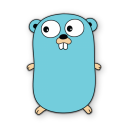
**Go 语言教程**



Go 是一个开源的编程语言，它能让构造简单、可靠且高效的软件变得容易。

Go是从2007年末由Robert Griesemer, Rob Pike, Ken Thompson主持开发，后来还加入了Ian Lance Taylor, Russ Cox等人，并最终于2009年11月开源，在2012年早些时候发布了Go 1稳定版本。现在Go的开发已经是完全开放的，并且拥有一个活跃的社区。

**Go 语言特色**

* 简洁、快速、安全
* 并行、有趣、开源
* 内存管理、v数组安全、编译迅速

**Go 语言用途**

Go 语言被设计成一门应用于搭载 Web 服务器，存储集群或类似用途的巨型中央服务器的系统编程语言。

对于高性能分布式系统领域而言，Go 语言无疑比大多数其它语言有着更高的开发效率。它提供了海量并行的支持，这对于游戏服务端的开发而言是再好不过了。

**第一个 Go 程序**

接下来我们来编写第一个 Go 程序 hello.go（Go 语言源文件的扩展是 .go），代码如下：

**实例**

package main  
import "fmt"  
func main() {  
    fmt.Println("Hello, World!")  
}

执行以上代码输出

$ go run hello.go

Hello, World!

# Go 语言环境安装

Go 语言支持以下系统：

* Linux
* FreeBSD
* Mac OS X（也称为 Darwin）
* Window

安装包下载地址为：<https://golang.org/dl/>。

各个系统对应的包名：

|  |  |
| --- | --- |
| **操作系统** | **包名** |
| Windows | go1.4.windows-amd64.msi |
| Linux | go1.4.linux-amd64.tar.gz |
| Mac | go1.4.darwin-amd64-osx10.8.pkg |
| FreeBSD | go1.4.freebsd-amd64.tar.gz |

## UNIX/Linux/Mac OS X, 和 FreeBSD 安装

以下介绍了在UNIX/Linux/Mac OS X, 和 FreeBSD系统下使用源码安装方法：

1、下载源码包：go1.4.linux-amd64.tar.gz。

2、将下载的源码包解压至 /usr/local目录。

tar -C /usr/local -xzf go1.4.linux-amd64.tar.gz

3、将 /usr/local/go/bin 目录添加至PATH环境变量：

export PATH=$PATH:/usr/local/go/bin

注意：MAC 系统下你可以使用 .pkg 结尾的安装包直接双击来完成安装，安装目录在 /usr/local/go/ 下。

## Windows 系统下安装

Windows 下可以使用 .msi 后缀(在下载列表中可以找到该文件，如go1.4.2.windows-amd64.msi)的安装包来安装。

默认情况下.msi文件会安装在 c:\Go 目录下。你可以将 c:\Go\bin 目录添加到 PATH 环境变量中。添加后你需要重启命令窗口才能生效。

### 安装测试

创建工作目录 C:\>Go\_WorkSpace。

文件名: test.go，代码如下：

package main

import "fmt"

func main() {

fmt.Println("Hello, World!")

}

使用 go 命令执行以上代码输出结果如下：

C:\Go\_WorkSpace>go run test.go

Hello, World!

**Go 语言结构**

在我们开始学习 Go 编程语言的基础构建模块前，让我们先来了解 Go 语言最简单程序的结构。

**Go Hello World 实例**

Go 语言的基础组成有以下几个部分：

* 包声明
* 引入包
* 函数
* 变量
* 语句 & 表达式
* 注释

接下来让我们来看下简单的代码，该代码输出了"Hello World!":

package main

import "fmt"

func main() {

/\* 这是我的第一个简单的程序 \*/

fmt.Println("Hello, World!")

}

让我们来看下以上程序的各个部分：

1. 第一行代码 *package main* 定义了包名。你必须在源文件中非注释的第一行指明这个文件属于哪个包，如：package main。package main表示一个可独立执行的程序，每个 Go 应用程序都包含一个名为 main 的包。
2. 下一行 *import "fmt"* 告诉 Go 编译器这个程序需要使用 fmt 包（的函数，或其他元素），fmt 包实现了格式化 IO（输入/输出）的函数。
3. 下一行 *func main()* 是程序开始执行的函数。main 函数是每一个可执行程序所必须包含的，一般来说都是在启动后第一个执行的函数（如果有 init() 函数则会先执行该函数）。

//===================20171104

1. 下一行 /\*...\*/ 是注释，在程序执行时将被忽略。单行注释是最常见的注释形式，你可以在任何地方使用以 // 开头的单行注释。多行注释也叫块注释，均已以 /\* 开头，并以 \*/ 结尾，且不可以嵌套使用，多行注释一般用于包的文档描述或注释成块的代码片段。
2. 下一行 *fmt.Println(...)* 可以将字符串输出到控制台，并在最后自动增加换行字符 \n。   
   使用 fmt.Print("hello, world\n") 可以得到相同的结果。   
   Print 和 Println 这两个函数也支持使用变量，如：fmt.Println(arr)。如果没有特别指定，它们会以默认的打印格式将变量 arr 输出到控制台。
3. 当标识符（包括常量、变量、类型、函数名、结构字段等等）以一个大写字母开头，如：Group1，那么使用这种形式的标识符的对象就可以被外部包的代码所使用（客户端程序需要先导入这个包），这被称为导出（像面向对象语言中的 public）；标识符如果以小写字母开头，则对包外是不可见的，但是他们在整个包的内部是可见并且可用的（像面向对象语言中的 protected ）。

**执行 Go 程序**

让我们来看下如何编写 Go 代码并执行它。步骤如下：

1. 打开编辑器如Sublime2，将以上代码添加到编辑器中。
2. 将以上代码保存为 *hello.go*
3. 打开命令行，并进入程序文件保存的目录中。
4. 输入命令 *go run hello.go* 并按回车执行代码。
5. 如果操作正确你将在屏幕上看到 *"Hello World!"* 字样的输出。

$ go run hello.go

Hello, World!