一、Go发展

诞生

一切要从Ken Thompson说起。

1966年，Ken Thompson就职于贝尔实验室，参与了MULTICS项目，独自开发了B语言，并利用一个月时间用B语言开发了一个精简的操作系统，起名UNICS。后来，同事Dennis Ritchie基于B语言发明了C语言。他们一起用C重写了Unix系统。

Rob Pike是贝尔实验室Unix组成员，也是《Unix环境编程》和《程序设计实践》作者之一。1992年和Ken Thompson共同设计了UTF-8编码。2002年加入Google，研究操作系统。

2006年，Ken Thompson加入Google。

2007年9月20日，身处Google的Rob Pike对不断扩充新特性的C++非常不满，与Robert Griesemer(之前参与JavaScript V8引擎和Java HotSpot虚拟机)、Ken Thompson进行了一次关于设计一门新语言的讨论。第二天，三人继续进行了对新语言设计的讨论会，并在会后有Robert Griesemer发出了一封邮件，总结了设计思路：要在C语言基础上，修正一些明显缺陷，删除一些被诟病的特征，增加一些缺失的功能。

上图为2012年采访时的Go语言之父：Robert Griesemer、Rob Pike、Ken Thompson

2007年9月25日，Rob Pike在一封回复邮件中把新语言命名为Go。

2009年10月30日，Go语言首次公之于众。

2009年11月10日正式开源，这一天被Go官方确定为**Go语言诞生日**。

Go语言也拥有了自己的吉祥物（Rob Pike夫人Renee French设计的地鼠）。Go程序员被称为**Gopher**。

版本

2012年3月28日，Go 1.0正式发布。

2015年8月19日，Go 1.5这个里程碑版本发布。

- Go不在依赖C编译器，Go编译器和运行时都使用Go代码了，实现了自举
- GOMAXPROCS的默认值1改为运行环境的CPU核数

2018年8月25日，Go1.11版本发布。

- 引入Go Module包管理机制

2021年2月18日，Go 1.16版本发布。

- Go Module-aware模式成为默认模式，即GO111MODULE默认从auto改为on
- go build/run命令不再自动更新go.mod和go.sum文件

2022年3月15日，Go 1.18版本发布

- 引入了泛型Generic
- 支持工作区Workspace

设计理念

其它语言的弊端

- 对多核CPU缺乏支持。很多语言诞生过早，还停留在单核CPU时代，未对多核CPU并行做优化
- C语言等原生语言缺乏好的依赖管理（C语言依赖头文件）
- Java、C++等语言过于笨重。很小一个需求，也需要很多代码，甚至需要引入很重的框架
- C、C++没有提供垃圾回收，这对很多程序员来说并不友好

Go设计者推崇"最小方式"思维，即一件事仅有一种方式，或尽可能少的方式去完成。目的是减少开发人员选择的痛苦，减少理解别人选择的苦恼。

- 仅有25个关键字，简洁的语法
- 内置垃圾回收器，大大降低程序员管理内存的负担
- **去除隐式类型转换**、**去除指针算数**，提高语言健壮性
- 首字母大小写决定可见性，通过约定而不是声明告诉开发者，提高阅读效率
- 故意**不支持函数默认参数**，必须明确每个参数意义，提高设计的清晰度和代码可读性
- 没有面向对象的类，也没有子类，没有构造函数和析构函数
- 偏好`组合`
- 任何类型都可以拥有方法
- 接口只是方法的集合，其实现是**隐式**的（无需implements显示声明）
- n--、n++都是语句，不是表达式。**没有**--n、++n
- **没有**三元运算符
- 内存总是初始化为零值
- 没有异常
- 内置字符串、切片、map等类型
- 内置并发支持，对多核计算机支持友好。goroutine、channel、select就是为并发而生的
- 官方提供了丰富的工具链，涵盖了编译、编辑、依赖获取、调试、文档、性能分析等诸多方面

二、环境与开发

官网 https://go.dev/ 或 https://golang.google.cn/

目前稳定版本是1.18.x和1.17.x。

注：对于所有语言，我们不追求最新版本。新版会引进新特性、新功能，这些都可能导致Bug，而且项目中使用的第三方库未必兼容。因此，使用最新版本有很大风险。如果尝鲜可以试装。

安装

安装文档 https://golang.google.cn/doc/install https://go.dev/doc/install

从 https://go.dev/dl/ 或 https://golang.google.cn/dl/ 下载对应操作系统平台指定的版本即可。

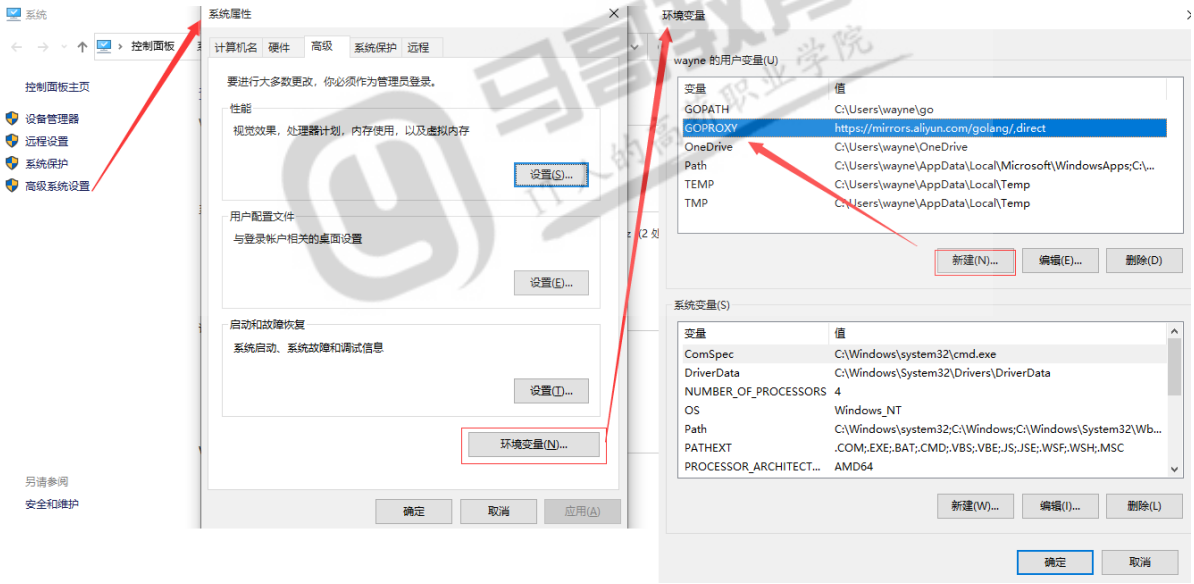
Windows安装

官网下载 go1.x.y.windows-amd64.msi ， 按照安装向导安装。

```
>go version
go version go1.18.7 windows/amd64
```

添加环境变量

如果是当前用户使用，就是用户环境变量；如果是系统所有用户使用，就是系统变量，需要管理员权限



观察环境变量PATH、GOPATH，新建GOPROXY。

可以运行命令go env看看Go的环境变量。

Linux安装

解压

```
# tar xf go1.18.7.linux-amd64.tar.gz -C /usr/local
```

添加到环境变量PATH，修改$HOME/.profile或/etc/profile

```
export PATH=$PATH:/usr/local/go/bin
export GOPROXY=https://mirrors.aliyun.com/golang/,direct
```

环境变量

- GOROOT：go安装目录
- PATH：环境变量PATH添加GOROOT下的bin目录，go可执行文件所在目录
- GOPATH 第三方依赖包安装路径
  - 例如我本机默认GOPATH=~\go
  - 以前放自己开发的代码，目前不要这样放了
  - GOMODCACHE存储第三方依赖包
- GO111MODULE=on
  - go早期版本管理依赖包方式使用GOPATH和vendor文件夹
  - 从1.11引入Go Module，1.16版本开始GO111MODULE默认为on，会根据go.mod管理和下载依赖
- **GOPROXY** 1.13开始支持。国外站点经常失败，建议改为国内镜像站地址
  - 参考 https://go.dev/ref/mod#environment-variables
  - GOPROXY 缺省是 `https://proxy.golang.org,direct`
    - direct 表示从镜像下载失败后，直接从原版本库地址下载
    - off 表示不允许从任何源下载
  - https://mirrors.aliyun.com/goproxy/ 、 https://mirrors.huaweicloud.com/home
  - ```
    GOPROXY=https://mirrors.aliyun.com/goproxy/,direct
    GOPROXY=https://repo.huaweicloud.com/repository/goproxy/
    一定要配置，不然后面安装包经常下载包很困难
    ```

Windows主要环境变量，如下

```
>go env
set GO111MODULE=
set GOARCH=amd64
set GOENV=C:\Users\wayne\AppData\Roaming\go\env
set GOMODCACHE=C:\Users\wayne\go\pkg\mod
set GOOS=windows
set GOPATH=C:\Users\wayne\go
set GOPROXY=https://mirrors.aliyun.com/golang/,direct
set GOROOT=C:\Program Files\Go
set GOTOOLDIR=C:\Program Files\Go\pkg\tool\windows_amd64
set GOVERSION=go1.18.7

Windows安装版安装完PATH路径就添加了
>echo %PATH%
C:\Program Files\Go\bin;C:\Users\wayne\go\bin
```

Linux主要环境变量，如下

```
# go env
GO111MODULE=""
```
```

```
GOARCH="amd64"
GOENV="/root/.config/go/env"
GOMODCACHE="/root/go/pkg/mod"
GOOS="linux"
GOPATH="/root/go"
GOPROXY="https://mirrors.aliyun.com/golang/,direct"
GOROOT="/usr/local/go"
GOTOOLDIR="/usr/local/go/pkg/tool/linux_amd64"
GOVERSION="go1.18.7"

# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/local/go/bin
```

开发工具

> 好用的工具能提升生产效率，但不要迷信工具，写出好的代码才是王道

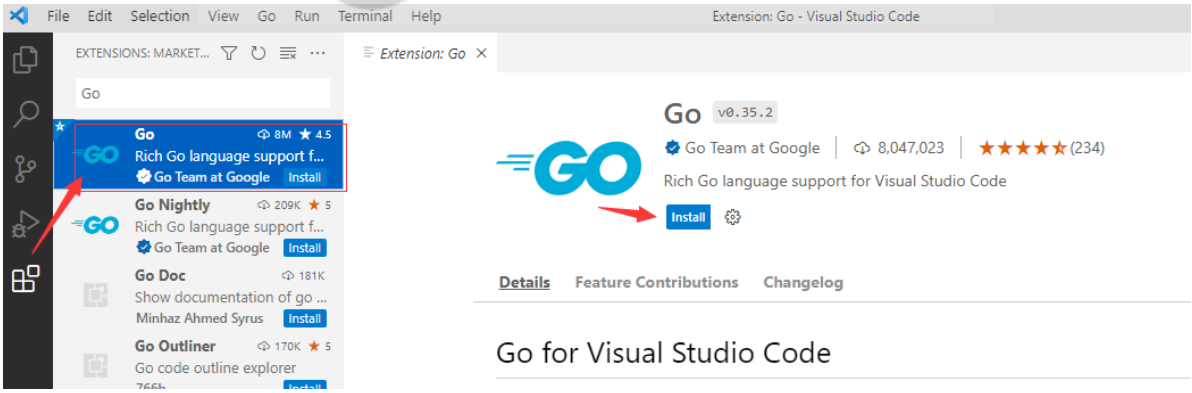
- Goland 收费工具
- VSCode 微软推出的轻量级、插件化、多语言开发IDE。课堂教学使用
  - Windows、MAC、Debian、RedHat
  - Windows User版是当前用户一人使用；System版当前机器所有用户使用

VSCode插件安装

切记，安装插件之前，一定要配置GO国内源的GOPROXY，否则可能报很多错误。

Go三件套

1、Go插件



2、gopls

gopls（发音Go please）是官方提供的模块，实现语言服务器协议LSP的Language Server，具有构建、格式化、自动完成等功能。

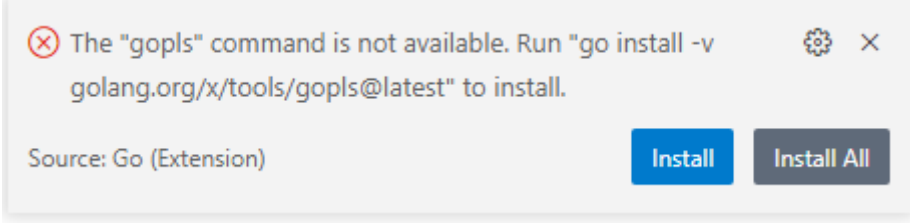
LSP是由微软开发的开发标准，它为便捷器提供对语言特性的支持。

当你使用模块和 VSCode 时，在编辑器中点击保存将不再直接运行 go build 命令。现在发生的情况是，一个请求被发送给 gopls，gopls 运行适当的 Go 命令和相关的 API 来提供编辑器反馈和支持。

参考 https://github.com/golang/tools/blob/master/gopls/README.md

可以手动安装，也可以在VSCode中编辑第一个xxx.go文件触发安装。

可以点击Install单独安装，也可以点击Install All连dlv等一起安装。



手动安装

```
$ go install golang.org/x/tools/gopls@latest
```

```
Tools environment: GOPATH=C:\Users\wayne\go
Installing 1 tool at C:\Users\wayne\go\bin in module mode.
  gopls

Installing golang.org/x/tools/gopls@latest (C:\Users\wayne\go\bin\gopls.exe) SUCCEEDED

All tools successfully installed. You are ready to Go. :)
```

配置

参考 https://github.com/golang/tools/blob/master/gopls/doc/settings.md

打开File/Preference/Settings(Ctrl +,)，搜索gopls，如下例配置

```
{
    "gopls": {
        "ui.completion.usePlaceholders": true
    }
}
```

3、dlv

go-delve是Go语言的调试利器。

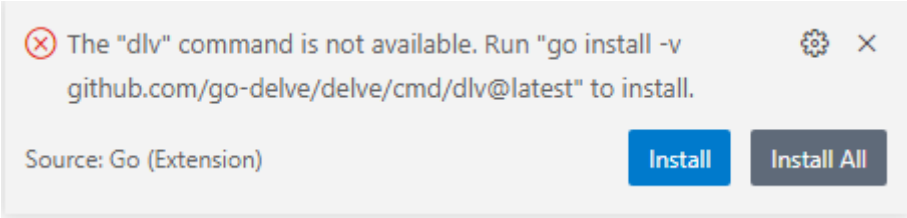
编写一个main.go，输入pkgm自动补全

```
package main

import "fmt"

func main() {
    fmt.Println("Hello magedu.com")
}
```

按F5启动调试，会出现安装dlv的提示。同样，可以手动安装，也可以根据提示安装。




```
Tools environment: GOPATH=C:\Users\wayne\go
Installing 1 tool at C:\Users\wayne\go\bin in module mode.
  dlv

Installing github.com/go-delve/delve/cmd/dlv@latest
(C:\Users\wayne\go\bin\dlv.exe) SUCCEEDED

All tools successfully installed. You are ready to Go. :)
```

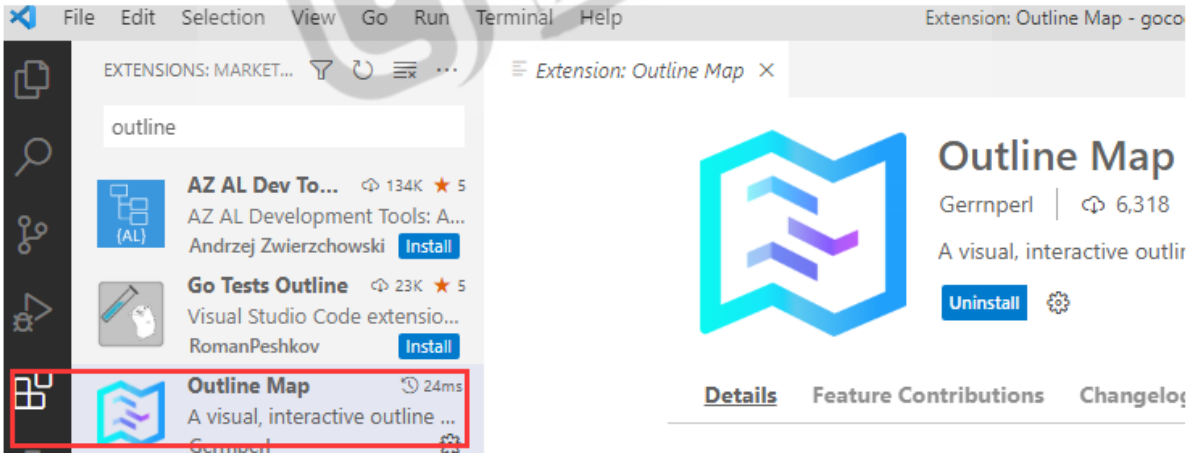
至此，运行Go所需的go、gopls、dlv

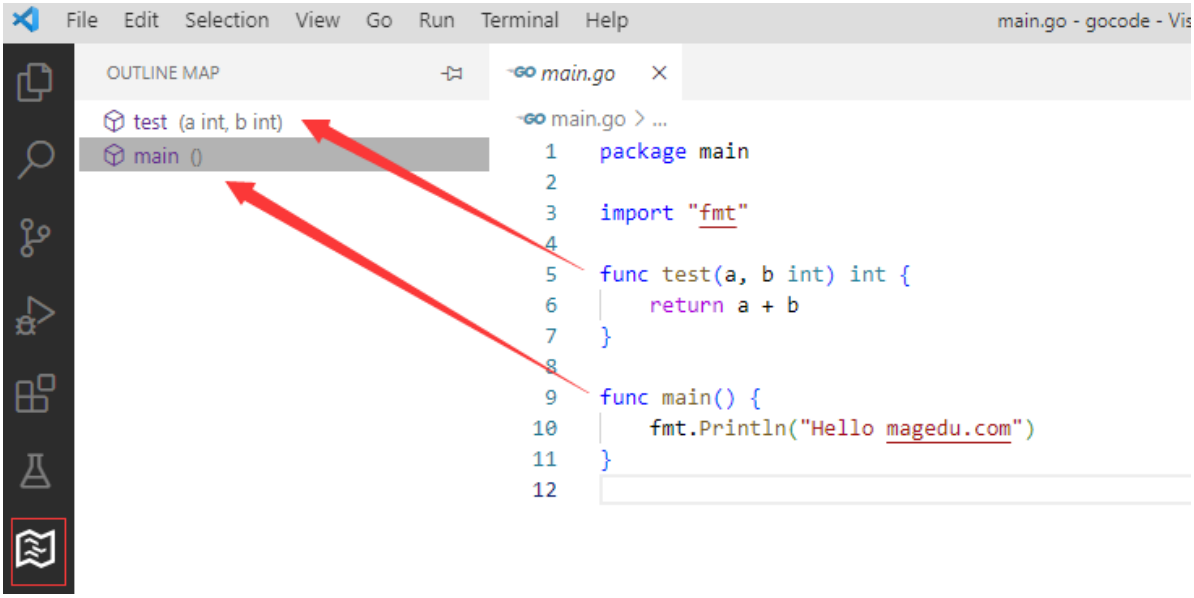
命令行中运行

```
$ go run ./main.go
Hello magedu.com
```

大纲

这个VSCode插件安装后，可以方便的阅读Go代码，尤其是源代码中有大量函数、结构体、接口时。





也可以安装Golang postfix这样的工具，提高编码效率。

Ubuntu

确保已经安装了Git。`sudo apt install git`

可以在软件市场里面搜索到VSCode，直接安装即可。

初始Go程序

在VSCode中打开一个目录，其中创建一个main.go。

main函数是整个程序的运行入口，必须是main包。

```go
package main

import "fmt"

func main() {
    fmt.Println("Hello magedu.com")
}
```

命令行中运行

```
$ go run main.go
```

也可以利用安装的Go三件套调试，按F5运行，可能有如下提示

```
go: go.mod file not found in current directory or any parent directory
```

只需要命令行中初始化模块即可。关于工程化开发我们后面再讲。

```
$ go mod init magedu.com/test
```

在按F5就可以调试运行了。

Go命令

使用`go help <command>`查看帮助

语言参考 https://golang.google.cn/doc/ 或 https://go.dev/doc/

version

查看版本

env

查看环境变量

mod

Go Module工程化模块相关命令。模块化后就可以进行更好的工程管理、依赖管理。

```
$ go mod init magedu.com/test    初始化模块目录，产生go.mod文件，和git init等很类似
$ go mod tidy                    扫描当前模块，从go.mod中移除未使用依赖，添加新的依赖
```

get

查找依赖包，如果未下载，则下载项目依赖包，添加相关信息到go.mod中，文件存储到环境变量`GOMODCACHE`指定的目录中（默认GOPATH/pkg/mod）。

-u 下载最新包或有效更新包。默认情况下，本地有包就不下载了。如果网上有最新版，-u选项会更新本地包。

install

除了具备get的功能外，如果需要编译包，就执行go install，并把编译好的可执行文件放到GOPATH/bin下。例如

```
$ go install golang.org/x/tools/gopls@latest
```

build

编译Go代码，生成可执行文件。

-o 指定生成的可执行文件。

```
go build [-o output] [build flags] [packages]

$ go build
$ go build magedu.com/test
$ go build main.go
$ > go build -o hello.exe main.go
```

run

编译生成可执行文件并执行。

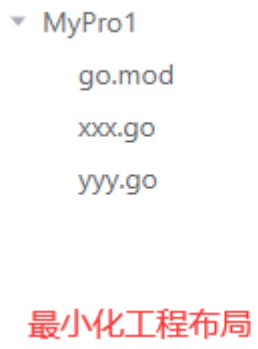
```
$ go run main.go            指定main函数所在文件
$ go run magedu.com/test    去该模块找main函数
```

fmt

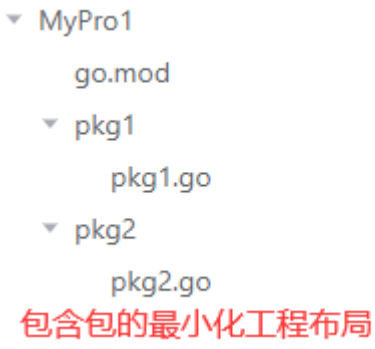
对代码格式化。已经安装的gopls已经支持gofmt、goimports等，在VSCode中会自动格式化代码，使得代码符合Go语言代码风格。

三、项目结构

Go官方并没有项目结构布局方案，因此经过实践，大家提出了各种可行方案。现阶段，我们学习的项目规模较小，可采用下面的方案



最小化工程布局



包含包的最小化工程布局

马哥教育
IT人的高薪职业学院