/\*\*

lumberjack介绍

Lumberjack是一个Go包，用于将日志写入滚动文件。

zap 不支持文件归档，如果要支持文件按大小或者时间归档，需要使用lumberjack，lumberjack也是zap官方推荐的。

\*/

import (

"go.uber.org/zap"

"go.uber.org/zap/zapcore"

"gopkg.in/natefinch/lumberjack.v2"

"os"

"path/filepath"

"strings"

)

var logger \*zap.SugaredLogger

func InitLog(level string, filePath string) {

hook := lumberjack.Logger{

Filename: filePath, // 日志文件路径

MaxSize: 1, // 每个日志文件保存的最大尺寸 单位：M

MaxBackups: 3, // 日志文件最多保存多少个备份

MaxAge: 7, // 文件最多保存多少天

Compress: true, // 是否压缩

}

//json key 自定义、时间格式化、日志级别

encoderConfig := zapcore.EncoderConfig{

TimeKey: "time",

LevelKey: "level",

NameKey: "logger",

CallerKey: "linenum",

MessageKey: "msg",

StacktraceKey: "stacktrace",

LineEnding: zapcore.DefaultLineEnding,

EncodeLevel: zapcore.LowercaseLevelEncoder, // 小写编码器

EncodeTime: zapcore.TimeEncoderOfLayout("2006-01-02 15:04:05.000"), // 自定义时间格式

EncodeDuration: zapcore.SecondsDurationEncoder, //

EncodeCaller: zapcore.FullCallerEncoder, // 全路径编码器

EncodeName: zapcore.FullNameEncoder,

}

//设置日志级别

atomicLevel := zap.NewAtomicLevel()

atomicLevel.SetLevel(getLevel(level))

core := zapcore.NewCore(

zapcore.NewJSONEncoder(encoderConfig), // 编码器配置

zapcore.NewMultiWriteSyncer(zapcore.AddSync(os.Stdout), zapcore.AddSync(&hook)), // 打印到控制台和文件

atomicLevel, // 日志级别

)

// 开启开发模式，堆栈跟踪

caller := zap.AddCaller()

// 开启文件及行号

development := zap.Development()

// 设置初始化字段

//filed := zap.Fields(zap.String("serviceName", "serviceName"))

// 构造日志

logger = zap.New(core, caller, development, zap.AddCallerSkip(1)).Sugar()

logger.Info("初始化日志")

}

func getLevel(level string) (l zapcore.Level) {

switch level {

case "debug":

l = zap.DebugLevel

case "info":

l = zap.InfoLevel

case "warn":

l = zap.WarnLevel

case "error":

l = zap.ErrorLevel

case "panic":

l = zap.PanicLevel

case "fatal":

l = zap.FatalLevel

default:

l = zap.InfoLevel

}

return

}

//获取当前文件夹

func getCurrentDirectory() string {

dir, err := filepath.Abs(filepath.Dir(os.Args[0]))

if err != nil {

logger.Info(err)

}

return strings.Replace(dir, "\\", "/", -1)

}

//重新包装日志的输出等级

func Debug(args ...interface{}) {

logger.Debug(args...)

}

func Debugf(template string, args ...interface{}) {

logger.Debugf(template, args...)

}

func Info(args ...interface{}) {

logger.Info(args...)

}

func Infof(template string, args ...interface{}) {

logger.Infof(template, args...)

}

func Warn(args ...interface{}) {

logger.Warn(args...)

}

func Warnf(template string, args ...interface{}) {

logger.Warnf(template, args...)

}

func Error(args ...interface{}) {

logger.Error(args...)

}

func Errorf(template string, args ...interface{}) {

logger.Errorf(template, args...)

}

func DPanic(args ...interface{}) {

logger.DPanic(args...)

}

func DPanicf(template string, args ...interface{}) {

logger.DPanicf(template, args...)

}

func Panic(args ...interface{}) {

logger.Panic(args...)

}

func Panicf(template string, args ...interface{}) {

logger.Panicf(template, args...)

}

func Fatal(args ...interface{}) {

logger.Fatal(args...)

}

func Fatalf(template string, args ...interface{}) {

logger.Fatalf(template, args...)

}