-

gcfg JSON/XML/YAML(YML)/TOML/INI/PROPERTIES

Content Menu

toml/yaml/yml/json/ini/xml/propertiesconfig.toml/yaml/yml/json/ini/xml
/properties

SetFileNamedefault.yaml, default.json, default.xmldefault.yamldatabase

```
//  default.yaml
g.Cfg().GetAdapter().(*gcfg.AdapterFile).SetFileName("default.yaml")

//  default.yaml
g.Cfg().Get(ctx, "database")
```

1. SetFileName
2. - gf.gcfg.file
3. - GF_GCFG_FILE

main(Linux)

1.

   ```
   g.Cfg().GetAdapter().(*gcfg.AdapterFile).SetFileName("default.yaml")
   ```

2.

   ```
   ./main --gf.gcfg.file config.prod.toml
   ```

3.
   -

     ```
     GF_GCFG_FILE=config.prod.toml ./main
     ```

   - genv

     ```
     genv Set("GF_GCFG_FILE", "config.prod.toml")
     ```

gcfgSetPathAddPath

gcfg

1. **configmanifest/config**/home/www
   a. /home/www
   b. /home/www/config
   c. /home/www/manifest/config

2. **configmanifest/config**/tmp
    a. /tmp
    b. /tmp/config
    c. /tmp/manifest/config
3. **mainconfigmanifest/config**()main/home/john/workspace/gf-app
    a. /home/john/workspace/gf-app
    b. /home/john/workspace/gf-app/config
    c. /home/john/workspace/gf-app/manifest/config

1. SetPath
2. -gf.gcfg.path
3. -GF_GCFG_PATH

main(Linux)

1.

```
g.Cfg().GetAdapter().(*gcfg.AdapterFile).SetPath("/opt/config")
```

2.

```
./main --gf.gcfg.path /opt/config/
```

3.
    - ```
      GF_GCFG_PATH /opt/config/ ./main
      ```
    - genv

      ```
      genv Set "GF_GCFG_PATH" "/opt/config"
      ```

gcfg

```
func (c *AdapterFile) SetContent(content string, file ...string)
func (c *AdapterFile) GetContent(file ...string) string
func (c *AdapterFile) RemoveContent(file ...string)
func (c *AdapterFile) ClearContent()
```

SetContent("v = 1", "config.toml")config.tomlconfig.tomlSetContent

gcfg.pattern -gjson patternconfig.yaml

```
server:
  address:    ":8199"
  serverRoot: "resource/public"

database:
  default:
    link:   "mysql:root:12345678@tcp(127.0.0.1:3306)/focus"
    debug:  true
```
```

```
// :8199
g.Cfg().Get(ctx, "server.address")

// true
g.Cfg().Get(ctx, "database.default.debug")
```

Golangmap/slice""""/gcfgmap/slicegcfg

```
// config.json:
`{"map":{"key":"value"}, "slice":[59,90]}`
```

```
var ctx = gctx.New()

m := g.Cfg().MustGet(ctx, "map").Map()
fmt.Println(m)

// Change the key-value pair.
m["key"] = "john"

// It changes the underlying key-value pair.
fmt.Println(g.Cfg().MustGet(ctx, "map").Map())

s := g.Cfg().MustGet(ctx, "slice").Slice()
fmt.Println(s)

// Change the value of specified index.
s[0] = 100

// It changes the underlying slice.
fmt.Println(g.Cfg().MustGet(ctx, "slice").Slice())

// output:
// map[key:value]
// map[key:john]
// [59 90]
// [100 90]
```