-

/

```
func (l *Logger) SetLevel(level int)
func (l *Logger) SetLevelStr(levelStr string) error
func (l *Logger) SetLevelPrint(enabled bool)
```

SetLevel

SetLevelglog

```
LEVEL_ALL
LEVEL_DEV
LEVEL_PROD
LEVEL_DEBU
LEVEL_INFO
LEVEL_NOTI
LEVEL_WARN
LEVEL_ERRO
```

LEVEL_ALLLEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO |
LEVEL_CRITLEVEL_ALL & ^LEVEL_DEBU & ^LEVEL_INFO & ^LEVEL_NOTILEVEL_DEBU
/LEVEL_INFO/LEVEL_NOTI

⚠ CRIT/PANI/FATAPANI/FATApanic/exit

```
package main

import (
        "context"

        "github.com/gogf/gf/v2/os/glog"
)

func main() {
        ctx := context.TODO()
        l := glog.New()
        l.Info(ctx, "info1")
        l.SetLevel(glog.LEVEL_ALL ^ glog.LEVEL_INFO)
        l.Info(ctx, "info2")
}
```

```
2021-12-31 11:16:57.272 [INFO] info1
```

SetLevelStr

SetLevelStrlevel

Content Menu

-
  - SetLevel
  - SetLevelStr
  - SetLevelPrint

```
"ALL":      LEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO
| LEVEL_CRIT,
"DEV":      LEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO
| LEVEL_CRIT,
"DEVELOP":  LEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO
| LEVEL_CRIT,
"PROD":     LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"PRODUCT":  LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"DEBU":     LEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO
| LEVEL_CRIT,
"DEBUG":    LEVEL_DEBU | LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO
| LEVEL_CRIT,
"INFO":     LEVEL_INFO | LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"NOTI":     LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"NOTICE":   LEVEL_NOTI | LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"WARN":     LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"WARNING":  LEVEL_WARN | LEVEL_ERRO | LEVEL_CRIT,
"ERRO":     LEVEL_ERRO | LEVEL_CRIT,
"ERROR":    LEVEL_ERRO | LEVEL_CRIT,
"CRIT":     LEVEL_CRIT,
"CRITICAL": LEVEL_CRIT,
```

`DEBU < INFO < NOTI < WARN < ERRO < CRITALL`, `DEV`, `PROD`

```
package main

import (
        "context"

        "github.com/gogf/gf/v2/os/glog"
)

func main() {
        ctx := context.TODO()
        l := glog.New()
        l.Info(ctx, "info1")
        l.SetLevelStr("notice")
        l.Info(ctx, "info2")
}
```

```
2021-12-31 11:20:15.019 [INFO] info1
```

SetLevelPrint

```
package main

import (
        "context"

        "github.com/gogf/gf/v2/os/glog"
)

func main() {
        ctx := context.TODO()
        l := glog.New()
        l.Info(ctx, "info1")
        l.SetLevelPrint(false)
        l.Info(ctx, "info2")
}
```

```
2023-03-14 10:28:18.598 [INFO] info1
2023-03-14 10:28:18.631 info1
```

```
LEVEL_DEBU: "DEBU",
LEVEL_INFO: "INFO",
LEVEL_NOTI: "NOTI",
LEVEL_WARN: "WARN",
LEVEL_ERRO: "ERRO",
LEVEL_CRIT: "CRIT",
LEVEL_PANI: "PANI",
LEVEL_FATA: "FATA",
```

```
func (l *Logger) SetLevelPrefix(level int, prefix string)
func (l *Logger) SetLevelPrefixes(prefixes map[int]string)
```

```
package main

import (
        "context"

        "github.com/gogf/gf/v2/os/glog"
)

func main() {
        ctx := context.TODO()
        l := glog.New()
        l.SetLevelPrefix(glog.LEVEL_DEBU, "debug")
        l.Debug(ctx, "test")
}
```

```
2021-12-31 11:21:45.754 [debug] test
```