Redis-

gredisredisredis+

g.RedisredisRedisconfig.yaml

Content Menu

-
  -
  -
  -
-


```
# Redis
redis:
  # 1
  default:
    address: 127.0.0.1:6379
    db:      1

  # 2
  cache:
    address:     127.0.0.1:6379
    db:          1
    pass:        123456
    idleTimeout: 600
```

default cache redis redis.default)redis

```
# Redis
redis:
   #
  default:
    address: 127.0.0.1:6379,127.0.0.1:6370
    db:      1
```

address		-	: Redis, 192.168.1.1:6379, 192.168.1.2:6379
db		0	
user		-	
pass		-	
minIdle		0	
maxIdle		10	(0)
maxActive		100	(0)
idleTimeout		10	30s/1m/1d
maxConnLifetime		30	30s/1m/1d
waitTimeout		0	30s/1m/1d
dialTimeout		0	TCP30s/1m/1d
readTimeout		0	TCPRead30s/1m/1d
writeTimeout		0	TCPWrite30s/1m/1d

masterName		-	,MasterName
tls		false	TLS
tlsSkipVerify		false	TLS
cluster		false	addressendpointtrue
protocol		3	Redis ServerRESP
sentinelUsername			Sentinel
sentinelPassword			Sentinel

config.yaml

```
# Redis
redis:
  # 1
  default:
    address: 127.0.0.1:6379
    db:      1
        pass:    "password" # ,
```

```
package main

import (
        "fmt"

        _ "github.com/gogf/gf/contrib/nosql/redis/v2"

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

func main() {
        var ctx = gctx.New()
        _, err := g.Redis().Set(ctx, "key", "value")
        if err != nil {
                g.Log().Fatal(ctx, err)
        }
        value, err := g.Redis().Get(ctx, "key")
        if err != nil {
                g.Log().Fatal(ctx, err)
        }
        fmt.Println(value.String())
}
```

```
value
```

GoFrameggredis

gredis

```
func SetConfig(config Config, name ...string)
func SetConfigByMap(m map[string]interface{}, name ...string) error
func GetConfig(name ...string) (config Config, ok bool)
func RemoveConfig(name ...string)
func ClearConfig()
```

```
package main

import (
        "fmt"

        _ "github.com/gogf/gf/contrib/nosql/redis/v2"

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

var (
        config = gredis.Config{
                Address: "127.0.0.1:6379",
                Db:      1,
                Pass:    "password",
        }
        group = "cache"
        ctx   = gctx.New()
)

func main() {
        gredis.SetConfig(&config, group)

        _, err := g.Redis(group).Set(ctx, "key", "value")
        if err != nil {
                g.Log().Fatal(ctx, err)
        }
        value, err := g.Redis(group).Get(ctx, "key")
        if err != nil {
                g.Log().Fatal(ctx, err)
        }
        fmt.Println(value.String())
}
```