# Tigo(中文文档点击此处)

A web framework developed in go language.

## Install

```
go get github.com/karldoenitz/Tigo/...
```

## Demo

### Hello Tigo

```
package main

import "github.com/karldoenitz/Tigo/TigoWeb"

// handler
type HelloHandler struct {
    TigoWeb.BaseHandler
}

func (helloHandler *HelloHandler)Get() {
    helloHandler.ResponseAsHtml("<p1 style='color: red'>Hello Tigo!</p1>")
}

// url路由配置
var urls = map[string]interface{}{
    "/hello-tigo": &HelloHandler{},
}

// 主函数
func main() {
    application := TigoWeb.Application{
        IPAddress:  "127.0.0.1",
        Port:       8888,
        UrlPattern: urls,
    }
    application.Run()
}
```

## Compile

Open terminal, cd to target directory, input the command：

```
go build main.go
```

## Run

After compiled, there will be a runnable file named `main`, input the command：

```
./main
```

The info will display in terminal：

```
INFO: 2018/07/09 15:02:36 Application.go:22: Server run on: 0.0.0.0:8888
```

Open web browser and visit `http://127.0.0.1:8888/hello-tigo`, you will see Hello Tigo.

# API index:

# Tigo.TigoWeb

TigoWeb is the core part of Tigo framework, it contains Handler,URLpattern and Application.

## type BaseHandler

```
type BaseHandler struct {
    ResponseWriter  http.ResponseWriter
    Request        *http.Request
}
```

`BaseHandler` is the base structure of all handlers, the handlers developed by developers must extend this structure.

### func (*BaseHandler)InitHandler

```
func (baseHandler *BaseHandler)InitHandler(responseWriter http.ResponseWriter,
```

`InitHandler` is the method to initialize handler, all the `Handle` method must call this method in handlers.

### func (*BaseHandler)GetJsonValue

```
func (baseHandler *BaseHandler)GetJsonValue(key string) (interface{})
```

`GetJsonValue` is the method to get json object value in http body, the request's Content-Type must be application/json.

### func (*BaseHandler)GetParameter

```
func (baseHandler *BaseHandler)GetParameter(key string) (value string)
```

`GetParameter` is the method to get value by key.

- Get the value in url, form or json.

## func (*BaseHandler)GetHeader

```
func (baseHandler *BaseHandler)GetHeader(name string) (value string)
```

`GetHeader` is the method to get http header.

## func (*BaseHandler)SetHeader

```
func (baseHandler *BaseHandler)SetHeader(name string, value string)
```

`SetHeader` is the method to set http header.

## func (*BaseHandler)GetCookie

```
func (baseHandler *BaseHandler)GetCookie(name string) (value string)
```

`GetCookie` is the method to get cookie value.

## func (*BaseHandler)SetCookie

```
func (baseHandler *BaseHandler)SetCookie(name string, value string)
```

`SetCookie` is the method to set cookie value.

## func (*BaseHandler)GetSecureCookie

```
func (baseHandler *BaseHandler)GetSecureCookie(name string, key ...string) (val
```

`GetSecureCookie` is the method to get cookie's security value.

- The key pass to method will instead of the key in configuration;

- This method can decrypt cookie value without key.

## func (*BaseHandler)SetSecureCookie

```
func (baseHandler *BaseHandler)SetSecureCookie(name string, value string, key .
```

`SetSecureCookie` is the method to set cookie's security value.

- The key pass to method will instead of the key in configuration;
- This method can encrypt cookie value without key.

## func (*BaseHandler)GetCookieObject

```
func (baseHandler *BaseHandler)GetCookieObject(name ...string) (Cookie, error)
```

`GetCookieObject` is the method to get cookie object by name.

## func (*BaseHandler)SetCookieObject

```
func (baseHandler *BaseHandler)SetCookieObject(cookie Cookie)
```

`SetCookieObject` is the method to set cookie object by name.

## func (*BaseHandler)ClearCookie

```
func (baseHandler *BaseHandler)ClearCookie(name string)
```

`ClearCookie` is the method to clear cookie.

## func (*BaseHandler)ClearAllCookie

```
func (baseHandler *BaseHandler)ClearAllCookie()
```

`ClearAllCookie` is the method to clear all cookie.

## func (*BaseHandler)Redirect

```
func (baseHandler *BaseHandler)Redirect(url string, expire ...time.Time)
```

`Redirect` is the method to redirect client to another url.

## func (*BaseHandler)RedirectPermanently

```
func (baseHandler *BaseHandler)RedirectPermanently(url string)
```

`RedirectPermanently` is the method to redirect client to another url permanently.

## func (baseHandler *BaseHandler)Render

```
func (baseHandler *BaseHandler)Render(data interface{}, templates ...string)
```

`Render` is the method to compile html template and response the result to client.

parameters:

- data: the instance of any type structure;
- templates: the name of html templates.

**Attention:**
If not configure base path of templates in configuration, this method will use relative path.

## func (*BaseHandler)ResponseAsHtml

```
func (baseHandler *BaseHandler)ResponseAsHtml(result string)
```

`ResponseAsHtml` is the method to response with html.

## func (*BaseHandler)ResponseAsText

```
func (baseHandler *BaseHandler)ResponseAsText(result string)
```

`ResponseAsText` is the method to response with text.

## func (*BaseHandler)ResponseAsJson

```
func (baseHandler *BaseHandler)ResponseAsJson(response interface{})
```

`ResponseAsJson` is the method to response with json, this method will response empty string if convert response object to json failed.

### func (*BaseHandler)ToJson

```
func (baseHandler *BaseHandler)ToJson(response interface{}) (result string)
```

`ToJson` is the method to convert response object to json string.

### func (*BaseHandler)DumpHttpRequestMsg

```
func (baseHandler *BaseHandler)DumpHttpRequestMsg(logLevel int) (result string)
```

`DumpHttpRequestMsg` is the method to dump http request message to console or log file. The parameter `logLevel` 's value:

- 1: dump message to trace level // logger.TraceLevel
- 2: dump message to info level // logger.InfoLevel
- 3: dump message to warning level // logger.WarningLevel
- 4: dump message to error level // logger.ErrorLevel
- others: dump message to console

## type UrlPattern

```
type UrlPattern struct {
    UrlMapping map[string] interface{Handle(http.ResponseWriter, *http.Request)
}
```

URL mapping structure.

### func (urlPattern *UrlPattern)AppendUrlPattern

```
func (urlPattern *UrlPattern)AppendUrlPattern(uri string, v interface{Handle(ht
```

Use this method to append a handler to an url.

### func (urlPattern *UrlPattern)Init

```
func (urlPattern *UrlPattern)Init()
```

Initialize url mapping.

## type Application

```go
type Application struct {
    IPAddress  string      // IP address
    Port       int         // port
    UrlPattern UrlPattern  // url mapping
    ConfigPath string      // global config
}
```

Application is the launcher of web server developed by Tigo.

- IPAddress: IP address, developers can config IP address in configuration file;
- Port: Port, developers can configure it in configuration file;
- UrlPattern: url mapping.
- ConfigPath: the configuration's path.

### func (application *Application)Run

```go
func (application *Application)Run()
```

Run the web server use this method, if config `cert` and `key` file, the server will run with https protocol.

## type Cookie

```go
type Cookie struct {
    Name        string
    Value       string

    IsSecurity  bool      // whether encrypt cookie's value
    SecurityKey string    // the key encrypt cookie's value

    Path        string    // options
    Domain      string    // options
    Expires     time.Time // options
    RawExpires  string    // for reading cookies only

    // MaxAge=0 means no 'Max-Age' attribute specified.
    // MaxAge<0 means delete cookie now, equivalently 'Max-Age: 0'
    // MaxAge>0 means Max-Age attribute present and given in seconds
    MaxAge      int
    Secure      bool
    HttpOnly    bool
    Raw         string
    Unparsed   []string    // Raw text of unparsed attribute-value pairs
}
```

The cookie structure, use this structure to operate the cookie.

## func (cookie *Cookie)GetCookieEncodeValue

```
func (cookie *Cookie)GetCookieEncodeValue()(result string)
```

Use this method to get cookie's encode value.

## func (cookie *Cookie)GetCookieDecodeValue

```
func (cookie *Cookie)GetCookieDecodeValue()(result string)
```

Use this method to get cookie's decode value.

## func (cookie *Cookie)ToHttpCookie

```
func (cookie *Cookie)ToHttpCookie()(http.Cookie)
```

Convert cookie to http.Cookie.

## func (cookie *Cookie)ConvertFromHttpCookie

```
func (cookie *Cookie)ConvertFromHttpCookie(httpCookie http.Cookie)
```

Convert http.Cookie to cookie.

## func (cookie *Cookie)SetSecurityKey

```
func (cookie *Cookie)SetSecurityKey(key string)
```

Set security cookie's key.

## type BaseResponse

```go
type BaseResponse struct {

}
```

The structure extend this structure can use method `func (baseHandler *BaseHandler)ResponseAsJson(response Response)` to response json to client.

### func (baseResponse *BaseResponse)Print

```
func (baseResponse *BaseResponse)Print()
```

Use this method to print json to console.

### func (baseResponse *BaseResponse)ToJson

```
func (baseResponse *BaseResponse)ToJson() (string)
```

Use this method to convert object to json string.

## type GlobalConfig

```go
type GlobalConfig struct {
    IP       string           `json:"ip"`        // IP address
    Port     int              `json:"port"`      // Port
    Cert     string           `json:"cert"`      // https cert
    CertKey  string           `json:"cert_key"`  // https key
    Cookie   string           `json:"cookie"`    // cookie's security key
    Log      logger.LogLevel  `json:"log_path"`  // log config
}
```

### func (globalConfig *GlobalConfig)Init

```go
func (globalConfig *GlobalConfig)Init(configPath string)
```

Use configuration file to initialize global configuration.
Glance:

- IP: IP address
- Port: the Port server blinded

- Cert: the path of https cert file
- CertKey: the path of https key file
- Cookie: the key to encrypt/decrypt cookie
- Log: the instance of Tigo.logger.LogLevel

configuration.json example：

```
{
    "ip": "0.0.0.0",
    "port": 8888,
    "cert": "/home/work/certfile.ext"
    "cert_key": "/home/work/certkeyfile.ext",
    "log_path": "/home/work/log/server_run.log",
    "cookie": "thisiscookiekey"
}
```

## utils

Encrypt method

```
func Encrypt(src[]byte, key []byte) string
```

Use this method to encrypt byte array.
Decrypt method

```
func Decrypt(src[]byte, key []byte) ([]byte)
```

Use this method to decrypt byte array.
Initialize global configuration

```
func InitGlobalConfig(configPath string)
```

Use this method to initialize global configuration.

# Tigo.logger

Use this package to print log.

## Demo

The demo about using `logger` package in the web application developed by Tigo.

```
package main

import (
    "net/http"
    "github.com/karldoenitz/Tigo/TigoWeb"
    "github.com/karldoenitz/Tigo/logger"
)

type HelloHandler struct {
    TigoWeb.BaseHandler
}

func (helloHandler *HelloHandler)Get() {
    logger.Info.Printf("info data: %s", "test") // print log here
    helloHandler.ResponseAsHtml("<p1 style='color: red'>Hello Tigo!</p1>")
}

var urls = map[string]interface{}{
    "/hello-tigo": &HelloHandler{},
}

func main() {
    application := TigoWeb.Application{
        UrlPattern: urls,
        ConfigPath: "./configuration.json",  // use configuration file to confi
    }
    application.Run()
}
```

`configuration.json` content:

```
{
  "cookie": "TencentCode",
  "ip": "0.0.0.0",
  "port": 8080,
  "log": {
    "trace": "stdout",  // only print trace message in console
    "info": "/Users/karllee/Desktop/run-info.log",  // print info message in ru
    "warning": "/Users/karllee/Desktop/run.log",  // print warning info and err
    "error": "/Users/karllee/Desktop/run.log",
    "time_roll": "H*2"  // slice log file every 2 hours
  }
}
```

If you wanna use logger package in your application not base on Tigo, you can use `func (globalConfig *GlobalConfig)Init(configPath string)` function to initialize logger package.

## Structure

The structure in logger package:

### type LogLevel

```
// log level structure
//   - Trace
//   - Info
//   - Warning
//   - Error
//   - TimeRoll
// discard: discard, stdout: print in console; the path of log file
type LogLevel struct {
    Trace    string   `json:"trace"`
    Info     string   `json:"info"`
    Warning  string   `json:"warning"`
    Error    string   `json:"error"`
    TimeRoll string   `json:"time_roll"`
}
```

Initialize this structure and pass the instance to `InitLoggerWithObject` to init logger package.
TimeRoll:

- D: slice log file by Day, E.g: "D*6" slice log file every six days.
- H: slice log file by Hour, E.g: "H*6" slice log file every six hours.
- M: slice log file by Minute, E.g: "M*6" slice log file every six minutes.
- S: slice log file by Second, E.g: "S*6" slice log file every six seconds.

## logger inner functions

### func SetLogPath

Set Log file's Path

```
func SetLogPath(logPath string)
```

example：

```
import "github.com/karldoenitz/Tigo/logger"

logger.Info.Printf("It is a test...")
logger.Warning.Printf("warning!")
logger.Error.Printf("ERROR!!!")
```

Attention: this method can override the config you set before.

## func InitLoggerWithConfigFile

```
func InitLoggerWithConfigFile(filePath string)
```

Initialize logger with configuration file.

## func InitLoggerWithObject

```
func InitLoggerWithObject(logLevel LogLevel)
```

Initialize logger with LogLevel instance.

## func InitTrace

```
func InitTrace(level string)
```

Initialize Trace instance.
Parameter values:

- discard: discard log message;
- stdout: print log message in console;
- real file path: the path of log file.

## func InitInfo

```
func InitInfo(level string)
```

Initialize Info instance.
Parameter values:

- discard: discard log message;
- stdout: print log message in console;
- real file path: the path of log file.

## func InitWarning

```
func InitWarning(level string)
```

Initialize Warning instance.
Parameter values:

- discard: discard log message;
- stdout: print log message in console;
- real file path: the path of log file.

### func InitError

```
func InitError(level string)
```

Initialize Error instance.
Parameter values:

- discard: discard log message;
- stdout: print log message in console;
- real file path: the path of log file.

# Tigo.request

Tigo.request provides some functions to request an url.

## type Response

```
type Response struct {
    *http.Response
    Content []byte
}
```

HTTP Response Instance.

### func (response *Response)ToContentStr

```
func (response *Response)ToContentStr() string
```

Convert `Response.Content` to string type.

## request module

## func Request

```go
func Request(method string, requestUrl string, postParams map[string]interface{
```

Make requestion to an url

## func Get

```go
func Get(requestUrl string, headers ...map[string]string) (*Response, error)
```

Http Get method.

## func Post

```go
func Post(requestUrl string, postParams map[string]interface{}, headers ...map[
```

Http Post method.

## func Put

```go
func Put(requestUrl string, postParams map[string]interface{}, headers ...map[s
```

Http Put method.

## func Patch

```go
func Patch(requestUrl string, postParams map[string]interface{}, headers ...map
```

Http Patch method.

## func Head

```go
func Head(requestUrl string, headers ...map[string]string) (*Response, error)
```

Http Head method.

## func Options

```
func Options(requestUrl string, headers ...map[string]string) (*Response, error
```

Http Options method.

## func Delete

```
func Delete(requestUrl string, headers ...map[string]string) (*Response, error)
```

Http Delete method.