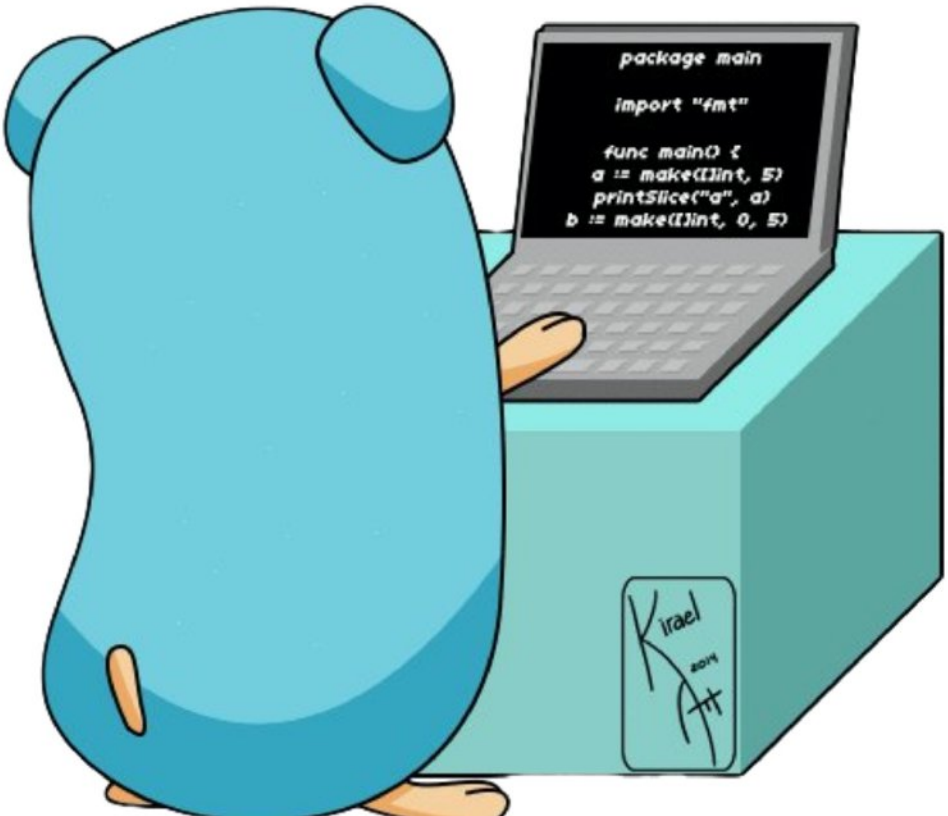
极客时间 | 训练营

Go工程师
零基础就业班

CONTENTS

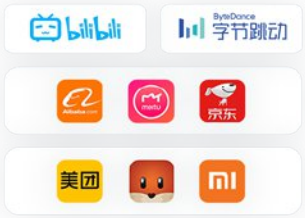


为什么你应该学习 Go 语言?

掌握云原生时代首选编程语言

01 Go 简单易学，性能好
已经成为新一代的企业级编程语言

02 一线大厂在大规模使用 Go 开发
创业公司也深爱 Go 语言



03 绝大多数的云原生项目都由 Go 语言编写
在云计算、金融科技等行业应用广泛

选择有前景的编程语言，拓宽你的未来职业赛道

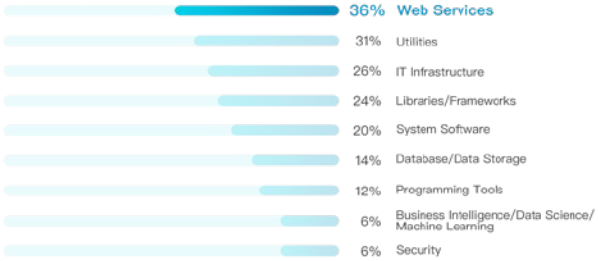
平均年薪逐年增高，起薪高、前景好

Go 开发工程师常位于企业核心技术部门

· 数据取自职友集 2021 年 8 月 30 日数据

岗位需求逐年增多，竞争压力更小

Go 语言的 TOP5 应用领域为 Web 服务、工具类应用、IT 基础设施、各类框架/库以及系统软件。



· 数据来源：JetBrains 官方调查

上手快、性能高，Go 语言非常适合刚开始编程的你

01 Go 语言出身“名门” Google，“血统”纯正

02 Go 语言语法简单规范，上手速度飞快

03 Go 语言编译时间很短，运行效率极高

04 Go 语言稳定性强，拥有强大的编译检查和严格的编码规范

谁适合学习 Go 语言?

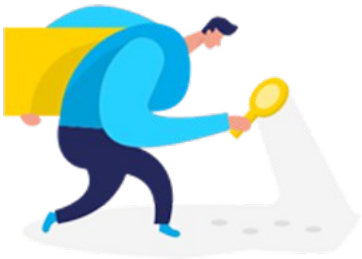
如果你属于以下情况，选择这门课程就对了

即将就业的准应届生

- 想拿大厂 Offer，想从同龄人中脱颖而出

缺实战的编程爱好者

- 热爱编程，想钻研技术却没有大佬带入门



想加薪的各类工程师

- 基础差，想系统学习快速成长为技术骨干

零基础传统行业从业者

- 转行人士，尝试新赛道，需要一个敲门砖

为什么要在极客时间学 Go

极客时间 App 是极客邦科技出品的一个 IT 知识服务平台，我们利用技术和模式，在新场景下提供快捷、方便、经济和丰富的学习方式。产品包含专栏、训练营、新闻、每日一课等板块，内容覆盖计算机从业者的全周期知识技能图谱。极客时间为技术专家和读者用户建立了高效的连接，解决了知识领域内的信息不对称等问题，降低了学习梯度，增加信息密度，让所有的用户高效学习、轻松学习。

累计服务个人用户

120+万人

程序员认可的
学习平台

累计服务企业

2000+家

企业选择的
员工技能提升平台

聚合全球一线技术专家

3000+位

国内外技术专家
实战经验锤炼

线上优质课程

1000+门

专业教研团队身经百战
交付优质课程

举办技术会议

50+场

洞察行业前沿技术
发展方向

极客时间如何带你成为 Golang 工程师

师资介绍

来自一线国内外互联网大厂的讲师，了解行业前沿实战案例，深知企业用人需求

教学+实战经验双丰富

讲师：李建强

eBay 资深软件工程师

- 他拥有 10 多年的一线研发经验，先后就职于 VMware 和 eBay，最近 6 年一直在使用 Golang 进行项目设计与开发工作，精通 Go 语言及相关开源项目。
- 他拥有企业级的研发项目管理经验，近年来专注于云原生领域，深刻洞悉 Go 语言的应用与发展前景。
- 他乐于分享，擅长教学，有多年企业内部的技术培训经验，同时也是《Kubernetes 生产化实践之路》的作者之一。

课程能力模型设计

以大厂能力模型为标准，用科学的学习路径带你实现能力提升

全方位系统的学习服务

教、学、练、评和就业推荐，为学习效果保驾护航

学前充分预习
充分做好
上课准备

- **办理入学**：添加班主任
- **学前预习**：提前预习 夯实基础

双师闭环教学
讲师 + 领教 + 助教
三师带你 学有所成

- **主讲直播教学**：每周 5 小时 直播教学
- **助教 1v1 答疑**：1v1 针对服务 细微疑问 不留缺漏
- **班主任全程伴学**：组织班级活动 活跃学习氛围
- **作业批改**：学习效果有保障 举一反三易提高

学习提效服务
学习活动 + 工具
提升学习效率

- **期末项目**：检测自己的学习成果，并可加入简历
- **社群联结**：联结行业菁英 共同学习进步
- **配套简历辅导**：毕业后简历辅导 帮你避坑

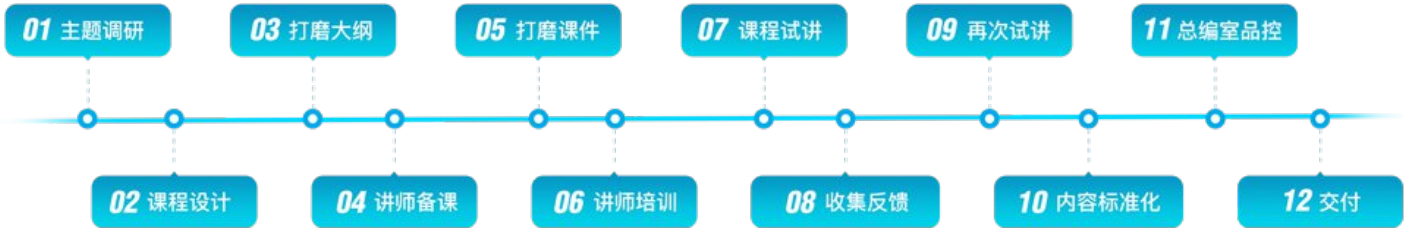
就业个性推荐
高薪就业
才是硬道理

- **毕业证书**：展示学习成果
- **企业内推**：结业一年以内 两次免费内推

奖学金计划
优秀学员
充分肯定

- **优秀学员福利**：结业优秀学员 专享大额奖金

12 道工序精心打磨，致力于交付高质量课程



课程大纲

模块一：Go 语言基础语法

教学目标：

1. 掌握 Go 语言基础语法
2. 掌握 Go 语言常用操作符
3. 掌握 Go 语言的控制结构

学习和工作中的痛点：

1. 编程环境搭建困难，不知道如何配置开发环境
2. 不了解 Go 的基础语法及各类字符的含义
3. 不理解程序的实现逻辑，无法快速上手实践

详细内容：

1. 搭建 Go 语言的编程环境

1) Visual Studio Code + core plugins 配置
2) outliner plugins 配置
3) Windows 和 macOS 系统下的环境变量配置
4) 测试安装
5) 实战案例
   A. 写下你的第一段代码：Hello，Go 语言
   B. 用 Go 程序打印一首诗
   C. 用 Go 程序画一颗五角星

2. Go 语言的基础语法

1) 基础数据类型
   A. int、float
   B. string
   C. bool

2) 变量、常量

A. 什么是变量，什么是常量?
B. 如何定义变量?
C. 默认值
D. 变量类型推断
E. 变量使用规则
F. 实战案例
- 计算圆面积并输出结果（要求有常量）
- 计算两个坐标点之间的距离（需要查询 math 开根号函数）

3) 操作符

A. 算数操作符：加、减、乘、除、取余
B. 布尔操作符：&&、||、&、|、^
C. 比较操作符：==、!=、<、<=、>、>=
D. 循环：if、else if、else、for
E. 实战案例
- 连续多次输入半径并计算圆面积，输入特定符号后主动退出程序
- 生成一个数组，计算数组元素的总和以及平均值，并找出超过平均值的数字
- 写一个九九乘法口诀表的程序

4) 数组

A. 什么是数组?
B. 数组的常用操作
C. 一维数组
D. 多维数组
E. 实战案例
- 对数组进行排序
- 用数组写一个日历表

5) 切片

A. 什么是切片
B. 切片常用操作
C. 切片与数组的差异
D. 数组和切片的常用操作
E. 实战案例
- string 与切片互相转换，并替换字符

6) Map

A. 什么是 Map

B. 设置、访问 Map 的内容

C. 遍历 Map

D. 实战案例

- 根据学生多次考试的分数，求平均分数

3. 编程的书写规范，注释格式

1) 格式化

2) 注释

模块二：函数与包

教学目标：

1. 掌握函数的定义与用法
2. 掌握公有函数、私有函数、init 函数
3. 掌握函数参数和函数返回值的用法
4. 掌握调用其他包的函数写法

学习和工作中的痛点：

1. 有两个项目不是同一个小组开发的，怎么使用其他组的项目里边的函数?
2. 有很多很棒的开源项目，如何利用他们已经开发出来的成果?

详细内容：

1. 函数定义

1) 函数名

A. 公有函数和私有函数的区别

2) 函数体

A. 花括号引用部分

B. 作用域

3) 返回值

A. 使用返回值

B. 单个和多个返回值

2. **包：方法的公与私、暴露方法与使用**

1） 跨包只能访问公有函数

3. **包与 GitHub：默认从 GitHub 上抓取**

1） 包路径、包名

2） Go 深度依赖 GitHub

3） 使用 module path 来使用 package

4. **Go module 和 Go vendor**

1） 用 module 定义一个项目

2） 使用一个 GitHub 上的项目

3） 使用 vendor 管理依赖

5. **实战案例：**

1） 实现一个 Go 项目并上传到 GitHub

2） 本地开发一个项目，要求调用到 GitHub 项目上的公有函数

模块三：异常处理与 Debug

教学目标：

1. 理解正常情况与异常情况
2. 学会 Debug 的基本思路和方法
3. 能够写出更稳定的代码
4. 掌握单元测试的基础用法

学习和工作中的痛点：

1. 看到程序异常后不知道如何 Debug
2. 不明白如何在异常情况下使用内置函数保护系统运行资源

详细内容：

1. **常见的 error 及其对应的处理方式**

1） 多返回值中包含 error 以及 error 的处理方法

2. panic、recover、defer 的应用场景及使用方法

1) 对不可预见的异常错误进行处理

3. 如何将 recover 与 defer 搭配使用

4. 实战案例：

1) 使用 error 来保证正确的体重输入值

2) Debug 计算体脂函数

3) 单元测试计算体脂函数

模块四：对象与指针

教学目标：

1. 理解什么是对象、指针
2. 掌握对象、指针的定义规则及使用方法
3. 掌握指针的使用方法及特殊处理方法
4. 掌握对象指针和指针函数的使用方法

学习和工作中的痛点：

1. 不知道什么是对象
2. 不了解使用对象有哪些好处
3. 不知道如何根据实际情况定义合适的对象
4. 不懂如何在合适的场景下使用指针来提高编程效率

详细内容：

1. 对象的定义和基本语法结构（属性、成员函数、对象实现接口等）

2. 定义特殊对象，添加扩展方法，定义特殊对象的注意点

3. 指针的定义方法、基本原理、使用场景

4. 对象指针的定义、基本原理、注意事项

5. **指针函数的定义、基本原理、使用方法**

6. **实战案例：**

1） 定义小狗对象，添加小狗对象的函数：小狗喜欢吃各种好吃的，还喜欢蹦蹦跳跳，喜欢转圈圈，喜欢汪汪叫，最喜欢和主人一起去玩儿飞盘游戏。

2） 定义鸭子对象，添加鸭子对象的函数：小鸭子喜欢吃各种好吃的，摇摇摆摆走，喜欢转圈圈，喜欢嘎嘎叫，最喜欢在水里嬉戏、游泳。

模块五：接口

教学目标：

1. 掌握接口的定义
2. 理解接口的用途
3. 掌握接口的实现方法

学习和工作中的痛点：

1. 每次开发新功能的时候，总是得重新更改引用的地方，还要做格式强制转换，非常麻烦
2. 和其他人员共同开发一个功能的时候，总是要等着另一方开发完成后才能继续开发，效率低下

详细内容：

1. **接口的定义与实现**

2. **编写接口的注意点以及对应的处理方式**

3. **实战案例：**

1） 为动物定义一个接口，并让小狗、小鸭子对象实现这个接口：小狗、小鸭子都会走、叫、吃东西。它们都是小动物，都有自己的吃法。小狗汪汪叫，小鸭嘎嘎叫；小狗吃骨头，小鸭吃青菜；小狗扑扑跑，小鸭摇摇摇。

模块六：Go 的并发编程

教学目标：

1. 掌握 Goroutine 的概念
2. 理解什么是并发编程
3. 理解 Go 语言在并发编程里的优势
4. 掌握 Go 语言并发编程的实现方法

学习和工作中的痛点：

1. 不理解线程和进程的区别
2. 不了解 Go 的多线程间通信的原理，无法提升编程效率
3. 不知道在什么情况下可以使用 Go Channel 来提高性能
4. 不知道如何使用选择器

详细内容：

1. Goroutine：Go 的多线程、如何启动多线程、输出结果

1）启动多个 Goroutine 看输出

2）启动多个 Goroutine 对同一个变量累加，看最终的计算结果

2. 锁： 锁的定义、如何加锁、读写锁与普通锁的区别

1）锁是什么

2）启动多个 Goroutine 对同一个变量累加，有锁的保护下，看最终的计算结果

3）读写锁

4）用读写锁锁定一个变量，观察它与普通锁的差别。

3. Channel：通信关键、无缓存与有缓存 Channel 的定义及差别、关闭 Channel 的方法

1）Channel 的定义

2）使用 Channel

3）无缓存、有缓存的 Channel 在使用上的差别

4）关闭 Channel

5）单向 Channel（出、入）及使用方法

4. **Context：上下文定义、可以 cancel 的上下文、带时间的上下文、带 KV 的上下文、多线程上下文控制、上下文树、关闭上下文**

1) Context 是什么
2) Go 语言中对 Context 的使用
3) Context 关闭后整个树都退出

5. **Select：选择器、Channel 互动、无 block 的选择器、选择器的随机性**

1) 优雅地使用 Channel
2) 优雅地关闭 Channel

6. **实战案例：**

1) 使用多个 Goroutine 求 N 个素数
2) 用锁实现生产者消费者模型
3) 用 Channel 实现生产者消费者模型

模块七：数据结构与算法

教学目标：

1. 理解常见数据结构的使用场景
2. 理解常见算法的基本原理

学习和工作中的痛点：

1. 不会用算法写出高质量的程序
2. 不知道怎样计算各种算法的时间和空间复杂度
3. 不知道有哪些常用的数据结构，在什么情况下使用
4. 常见的算法面试题要如何解答以及如何编写程序

详细内容：

1. **数组、队列、栈、单链表、双链表**

2. **排序算法的原理及实现（冒泡排序、归并排序等）**

3. 常见的算法及数据结构面试题讲解

1) 用队列实现栈
2) 用栈实现队列
3) 在一个有序二维数组中找一个数字

4. 实战案例：

1) 实现单链表、双向链表、冒泡排序、归并排序
2) 实现一个猜数字程序：生成一个随机数，然后用二分法最快地找到这个数字，并输出查找过程中尝试过的数字
3) 拓展学习内容推荐：链表、排序、树、遍历树、图论，相关书籍和课程

模块八： I/O 基础

教学目标：

1. 理解 I/O 和字节流
2. 掌握基本的标准 I/O 和文件 I/O

学习和工作中的痛点：

1. 所有的工作内容都需要保存，它们是怎么保存到文件中的，又是如何从文件中解析出来的?
2. 为什么有时候直接断电的时候会丢失数据？怎样避免丢失数据?

详细内容：

1. 编码、解码、文件处理、保存输出

1) 回顾 fmt 输出与标准输入
2) 编码
3) 把身高体重的内容输出到文件
4) 解码

2. I/O 的定义与分类

1) 标准 I/O
2) 文件 I/O
3) 网络 I/O

3. I/O 的编码与解码

1） JSON/YAML Marshal/Unmarshal

2） 其他自定义格式

4. 实战案例：

1） 录入 10 个人的身高体重数据，并保存到文件中

2） 重新运行程序，可以继续录入更多人的身高体重数据，并保存

3） 生成 10000 人的数据，并保存文件，记录保存时长

模块九：依赖管理

教学目标：

1. 理解 Go module
2. 掌握 Go module 的基本用法
3. 掌握 vendor 的基本用法

学习和工作中的痛点：

1. Go 是纯源码管理的，在开发过程中，不可能所有的内容都自己来写，从开源项目中复制粘贴很难维护，有什么方法很方便地管理依赖?
2. Go 的依赖管理中，有时候用的是同一个 module，为何最后编译出来的程序的行为却是不一致的?

详细内容：

1. 认识 Go module

2. 了解 go.mod 的定义与写法

3. 使用 go.mod 管理项目依赖

4. 了解 vendor

5. 使用 vendor 保证项目行为的一致性

模块十：数据库

教学目标：

1. 理解数据库的基本概念
2. 掌握 SQL 的基本语法及 STMT 的使用方法
3. 理解数据库事务
4. 理解什么是 ORM

学习和工作中的痛点：

1. 不知道数据库的组成部分，以及常见的数据库类型
2. 面对大量数据时，不知道应该如何拆分再存入数据库
3. 不知道如何操作数据，SQL 语句不熟练

详细内容：

1. **什么是数据库、表**
2. **表的组成及字段属性的基本概念**
3. **数据库操作基本语法：使用 SQL 语句进行增删改查**
4. **使用 Go 程序操作数据库的数据**
5. **STMT 的用法以及原因**
6. **ORM：模型映射与 ORM 库**
7. **实战案例：**

1) 将名字、身高、体重、BMI 信息保存到数据库
2) 使用 SQL 语句查询、增加、更改、删除身高体重条目
3) 使用 Go 程序查询、增加、更改、删除身高体重条目
4) 使用 ORM 模型查询、增加、更改、删除身高条目

模块十一：网络编程

教学目标：

1. 掌握 HTTP 网络编程基础
2. 理解证书的基本使用方法
3. 掌握常见的加密与解密方法

学习和工作中的痛点：

1. 不理解 HTTP 协议
2. 不知道如何进行加解密操作

详细内容：

1. **用 Go 开发简单的 HTTP 客户端和服务端**

2. **加密与解密：内部算法、加密解密示意、安全散列算法（SHA）**

3. **实战案例：**

1） 启动一个 HTTP 服务端，使浏览器可以访问它

2） 编写 HTTP 客户端，并访问 HTTP 服务端，查看身高体重

模块十二：Go Web 编程

教学目标：

1. 理解并掌握常见的 Go Web 框架
2. 掌握 Web 开发前后端的交互设计

学习和工作中的痛点：

1. 不会使用框架来快速开发项目
2. 不知道前后端的数据如何进行交互
3. 不会合理组织项目代码结构

详细内容：

1. Gin/Beego：示例、分析用法、扩展用法
2. 使用 Gin 框架注册后端服务
3. 前后端的交互设计
4. 分析并设计前后端交互逻辑
5. 前后端交互的数据、API
6. 完成前后端交互的模型设计
7. 实现服务器后端的 API 和业务逻辑
8. 实现客户端通过 API 进行逻辑操作

模块十三：企业级应用开发

教学目标：

1. 掌握分布式的概念
2. 理解分布式的关键问题
3. 了解企业级分布式应用场景与解决方案

学习和工作中的痛点：

1. 搞不懂分布式的关键概念
2. 不会用分布式相关的技术来优化现有项目

详细内容：

1. 为什么要使用分布式
2. etcd 的基本操作与分布式集群

3. **Redis 的操作方法**

4. **如何使用 Redis 提升性能**

5. **RPC：gRPC、ProtoBuf**

6. **实战案例：**

1） 部署 etcd 服务

2） 使用 etcd 作为存储后端，存储身高体重信息，并做相应的操作

3） 将身高体重信息使用 ProtoBuf 编码进行存取

模块十四：容器入门

教学目标：

1. 掌握容器基本操作
2. 学会将开发的应用打包为镜像并运行

学习和工作中的痛点：

1. 应用开发环境不同，在其他人的环境里边正常运行，在我的环境却无法正常运行，常常在环境配置上浪费大量的时间
2. 开发出来的应用部署到生产服务器时很复杂，需要安装许多软件后才能运行，上线过程很复杂，而且很容易出故障

详细内容：

1. **容器：**

1） 什么是容器

2） Docker 有哪些作用

3） 安装 Docker

4） 将应用打包成镜像

5） 使用打包好的镜像

模块十五：企业级项目实践思想

教学目标：

1. 认识模型驱动编程思想
2. 学会分析业务模型并作出抽象
3. 能够独立写出模型驱动器

学习和工作中的痛点：

1. 在企业的业务开发中，有很多代码是为特定流程编写的。随着业务逻辑的累加，代码有很多 if/else 的嵌套，到某个点的时候代码就会变得无法维护。
2. 系统的业务后端做了版本升级后，好多系统无法运行，必须让所有的系统都做适配后才能升级。导致本来一个很小的组件升级，结果演变成大范围的系统变更，效率极其低下，甚至无法推动升级。

详细内容：

1. 介绍模型驱动思想

2. 以部署应用为例，对比面向过程与模型驱动的区别

1） 面向过程会引入很多 if/else，逐渐难以维护

2） 模型驱动将业务信息暴露出来，内部信息隐藏，可以随时做实现的切换

3. 编写一个模型驱动的实例

1） 部署一个应用，需要配置：磁盘、配置文件、启动、启动状态检查、健康状态检查

4. 版本支持

1） 什么是版本

2） 为什么要多版本支持

3） 版本升级过程中的难点

5. 如何做多版本支持

1） 向后兼容的冗余度与版本转换

2） 向后兼容的多版本组件升级

THANKS FOR READING

谢谢阅读