Habits

# 环境搭建

[官网下载](http://code.google.com/p/go/downloads/list)*可能被墙了*  
[golang中国下载](http://golangtc.com/) 选择自己想要的版本进行安装

## 安装

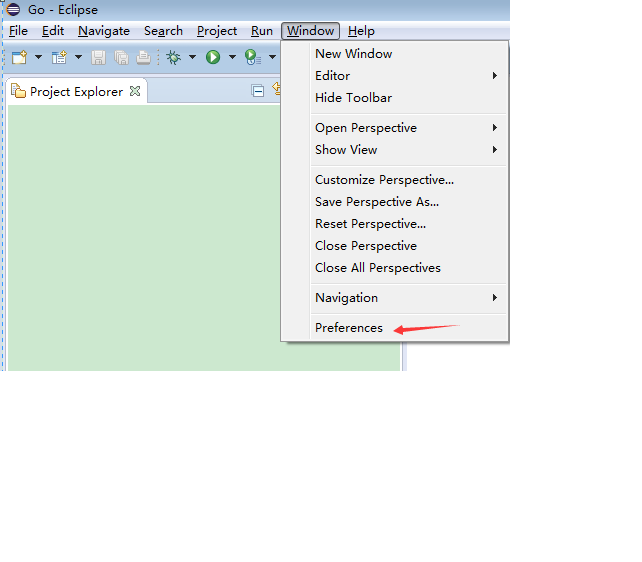
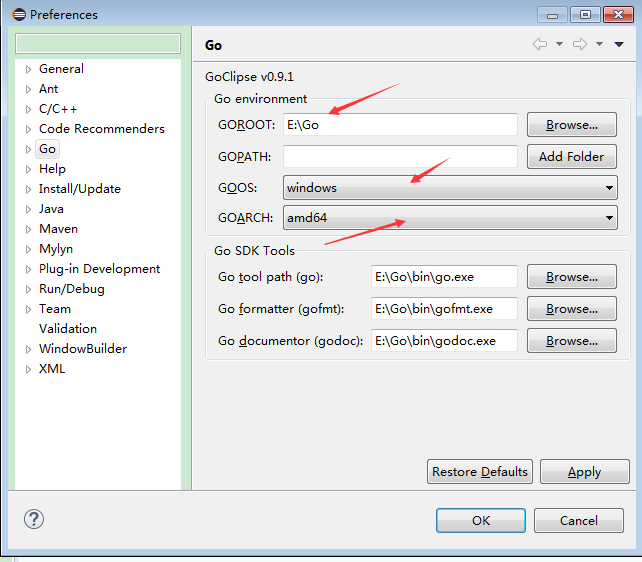
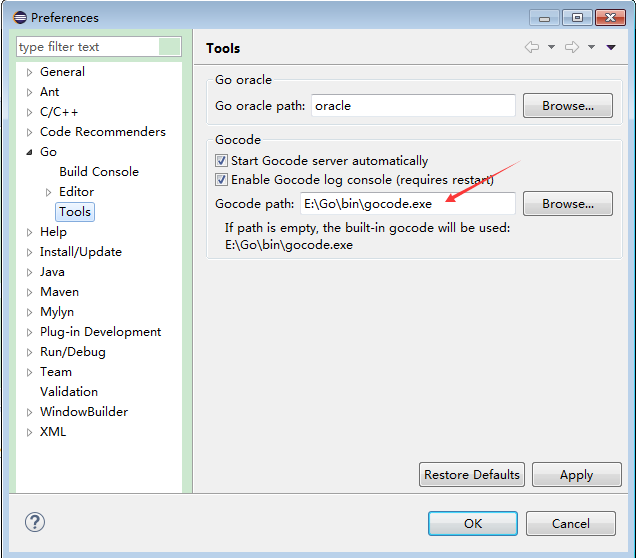
在linux环境中Go默认会被安装到/usr/local/go目录中。安装包在安装完成后会自动添加执行 文件目录到系统路径中。 安装完成后，请重新启动命令行程序，然后运行下命令go version以验证Go是否已经正确安装

### eclipse的golang插件

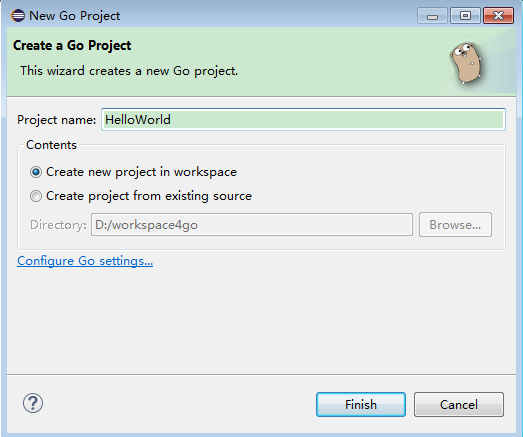
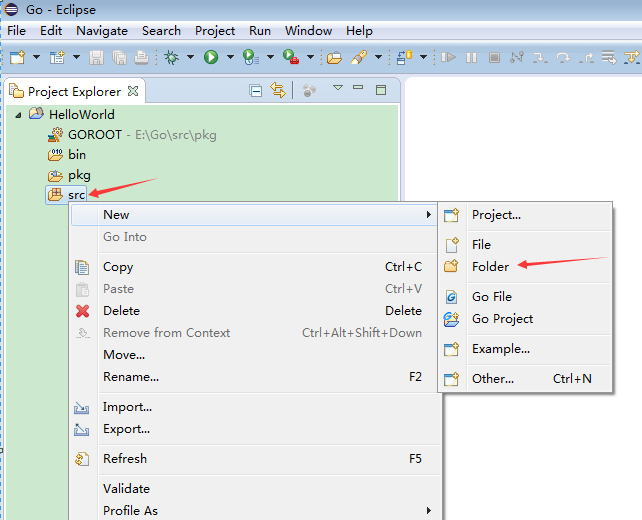
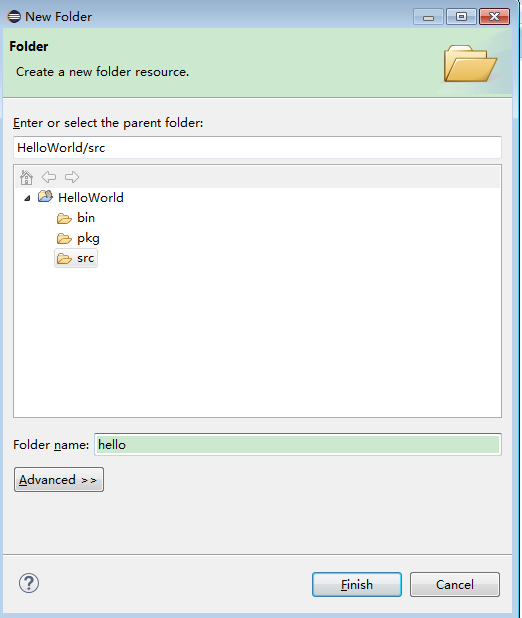
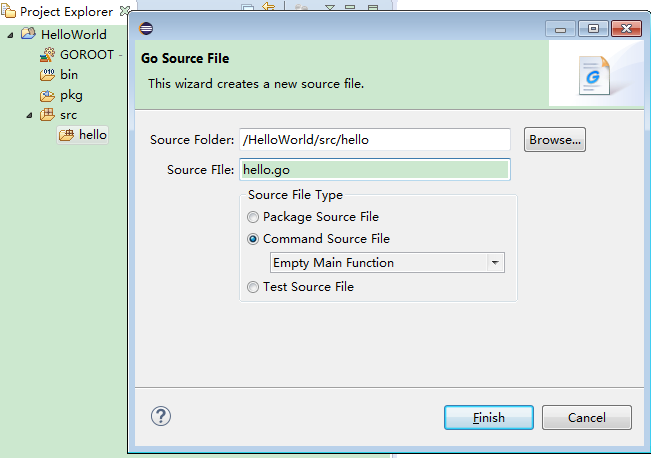
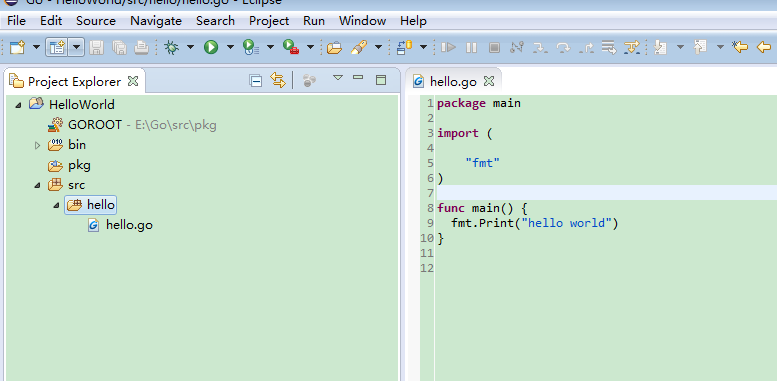
Installation  
  
Requirements:  
  
Eclipse 4.4 (Luna) or later.  
A 1.7 Java VM or later (http://www.java.com/). Otherwise GoClipse will silently fail to start.  
Go development tools and libraries installed on target machine (http://golang.org/doc/install.html). Currently, only the plan 9 style compilers are supported. GCCGO may be supported in the future  
Instructions:  
Use your existing Eclipse, or download a new Eclipse package from http://www.eclipse.org/downloads/.  
For an Eclipse package without any other IDEs or extras (such a VCS tools), download the "Platform Runtime Binary".  
Start Eclipse, go to Help -> Install New Software...  
Click the Add... button, then enter the Update Site URL: http://goclipse.github.io/releases/ in the Location field, click OK.  
Select the recently added update site in the Work with: dropdown. Type GoClipse in the filter box. Now the Goclipse feature should appear below.  
Select the GoClipse feature, and complete the wizard.  
Dependencies such as CDT will automatically be added during installation.  
Restart Eclipse. After that take a look at the configuration section in the User Guide.  
Updating:  
  
If you already have GoClipse installed, and want to update it to a newer release, click Help -> Check for Updates...

[详情参见](https://github.com/GoClipse/goclipse/blob/master/documentation/Installation.md)

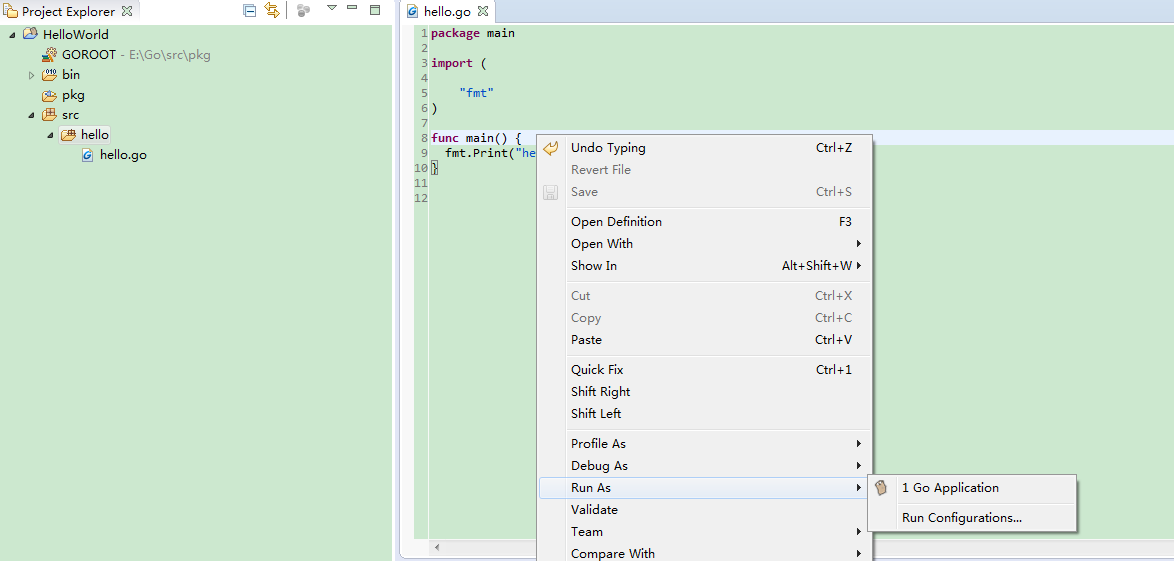
### GoClipse配置

1. 首先需要下载gocode 。  
   >go get -u github.com/nsf/gocode (-u flag for "update") >windows使用 go get -u -ldflags -H=windowsgui github.com/nsf/gocode
2. 配置golang
3. 配置gocode（这个是编写代码时候的自动补全功能）

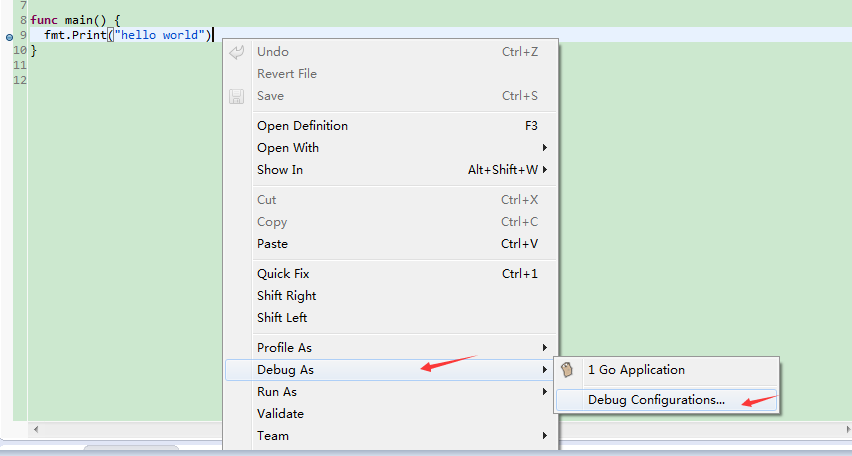
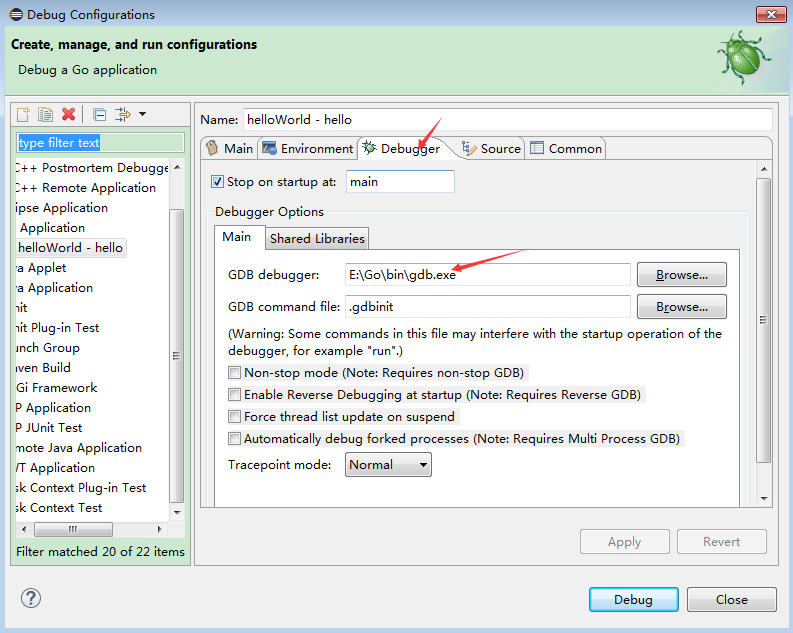
## 经典实例HelloWorld

1.新建工程   
2.在src目录下新建一个文件**在src建立go文件是无法被处理的**   
3.  
4.新建一个go文件  
5.  
6.

package main   
import (  
 "fmt"  
)  
func main() {  
 fmt.Print("hello world")  
}

7.运行结果

### debug

 下载gdb 

# 变量

## 变量定义

使用关键字var 定义变量,自动初始化为零值.如果提供初始化值，可省略变量类型，由编译器自动推导。

var v1 int   
var v2 string   
var v3 [10]int // 数组   
var v4 []int // 数组切片   
var v5 struct {   
 f int   
}   
var v6 \*int // 指针   
var v7 map[string] int // map，key为string类型，value为int类型   
var v8 func(a int) int   
var v9 int = 10  
var v10 = 10 //编译器可以自动推导出v10的类型   
v11 := 10 //编译器可以自动推导出v11的类型

## 多变量定义

var v1, v2, v3 int  
var v4, v5 = "abc", 123

## 将变量放到一起

不能有“:=”的声明方式在里面

var (  
 v1 int  
 v2 string  
 v3 = 10  
)

## 变量赋值

多变量赋值时,从左到右依次赋值

var v4, v5 = "abc", 123

## golang多重赋值功能

如下面这个交换v1和v2变量的语句：

v1, v2 = v2, v1

不支持多重赋值功能的代码的语句

v3=v1,v1=v2,v2=v3

## 匿名变量"\_"用于忽略占位

会报错的代码

package main  
import (  
 "fmt"  
)  
func test() (int, string) {  
 return 1, "rv1"  
}  
func main() {  
 var v1 int //会报错： v1 declared and not used  
 var v2 int  
 fmt.Println(v2)  
}

使用“\_”占位屏蔽报错

package main  
import (  
 "fmt"  
)  
func test() (int, string) {  
 return 1, "rv1"  
}  
func main() {  
 var v1 int  
 \_=v1 //使用匿名变量规避 “v1 declared and not used”的报错  
 var v2 int  
 fmt.Println(v2)  
}

## 常量

定义:编译期可确定的数字、字符串、布尔值

const v1 int=1  
const v2 = 2 // 无类型整型常量  
const ( v3 int = 3 v4 = -1 // 无类型整型常量 )  
const v5, v6 float32 = 0, 3 // v5 = 0.0, v6 = 3.0， const v7, v8, v9 = 7, 8, "9str" const v10=1<<2 //在编译期运算的常量表达式 const v11=os.Geteuid()//报错const initializer os.Geteuid() is not a constant**常量的赋值是一个编译期行为， 所以右值不能出现任何需要运行期才能得出结果的表达式**

在常量组中，如不提供类型和初始化值，那么视作与上一常量相同

const (  
 v1 = "v1"  
 v2 // v2 = "v1"  
)

## 预定义常量iota

定义:是一个可被编译器修改的常量，在每一个const关键字出现时被重置为0，然后在下一个const出现之前，每出现一次iota，其所代表的数字会自动增1。

const (//const出现iota被重置为0  
 v0 = iota //v0=0  
 v1 = iota //v1=1  
 v2 = iota //v2=2  
)  
const v3 = iota //const出现iota被重置为0 v3=0  
const ( //const出现iota被重置为0  
 v4 = iota //v4=0  
 v5 //v5=1  
 v6 //v6=2  
)

在同一常量组中，可以提供多个iota，它们各自增长

const (  
 v1, v2 = iota, iota+1 // v1=0, v2=1  
 v3, v4 // v3=1,v4=2  
)

如果iota被中断必须要显示回复

const (  
 v1 = iota // v1=0  
 v2 // v2=1  
 v3 = "v3" // v3="v3"   
 v4 // v4="v3"，与上一行相同。  
 v5 = iota // v5=4，显式恢复。注意计数包含了 v3,v4两行。  
 v6 // v6=5  
)

## 枚举

定义:枚举是指指一系列相关的常量，比如性别

const(  
 Male = iota   
 Female  
)

自定义类型来实现枚举类型限制

package main  
import(  
 "fmt")  
type Sex int  
const(  
 Male Sex = iota   
 Female  
 )  
func printSex(s Sex){  
 fmt.Println(s)  
 }  
func main() {  
 v1 := Male  
 v2 := Female  
 printSex(v1)  
 printSex(v2)  
 printSex(3)//常量会被编译器自动转换  
 v3 :=1  
 printSex(v3)//Error:cannot use v3 (type int) as type Sex in argument to printSex  
}

# 常见错误

## go: missing Git command

处理办法，把git的bin路径加入环境变量Path中。 我的路径是：C:Files (x86)