

Working with Requests



Michael Van Sickle

@vansimke



Introduction



Receiving Data

- Primitives
- Files
- Structs
- JSON and XML

Data Validation

Incoming Data Sources

URL

/foo/:id

Query Parameter

/foo?id=42

Request Body

POST /foo

id=42



Receiving Primitives

```
GetString(key string, default string) string
```

```
beego.Router("/foo/:id", &Controller{})
```

```
/foo/42
```

```
GetString(":id") //42
```



Receiving Primitives

```
GetStrings(key string, default []string) []string
```

```
beego.Router("/foo", &Controller{})
```

```
/foo?name=bar&name=baz
```

```
GetStrings("name") //[]string{"bar", "baz"}
```



Receiving Primitives

`GetInt(key string, default int) (int, error)`

`GetBool(key string, default bool) (bool, error)`

`GetFloat(key string, default float64) (float64, error)`



Receiving Primitives - Alternatives

```
func (c *MyController) Foo() {  
    c.Input() //url.Values  
  
    var u User  
    c.Ctx.Input.Bind(&u, "user")  
}
```

```
user.Name="Fred Flintstone"&  
user.Age=40000
```



Receiving Files

`beego.MaxMemory = 5 000 000 //in application`

`maxmemory = 5 000 000 //in configuration file`

`GetFile(key string)`

`(multipart.File, *multipart.FileHeader, error)`

`SaveToFile(fromFile, toFile string) error`



Receiving Structs

```
<form>  
  <input name="username" />  
  <input name="age" />  
  <input name="email" />  
</form>
```

```
type User struct {  
    ID int  
    Name string  
    Age int  
    Email string  
}
```



Receiving Structs

<form>

```
type User struct {  
    ID int      `form:"-`  
    Name string `form:"username"`  
    Age int     `form:"age"`  
    Email string `form:"email"`  
}
```

```
utils.ParseForm(&u) // returns error
```



Receiving JSON and XML

```
copyrequestbody = true //in configuration file
```

```
var u User
```

```
json.Unmarshal(this.Ctx.Input.RequestBody, &u)
```



Imperative Data Validation

```
type User struct {  
    Name string  
    Age  int  
}  
  
valid := validation.Validation{}  
valid.Required(u.Name, "name")  
valid.MaxSize(u.Name, 15, "nameMax")  
valid.Range(u.Age, 0, 18, "age")
```



Declarative Data Validation

```
type User struct {  
    Name string    `valid:"Required;MaxSize(15)"`  
    Age  int        `valid:"Range(0,18)"`  
}  
  
func (u *User) Valid(v *validation.Validation) {  
    //custom validation rules  
}  
  
valid.Valid(&user)
```



Validators

Required

MinSize/MaxSize

Min / Max

Range

Length

Match / NoMatch

Alpha / Numeric

...



Summary



Receiving Data

- Primitives
- Files
- Structs
- JSON and XML

Data Validation