Урок Спринт 1.10

Рефлексия

Рефлексия в Go позволяет программам анализировать собственную структуру во время выполнения. В Go этот механизм выполняется с помощью пакета **reflect**.

`reflect` позволяет получить:

- количество полей в структуре и элементов в срезе
- тип значения (и проверить, нулевое ли оно)
- значение поля или элемента (и изменить их) и т. д.

Рефлексия поможет вам выяснить тип передаваемого значения. Для этого подойдёт функция `reflect.TypeOf()`, которая принимает значение любого типа и возвращает его тип в виде объекта `reflect.Type`.

Вот как можно использовать `reflect.TypeOf()`:

```
func myFunc(a interface{}) {
    t := reflect.TypeOf(a)
    fmt.Printf("Type of '%v' is %v\n", a, t)
}

func main() {
    myFunc("hello")
    myFunc(42)
}
```

Функция `myFunc` принимает интерфейс `a`. Функция `reflect.TypeOf()` получает типа значения `a`. Результат возвращается в виде объекта `reflect.Type`, с помощью которого выводится информация о типе значения.

Получение значения поля структуры

Чтобы получить значение поля структуры, будет полезен метод `FieldByName` объекта `reflect.Value`. Этот метод принимает имя поля в качестве аргумента и возвращает `reflect.Value` со значением поля:

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

func main() {
    p := Person{Name: "John", Age: 30}
    v := reflect.ValueOf(p)
    name := v.FieldByName("Name").String()
    age := v.FieldByName("Age").Int()
    fmt.Println(name, age)
}
```

Здесь мы создаём структуру `Person` с полями `Name` и `Age`. Затем получаем объект `reflect.Value` для структуры `p` и используем метод `FieldByName`, чтобы получить значения полей `Name` и `Age`. Далее преобразуем значение поля `Name` в строку с помощью метода `String`, а значение поля `Age` — в число с помощью метода `Int`.

Изменение значения поля структуры

Чтобы изменить значения поля структуры, можно использовать метод `SetField` объекта `reflect.Value`. Этот метод принимает имя поля и новое значение в качестве аргументов:

```go
type Person struct {
    Name string
    Age  int
}

func main() {
    p := Person{Name: "John", Age: 30}
    v := reflect.ValueOf(&p).Elem()
    v.FieldByName("Name").SetString("Jane")
    v.FieldByName("Age").SetInt(25)
    fmt.Println(p)
}
```

Здесь мы создаём структуру `Person` с полями `Name` и `Age`. Затем получаем указатель на структуру p и используем метод `Elem` объекта `reflect.Value`, чтобы получить значение структуры p. Далее используем метод `FieldByName`, чтобы получить значения полей `Name` и `Age`. Затем пишем методы `SetString` и `SetInt`, чтобы изменить значения полей. Наконец, мы выводим изменённую структуру p.

Кстати, на рефлексии работает больша́я часть механизмов Go — например, модуль `json`. Почитать об этом подробнее можно **здесь**. А о рефлексии в целом можно узнать из **документации**.