
}
func main() {
    my_rect := Rect{10,20}
    fmt.Print(area(my_rect))
}
```

При передаче в таком виде объекта в функцию, происходит его копирование. Если изменить значение поля в функции, оно не изменится.

Объекты. Передача в функцию

```
func scale(r *Rect, f int) {
    r.A *= f
    r.B *= f
}
func main() {
    my_rect := Rect{10, 20}
    scale(my_rect, 5)
    fmt.Print(my_rect) //{50,100}
}
```

В случае передачи по указателю, все внесенные изменения отразятся на самом объекте.

Методы

Метод - это функция, “привязанная” ко всем объектам (экземплярам структуры). Методы доступныу убъекта через точку.

```
func (переменная Структура) имя(арг) возв {
}
func (p Rect) area() int {
    return p.A * p.B
}
func main() {
    my_rect := Rect{20, 30}
    fmt.Print(my_rect.area())
}
```

Методы. Передача по указателю.

По умолчанию, даже в метод передается копия объекта. Чтобы внутри метода изменить объект - надо передать указатель.

```
func (p *Rect) resize(f int) {
    p.A *= f
    p.B *= f
}
func main() {
    my_rect := Rect{20, 30}
    my_rect.resize(3) //Почему передаем не ссылку??
    fmt.Println(my_rect) //{60,90}
}
```