# golang 注释规范

# 前言

好的代码规范是一个程序员的基本修炼，好的代码注释更能体现一个程序员的思维逻辑，虽然代码是用来给机器运行的，我们只要能写出能让编译器运行的代码就行了，但是如果没有好的编码规范，到项目后期，加入开发的人员逐渐增多时，每个人的编码风格都不一样，这就会让项目维护者很难维护，所以开始就要制定一些好的规范来让大家遵守，这样才能写出可维护，健壮的项目。

# 基本规则

## 注释应该使代码更加清晰易懂

## 注释要简洁明了，只要提供能够明确理解程序必要的信息就可以了。如果注释过于复杂会影响程序整洁度和阅读感。

## 注释不仅描述程序作了什么，还要描述为什么这样做以及约束。

## 对于一般的getter和setter方法不用注释。

## 类、接口、构造函数、方法、全局变量必须添加注释。字段属性可以选择添加简单注释。

## 简单注释一般不超过10个字。

## 特殊地方必须要添加注释。比如一下几个地方：典型算法，代码不明晰处，在代码修改处，在循环和逻辑分支组成代码处，为他人提供的接口

# 约定

## 版权注释

版权注释主要用来声明公司的一些基本信息等:

/\*\*

\* projectName: xxx

\* fileName: network.go

\* packageName: xxxx

\* date: 日期时间，如：2020-07-21 10:20

\* copyright(c) nova

\*/

## 包注释

每个包应该有一个包注释，一个位于package子句之前行注释，包注释包含下面的基本信息。

/\*\*

\* packageName: xxx

\* description: 简单描述包功能

\* date: 日期时间，如：2020-07-21 10:20

\* copyright(c) nova公司

\*/

## 类/接口注释(Class)

类注释(Class)主要用来声明该类用来做什么，以及创建者、创建日期版本、包名、类的功能等一些信息:

/\*\*

\* @className: 类名

\* @description: 简单描述类的功能

\* @createAuthor: 作者

\* @createDate: 创建该类日期时间，如：2020-07-21 10:20

\* @updateAuthor:修改者

\* @updateDate: 创建该类日期时间，如：2020-07-21 10:20

\*/

## 方法注释(Methods)

方法注释(Methods)主要用来声明该类的作用、入参、返回值、异常等信息:

/\*\*

\* @author: fendo

\* @methodsName: addUser

\* @description: 添加一个用户

\* @param: xxxx

\* @return: string

\*/

## 文件注释

文件注释包括作者，该文件功能和日期时间描述

/\*\*

\* @author: fendo

\* @description: 描述该文件的功能

\* @date: 创建该文件日期时间，如：2020-07-21 10:20

\*/

## 全局变量的注释

全局变量的功能，取值范围。