struct/map/slice

1struct

```
package main

import (
	"fmt"

	"github.com/gogf/gf/v2/frame/g"
	"github.com/gogf/gf/v2/os/gctx"
)

type SearchReq struct {
	Key    string `v:"required"`
	Option SearchOption
}

type SearchOption struct {
	Page int `v:"min:1"`
	Size int `v:"max:100"`
}

func main() {
	var (
		ctx = gctx.New()
		req = SearchReq{
			Key: "GoFrame",
			Option: SearchOption{
				Page: 1,
				Size: 10000,
			},
		}
	)
	err := g.Validator().Data(req).Run(ctx)
	fmt.Println(err)
}
```

```
The Size value `10000` must be equal or lesser than 100
```

2slice

Content Menu

```
package main

import (
	"fmt"

	"github.com/gogf/gf/v2/frame/g"
	"github.com/gogf/gf/v2/os/gctx"
)

func main() {
	type Student struct {
		Name string `v:"required#Student Name is required"`
		Age  int
	}
	type Teacher struct {
		Name     string
		Students []Student
	}
	var (
		ctx     = gctx.New()
		teacher = Teacher{}
		data    = g.Map{
			"name":     "john",
			"students": `[{"age":2},{"name":"jack", "age":4}]`,
		}
	)
	err := g.Validator().Assoc(data).Data(teacher).Run(ctx)
	fmt.Println(err)
}
```

```
Student Name is required
```

3map

```
package main

import (
	"fmt"

	"github.com/gogf/gf/v2/frame/g"
	"github.com/gogf/gf/v2/os/gctx"
)

func main() {
	type Student struct {
		Name string `v:"required#Student Name is required"`
		Age  int
	}
	type Teacher struct {
		Name     string
		Students map[string]Student
	}
	var (
		ctx     = gctx.New()
		teacher = Teacher{
			Name: "Smith",
			Students: map[string]Student{
				"john": {Name: "", Age: 18},
			},
		}
	)
	err := g.Validator().Data(teacher).Run(ctx)
	fmt.Println(err)
}
```

```
Student Name is required
```

```
package main

import (
	"fmt"

	"github.com/gogf/gf/v2/frame/g"
	"github.com/gogf/gf/v2/os/gctx"
)

func main() {
	type Student struct {
		Name string `v:"required"`
	}
	type Teacher struct {
		Students Student
	}
	var (
		ctx     = gctx.New()
		teacher = Teacher{}
		data    = g.Map{
			"students": nil,
		}
	)
	err := g.Validator().Assoc(data).Data(teacher).Run(ctx)
	fmt.Println(err)
}
```

```
Student Name is required
```

StudentStudentName Student Name Student

```
package main

import (
	"fmt"

	"github.com/gogf/gf/v2/frame/g"
	"github.com/gogf/gf/v2/os/gctx"
)

func main() {
	type Student struct {
		Name string `v:"required"`
	}
	type Teacher struct {
		Students *Student
	}
	var (
		ctx     = gctx.New()
		teacher = Teacher{}
		data    = g.Map{
			"students": nil,
		}
	)
	err := g.Validator().Assoc(data).Data(teacher).Run(ctx)
	fmt.Println(err)
}
```

Student*Student