-Struct

structstruct/json/xml/gconv

gconvstruct

```
func Struct(params interface{}, pointer interface{}, mapping ...map[string]
string) error
func StructDeep(params interface{}, pointer interface{}, mapping ...map
[string]string) error
```

1. paramsstructmap
2. pointerstructstruct
3. mappingmapstrcutparamsmap
4. StructDeepStructstruct

structhttps://godoc.org/github.com/gogf/gf/util/gconv

gconvstructstructmapping

1. struct ()
2. params
   - paramsmap struct
   - paramsstruct
   - structslice,map,strcut
3.

mapstruct

```
map    struct
name       Name           match
Email      Email          match
nickname   NickName       match
NICKNAME   NickName       match
Nick-Name  NickName       match
nick_name  NickName       match
nick name  NickName       match
NickName   Nick_Name      match
Nick-name  Nick_Name      match
nick_name  Nick_Name      match
nick name  Nick_Name      match
```

pointer**structStructstruct

```
package main

import (
        "github.com/gogf/gf/frame/g"
        "github.com/gogf/gf/util/gconv"
)

func main() {
        type User struct {
                Uid  int
                Name string
        }
        params := g.Map{
                "uid":  1,
                "name": "john",
        }
        var user *User
        if err := gconv.Struct(params, &user); err != nil {
                panic(err)
        }
        g.Dump(user)
}
```

Content Menu

-
-
- Struct
- 1
- 2

```
{
        "Name": "john",
        "Uid": 1
}
```

Struct

structparamsstructStructDeep

```
package main

import (
        "github.com/gogf/gf/frame/g"
        "github.com/gogf/gf/util/gconv"
)

func main() {
        type Ids struct {
                Id         int    `json:"id"`
                Uid        int    `json:"uid"`
        }
        type Base struct {
                Ids
                CreateTime string `json:"create_time"`
        }
        type User struct {
                Base
                Passport   string `json:"passport"`
                Password   string `json:"password"`
                Nickname   string `json:"nickname"`
        }
        data := g.Map{
                "id"          : 1,
                "uid"         : 100,
                "passport"    : "john",
                "password"    : "123456",
                "nickname"    : "John",
                "create_time" : "2019",
        }
        user := new(User)
        gconv.StructDeep(data, user)
        g.Dump(user)
}
```

```
{
        "Base": {
                "id": 1,
                "uid": 100,
                "create_time": "2019"
        },
        "nickname": "John",
        "passport": "john",
        "password": "123456"
}
```

1

```
package main

import (
    "github.com/gogf/gf/frame/g"
    "github.com/gogf/gf/util/gconv"
)

type User struct {
    Uid      int
    Name     string
    Site_Url string
    NickName string
    Pass1    string `gconv:"password1"`
    Pass2    string `gconv:"password2"`
}

func main() {
    user    := (*User)(nil)

    //
    user     = new(User)
    params1 := g.Map{
        "uid"       : 1,
        "Name"      : "john",
        "siteurl"   : "https://goframe.org",
        "nick_name" : "johng",
        "PASS1"     : "123",
        "PASS2"     : "456",
    }
    if err := gconv.Struct(params1, user); err == nil {
        g.Dump(user)
    }

    // struct tag
    user     = new(User)
    params2 := g.Map {
        "uid"       : 2,
        "name"      : "smith",
        "site-url"  : "https://goframe.org",
        "nick name" : "johng",
        "password1" : "111",
        "password2" : "222",
    }
    if err := gconv.Struct(params2, user); err == nil {
        g.Dump(user)
    }
}
```

Structmapstructstruct tagStructmap

```
{
        "Uid": 1,
        "Name": "john",
        "Site_Url": "https://goframe.org",
        "NickName": "johng",
        "Pass1": "123",
        "Pass2": "456"
}
{
        "Uid": 2,
        "Name": "smith",
        "Site_Url": "https://goframe.org",
        "NickName": "johng",
        "Pass1": "111",
        "Pass2": "222"
}
```

2

structstructnil

```
package main

import (
    "github.com/gogf/gf/util/gconv"
    "github.com/gogf/gf/frame/g"
    "fmt"
)

func main() {
    type Score struct {
        Name   string
        Result int
    }
    type User1 struct {
        Scores Score
    }
    type User2 struct {
        Scores *Score
    }

    user1  := new(User1)
    user2  := new(User2)
        scores := g.Map{
                "Scores": g.Map{
                        "Name":   "john",
                        "Result": 100,
                },
        }

    if err := gconv.Struct(scores, user1); err != nil {
        fmt.Println(err)
    } else {
        g.Dump(user1)
    }
    if err := gconv.Struct(scores, user2); err != nil {
        fmt.Println(err)
    } else {
        g.Dump(user2)
    }
}



{
        "Scores": {
                "Name": "john",
                "Result": 100
        }
}
{
        "Scores": {
                "Name": "john",
                "Result": 100
        }
}
```