-Scan

Scanstruct/struct/map/map

Content Menu


```
// Scan automatically calls MapToMap, MapToMaps, Struct or Structs
function according to
// the type of parameter `pointer` to implement the converting.
// It calls function MapToMap if `pointer` is type of *map to do the
converting.
// It calls function MapToMaps if `pointer` is type of *[]map/*[]*map to
do the converting.
// It calls function Struct if `pointer` is type of *struct/**struct to do
the converting.
// It calls function Structs if `pointer` is type of *[]struct/*[]*struct
to do the converting.
func Scan(params interface{}, pointer interface{}, mapping ...map[string]
string) (err error)
```

Struct

```
package main

import (
        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/util/gconv"
)

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

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

Struct

```
package main

import (
        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/util/gconv"
)

func main() {
        type User struct {
                Uid  int
                Name string
        }
        params := g.Slice{
                g.Map{
                        "uid":  1,
                        "name": "john",
                },
                g.Map{
                        "uid":  2,
                        "name": "smith",
                },
        }
        var users []*User
        if err := gconv.Scan(params, &users); err != nil {
                panic(err)
        }
        g.Dump(users)
}
```

```
[
    {
        Uid:  1,
        Name: "john",
    },
    {
        Uid:  2,
        Name: "smith",
    },
]
```

Map

```
package main

import (
        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/util/gconv"
)

func main() {
        var (
                user   map[string]string
                params = g.Map{
                        "uid":  1,
                        "name": "john",
                }
        )
        if err := gconv.Scan(params, &user); err != nil {
                panic(err)
        }
        g.Dump(user)
}
```

```
{
    "uid":  "1",
    "name": "john",
}
```

Map

```
package main

import (
        "github.com/gogf/gf/v2/frame/g"
        "github.com/gogf/gf/v2/util/gconv"
)

func main() {
        var (
                users  []map[string]string
                params = g.Slice{
                        g.Map{
                                "uid":  1,
                                "name": "john",
                        },
                        g.Map{
                                "uid":  2,
                                "name": "smith",
                        },
                }
        )
        if err := gconv.Scan(params, &users); err != nil {
                panic(err)
        }
        g.Dump(users)
}
```

```
[
    {
        "uid":  "1",
        "name": "john",
    },
    {
        "uid":  "2",
        "name": "smith",
    },
]
```