

NekoBytes Week1

- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

Agenda

- **为什么学习 C 语言**
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

C 语言介绍

- “C 不是一种“非常高级”的语言，也不是一种“体量大”的语言，并且不专门针对任何特定的应用领域。但它没有限制且具有通用性，这使得它比所谓更强大的语言更方便、更有效地完成许多任务。” —— K&R
- 实现了第一个非汇编语言编写的操作系统
- 为什么要学习 C？
 - 我们可以编写程序来利用计算机体系结构的底层功能
 - 内存管理！
 - 然而，这也意味着在 C 语言中事情更容易出错

C 的优点

NekoBytes-TheMissing
2024

- 贴近硬件
- 功能强大
- 语法简介
- 学习轻松
- 应用广泛
- 关注底层
- 以及 ...

• 极致的速度与节能!



Table 4. Normalized global results for Energy, Time, and Memory

Total					
	Energy		Time		Mb
(c) C	1.00	(c) C	1.00	(c) Pascal	1.00
(c) Rust	1.03	(c) Rust	1.04	(c) Go	1.05
(c) C++	1.34	(c) C++	1.56	(c) C	1.17
(c) Ada	1.70	(c) Ada	1.85	(c) Fortran	1.24
(v) Java	1.98	(v) Java	1.89	(c) C++	1.34
(c) Pascal	2.14	(c) Chapel	2.14	(c) Ada	1.47
(c) Chapel	2.18	(c) Go	2.83	(c) Rust	1.54
(v) Lisp	2.27	(c) Pascal	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	(c) Ocaml	3.09	(c) Haskell	2.45
(c) Fortran	2.52	(v) C#	3.14	(i) PHP	2.57
(c) Swift	2.79	(v) Lisp	3.40	(c) Swift	2.71
(c) Haskell	3.10	(c) Haskell	3.55	(i) Python	2.80
(v) C#	3.14	(c) Swift	4.20	(c) Ocaml	2.82
(c) Go	3.23	(c) Fortran	4.20	(v) C#	2.85
(i) Dart	3.83	(v) F#	6.30	(i) Hack	3.34
(v) F#	4.13	(i) JavaScript	6.52	(v) Racket	3.52
(i) JavaScript	4.45	(i) Dart	6.67	(i) Ruby	3.97
(v) Racket	7.91	(v) Racket	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	(i) Hack	26.99	(v) F#	4.25
(i) Hack	24.02	(i) PHP	27.64	(i) JavaScript	4.59
(i) PHP	29.30	(v) Erlang	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	(i) Jruby	43.44	(v) Java	6.01
(i) Lua	45.98	(i) TypeScript	46.20	(i) Perl	6.62
(i) Jruby	46.54	(i) Ruby	59.34	(i) Lua	6.72
(i) Ruby	69.91	(i) Perl	65.79	(v) Erlang	7.20
(i) Python	75.88	(i) Python	71.90	(i) Dart	8.64
(i) Perl	79.58	(i) Lua	82.91	(i) Jruby	19.84


NekoBytes

Linux 系统内核

- bpf_local_storage.c
- bpf_lru_list.c
- bpf_lru_list.h
- bpf_lsm.c
- bpf_struct_ops.c
- bpf_struct_ops_types.h
- bpf_task_storage.c
- btf.c
- cgroup.c
- core.c
- cpumap.c
- devmap.c
- disasm.c
- disasm.h
- dispatcher.c
- hashtab.c
- helpers.c
- inode.c
- local_storage.c

The Linux Kernel Archives

[About](#)[Contact us](#)[FAQ](#)[Releases](#)[Signatures](#)[Site news](#)



Languages

C 98.4%

Shell 0.3%

Python 0.1%

Assembly 0.9%

Makefile 0.2%

Perl 0.1%

	Protocol	Location	
	HTTP	https://www.kernel.org/pub/	
	GIT	https://git.kernel.org/	
	RSYNC	rsync://rsync.kernel.org/pub/	

La

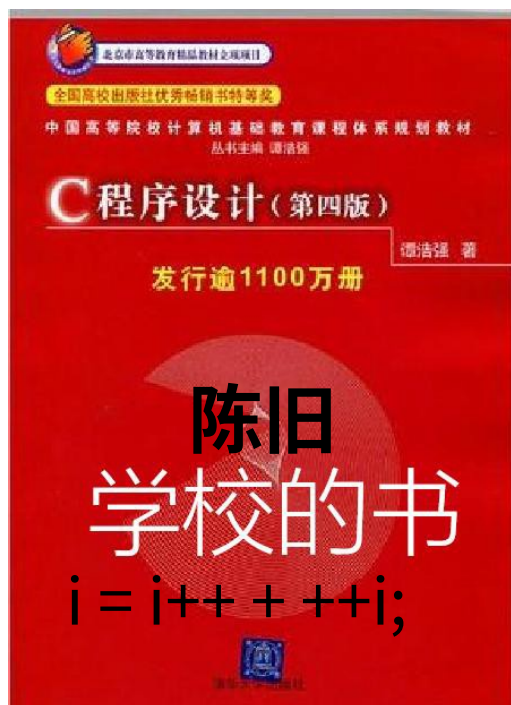
5.14

mainline:	5.15-rc4	2021-10-03	[tarball]	[patch]	[inc. patch]	[view diff]	[browse]	
stable:	5.14.10	2021-10-07	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
stable:	5.13.19 [EOL]	2021-09-18	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	5.10.71	2021-10-06	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	5.4.151	2021-10-06	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.19.209	2021-10-06	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.14.249	2021-10-06	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.9.285	2021-10-06	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.4.287	2021-10-07	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
linux-next:	next-20211007	2021-10-07						[browse]

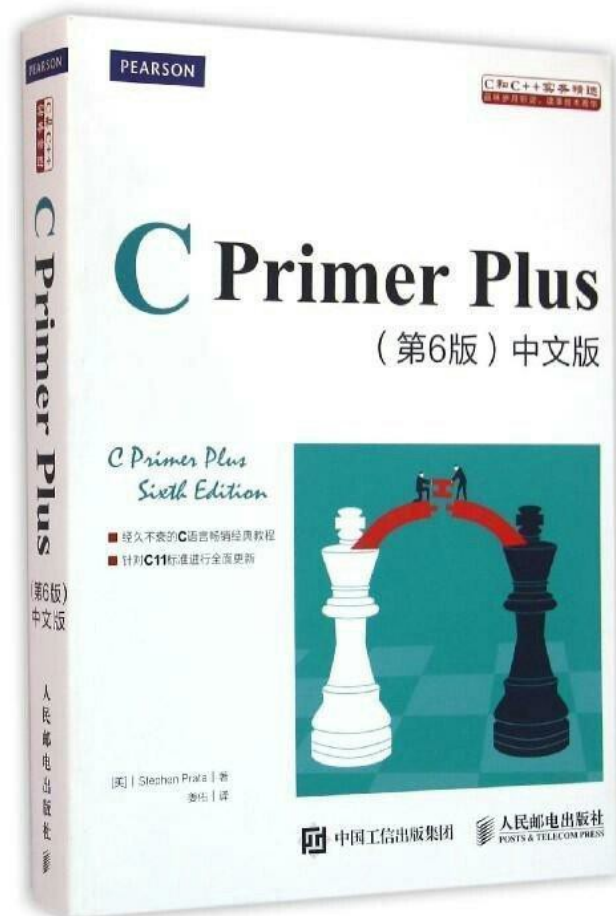
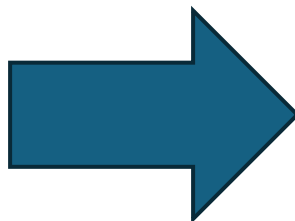
Agenda

- 为什么学习 C 语言
- **怎么学习 C 语言**
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

学习 C 语言看什么书？



浅薄

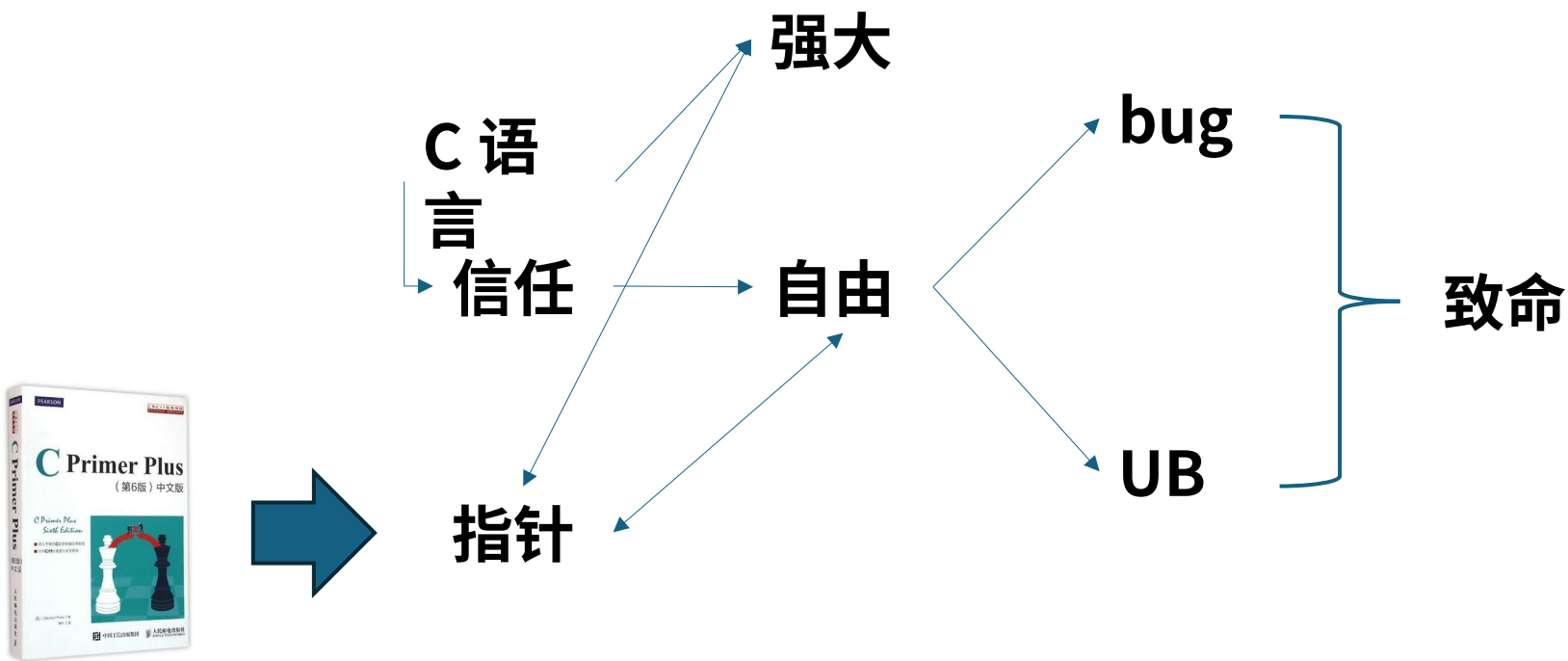


C 语言的推荐资源


- [Introductory C Programming 专项课程]
<https://www.coursera.org/specializations/c-programming?>
- [翁恺 C 语言基础入门]@bilibili:
<https://www.bilibili.com/video/BV1dr4y1n> 一节课入门C语言
- [C 参考手册] <https://zh.cppreference.com/>






快乐 C 之旅




遇到问题怎么解决？




有问题尽管问我...





片 视频 地图 资讯 更多 工具




计科协C语言课程
群号: 223085198










Google 搜索

Google 提供




首页 问答 专栏 讲堂 发现

在 SegmentFault, 学习技能、解决问题

每个月, 我们帮助 1000 万的开发者解决各种各样的技术问题。并助力他们在技术能力、职业生涯、影响力上获得提升。

为你推荐

为你推荐




Search...

Learn, Share, Build

Each month, over 50 million developers come to Stack Overflow to learn, share their knowledge, and build their careers.

Join the world's largest developer community.



RTFM ! STFW!

RTFM: Read The Friendly Manual

为什么要阅读手册？
准确、迅速地给你答案

STFW : Search The Friendly Web

RTFM 和 STFW：如何知道你已完全搞砸了

有一个古老而神圣的传统：如果你收到 RTFM 的回应，回答者认为你应该去读他妈的手册。当然，基本上他是对的，你应该去读一读。

RTFM 有一个年轻的亲戚。如果你收到 STFW 的回应，回答者认为你应该到他妈的网上搜索。那人多半也是对的，去搜索一下吧。（更温和一点的说法是 [Google 是你的朋友!](#)）

在论坛，你也可能被要求去爬爬论坛的旧文。事实上，有人甚至可能热心地为你提供以前解决此问题的讨论串。但不要依赖这种关照，提问前应该先搜索一下旧文。

通常，用这两句之一回答你的人会给你一份包含你需要内容的手册或者一个网址，而且他们打这些字的时候也正在读着。这些答复意味着回答者认为：

你需要的信息非常容易获得；

你自己去搜索这些信息比灌给你，能让你学到更多。

你不应该因此不爽；依照黑客的标准，他已经表示了对你一定程度的关注，而没有对你的要求视而不见。你应该对他祖母般的慈祥表示感谢。

Agenda

- 为什么学习 C 语言
- 怎么学习 C 语言
- **用什么写 C 语言**
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

IDE 与编辑器（1/2）

- IDE（集成开发环境）和编辑器是软件开发中常用的工具。它们都用于编写、编辑和管理代码，但在功能和用途上有一些区别。
- IDE 是一种集成了多个工具和功能的软件，旨在提供全面的开发环境。它通常包括代码编辑器、调试器、编译器、构建工具、版本控制系统等。IDE 提供了一个统一的界面，使开发人员可以在一个应用程序中完成多个开发任务。
 - Visual Studio
 - Clion
- 编辑器则更加轻量级，专注于提供代码编辑功能。它通常具有语法高亮、自动完成、代码折叠等基本功能，但不包含其他开发工具。编辑器通常更加灵活和可定制，适合开发人员根据自己的需求选择插件和扩展。
 - Visual Studio Code

IDE 与编辑器 (2/2)

IDE



编辑器



编译器

Agenda

- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- **数制与码制**
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

二进制

- 使用 0 和 1 字符串表示数字的方法
- 为什么计算机使用二进制？
 - 计算机的基本构建模块是只能表示两个值的晶体管。我们决定将这两个值标记为 0 和 1
- 通过在字符串前面添加 0b 或添加下标 2 来表示

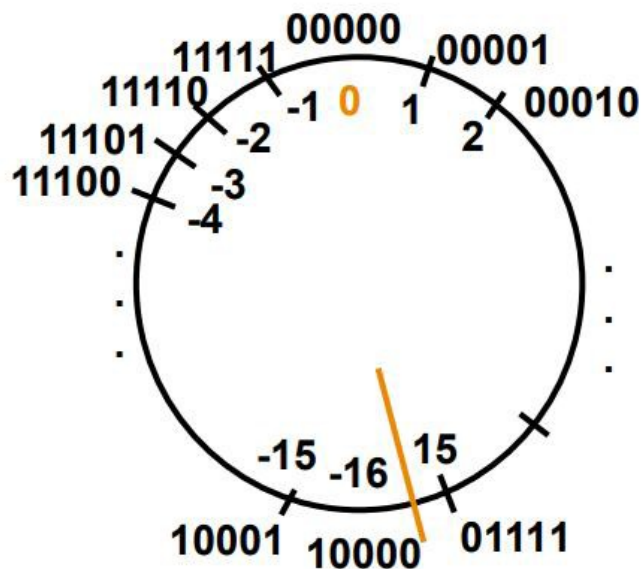
十六进制

- 更易于人类阅读的二进制表示方法
- 基数： 16
- 一位十六进制数字可以表示 16 个数字
- 一位十六进制数字 = 1 个半字节
- 通过前置 “0x” 或附加下标 16 来表示

Decimal	Binary(0b)	Hex(0x)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

补码

- 如何形成负数?
 - 翻转位并加一
- 最高位 代表值的符号
 - 0 表示其正数
 - 1 表示其负数



- 思考
- 有无符号位对表示范围的影响
- 为什么补码可以当作负号使用?
- 考虑忽略溢出

NekoBytes-TheMissing
2024

Single precision		Double precision		Object represented
Exponent	Fraction	Exponent	Fraction	
0	0	0	0	0
0	Nonzero	0	Nonzero	\pm denormalized number
1–254	Anything	1–2046	Anything	\pm floating-point number
255	0	2047	0	\pm infinity
255	Nonzero	2047	Nonzero	NaN (Not a Number)

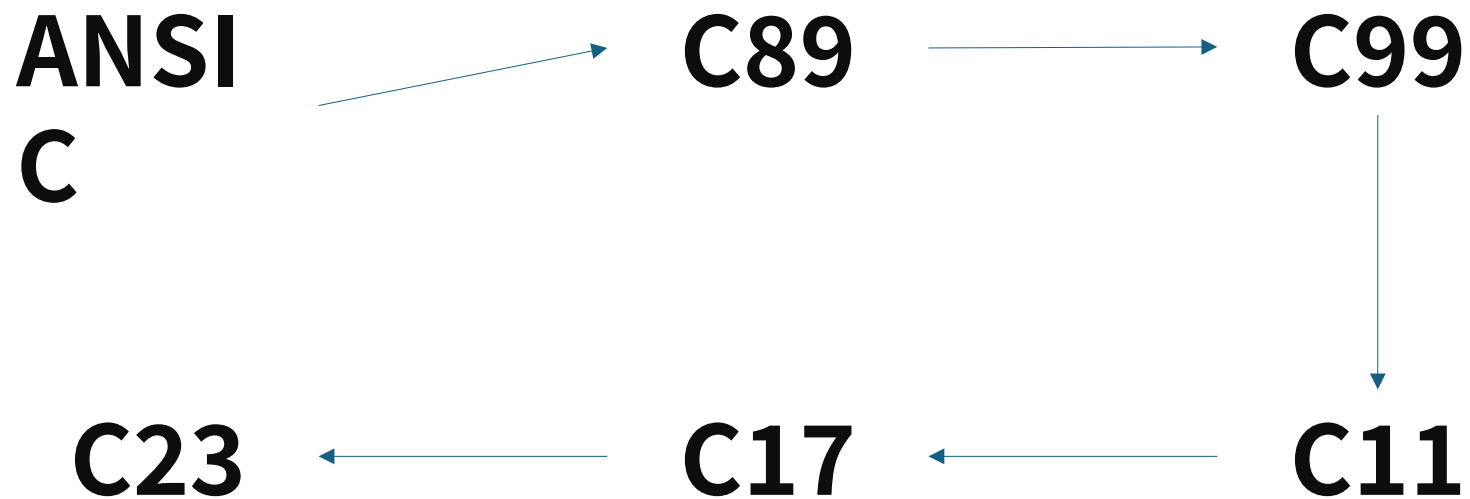
$$(-1)^s \times (1.\text{significand}) \times 2^{(\text{exponent}-127)}$$

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
s	exponent												fraction																		
11 bits													20 bits																		
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fraction																															
32 bits																															

Agenda

- 如何科学的提问
- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- **C 语言基础语法**
- 命令行工具
- VsCode 和 vim

语法标准



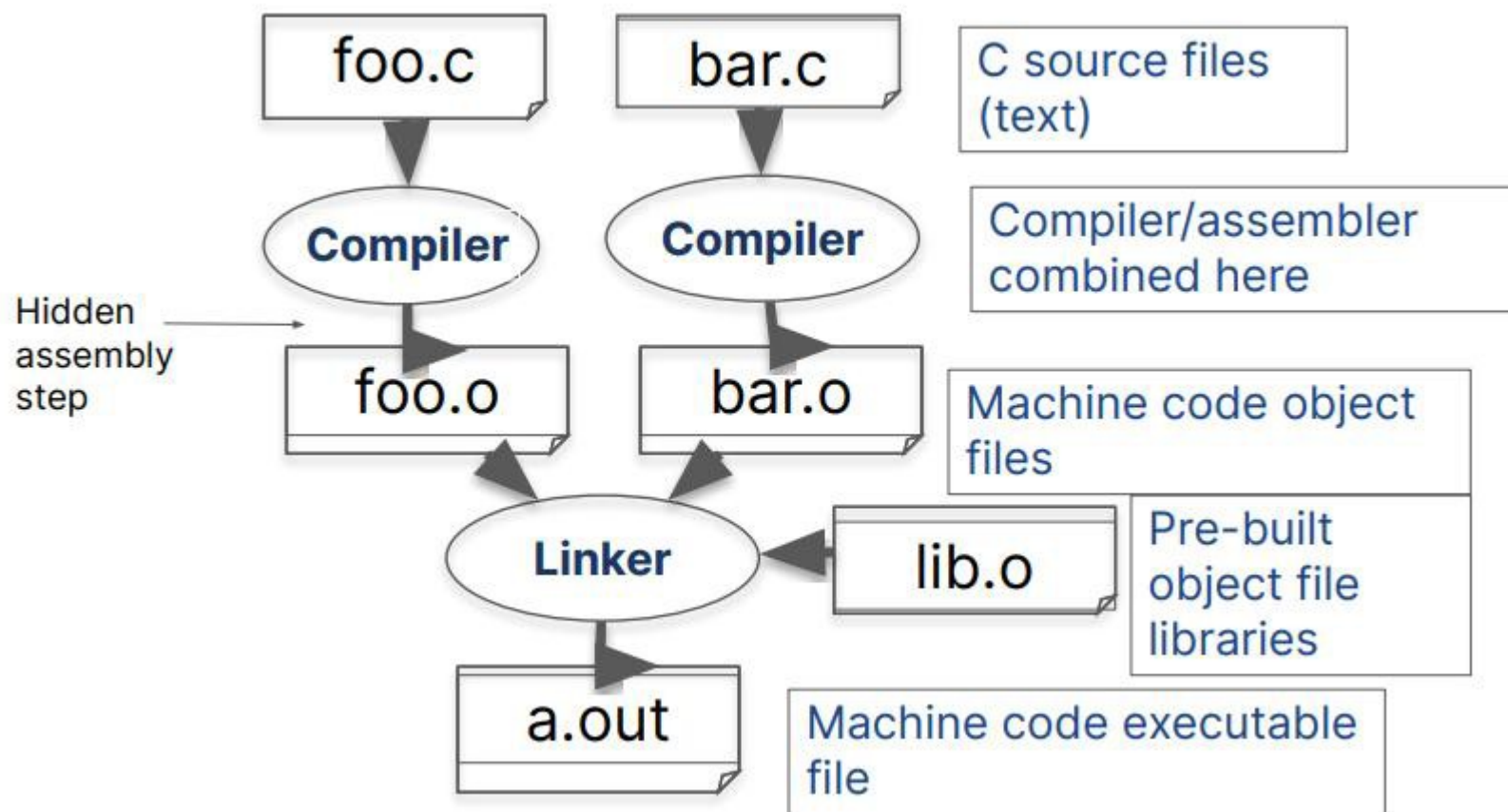
编译 vs 解释

- 我们在某种程度上用英语编写代码，但系统实际上只看到 0 和 1。我们该如何解决这个问题？
 - 如果有人不懂英语，但懂法语，我们会翻译文字！
 - 系统中的处理类似，但我们翻译成非人类可读的语言
- 翻译以两种方式进行
 - 编译（事前翻译）
 - 解释（在线翻译）
 - 有些语言同时使用这两种方式！

编译

- C 编译器将 C 程序直接映射为特定于体系结构的机器代码（1 和 0 的数值串）。
 - Java 转换为独立于体系结构的字节码，然后由即时 (JIT) 编译器进行编译。
 - Python 在运行时而不是编译时转换为 Python 字节码。
 - 运行时编译与 JIT 编译的不同之处在于程序转换为低级汇编语言并最终转换为机器代码的时间。
- 对于 C，处理 .c 文件通常分为 3 个部分
 - .c 文件被编译为 .s 文件 ⇒ 由编译器编译
 - .s 文件被汇编为 .o 文件 ⇒ 由汇编器汇编（此步骤一般是隐藏的，所以大多数时候我们直接将 .c 文件转换为 .o 文件）
 - .o 文件链接在一起创建可执行文件 ⇒ 由链接器链接

C 编译概述



起点： Hello World

- <https://godbolt.org/z/4esj9rvar>
- 程序的组成要件
 - 预编译指令
 - main 函数
 - 注释
 - 限定符
 - 花括号、函数体和块
 - 声明
 - 赋值
 - 函数调用
 - return 语句——返回值

C Pre-Processor (CPP) —— C 预处理器

- C 源文件在编译器看到代码之前首先经过预处理器 CPP
- CPP 用单个空格替换注释
- CPP 命令以 “#” 开头
 - `#include "file.h" /* 将 file.h 插入到文件 */`
 - `#include <stdio.h> /* 在标准位置查找文件，但没有实际区别 */`
 - `#define PI (3.14159) /* 定义常量 */`
 - `#if/#endif /* 有条件地包含文本 */`
- 使用 gcc 的 `-save-temps` 选项查看预处理结果
 - 完整文档位于: <http://gcc.gnu.org/onlinedocs/cpp/>

标准库——官方 DLC

- 标准库（Standard Library）是一组在编程语言中提供常用功能和工具的软件库。在 C 语言中，标准库是由 C 标准委员会定义的，它包含了一系列的头文件和函数，提供了许多常见的操作和功能，如输入输出、字符串处理、内存管理、数学运算等。
- 官方 DLC

此游戏的内容		浏览所有(15)
Fjordur - ARK Expansion Map	免费	
Lost Island - ARK Expansion Map	免费	
ARK: Genesis Season Pass	¥ 58.00	
Valguero - ARK Expansion Map	免费	
Ragnarok - ARK Expansion Map	免费	
ARK: Extinction - Expansion Pack	N/A	
ARK: Aberration - Expansion Pack	N/A	
ARK: Scorched Earth - Expansion Pack	N/A	
Primitive+ ARK Total Conversion	免费	
The Center - ARK Expansion Map	免费	
Crystal Isles - ARK Expansion Map	免费	
ARK: Survival Evolved Original Soundtrack	N/A	
ARK: Expansion Packs Original Soundtrack	N/A	
ARK: Genesis Part 1 Original Soundtrack	N/A	
ARK: Genesis Part 2 Original Soundtrack	N/A	
¥ 58.00		将所有 DLC 添加至购物车

C 标准库头文件

C 标准库的接口由下列头文件的汇集定义。

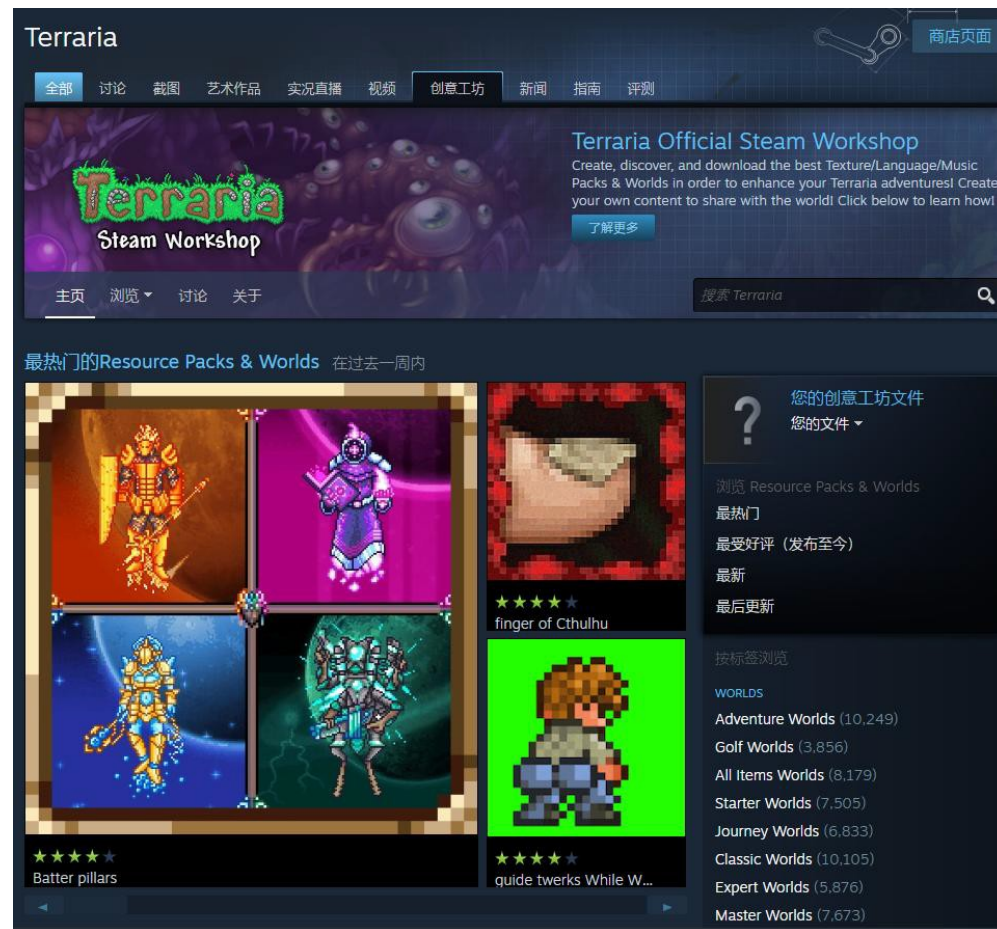
<assert.h>	条件编译宏，将参数与零比较
<complex.h> (C99)	复数运算
<ctype.h>	用来确定包含于字符数据中的类型的函数
<errno.h>	报告错误条件的宏
<fenv.h> (C99)	浮点环境
<float.h>	浮点类型的极限
<inttypes.h> (C99)	整数类型的格式转换
<iso646.h> (C95)	运算符的替代写法
<limits.h>	整数类型的范围
<locale.h>	本地化工具
<math.h>	常用数学函数
<setjmp.h>	非局部跳转
<signal.h>	信号处理
<stdalign.h> (C11)	alignas 与 alignof 便利宏
<stdarg.h>	可变参数
<stdatomic.h> (C11)	原子操作
<stdbool.h> (C99)	布尔类型的宏
<stddef.h>	常用宏定义
<stdint.h> (C99)	定宽整数类型
<stdio.h>	输入/输出
<stdlib.h>	基础工具：内存管理、程序工具、字符串转换、随机数、算法
<stdnoreturn.h> (C11)	noreturn 便利宏
<string.h>	字符串处理
<tgmath.h> (C99)	泛型数学（包装 math.h 和 complex.h 的宏）
<threads.h> (C11)	线程库
<time.h>	时间/日期工具
<uchar.h> (C11)	UTF-16 和 UTF-32 字符工具
<wchar.h> (C95)	扩展多字节和宽字符工具
<wctype.h> (C95)	用来确定包含于宽字符数据中的类型的函数

第三方库——社区模组

NekoBytes-TheMissing
2024

NekoBytes

- 第三方库（Third-party library）是由独立的开发者或组织创建和维护的软件库，它们不是编程语言的标准库的一部分。第三方库通常提供了额外的功能和工具，以扩展编程语言的能力，使开发者能够更快速、更方便地开发应用程序。
- 社区模组



主函数： `main`

- 简单的 `main` 函数形式：
 - `int main(void)`
- 要让 `main` 函数接受参数，请使用以下语句：
 - `int main (int argc, char *argv[])`
- 这是什么意思？
 - `argc` 将包含命令行上的字符串数量（可执行文件计为 1，每个参数加 1）。这里 `argc` 是 2
 - `$ touch a.txt`
 - `argv` 是一个指向数组的指针，该数组包含字符串形式的参数。

C 基本变量类型

- 必须声明变量的类型
 - 强变量类型——类型不能更改。 例如 `. int var = 2;`

类型	描述	例子
• char	• 8 位, ASCII	• 'a' , 'A' , '\n' , 12
• int	• 整数值 (正、负、 0) , >= 16 位, 一般为 32 位	• 0, 78, -217, 0x2E
• unsigned int	• 整数值 (正、 0)	• 0, 6, 35102
• short	• 整数值 (正、负、 0) , >= 16 位, 一般为 16 位	• 0, -8, 32767
• long	• 整数值 (正、负、 0) , >= 32 位, 一般为 32 位	• 0, 78, -217, 301713123194
• long long	• 整数值 (正、负、 0) , >= 32 位, 一般为 64 位	• 31705192721092512
• float	• 单精度浮点数, 32 位, IEEE 754	• 0.0, 3.14, 6.02e3
• double	• 双精度浮点数, 64 位, IEEE 754	• ^
• etc.		

类型的本质

- 类型的本质是对二进制串的不同理解方式
 - 计算机中本只有 0 和 1 ，理解的方式多了，便成了数据类型
- <https://godbolt.org/z/bP1rbWTae>

Agenda

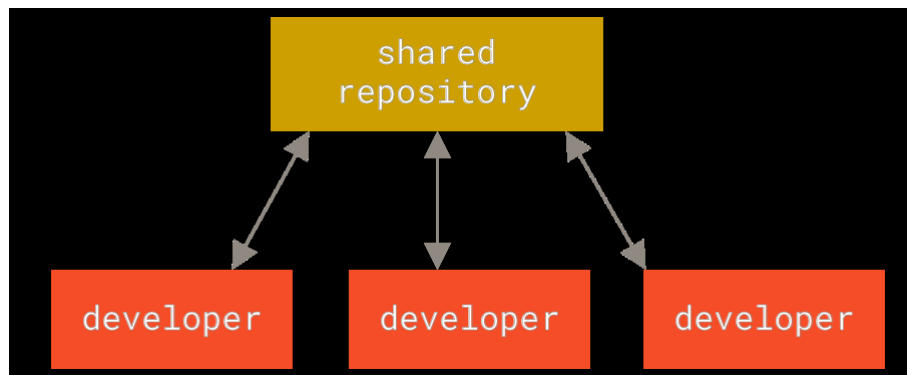
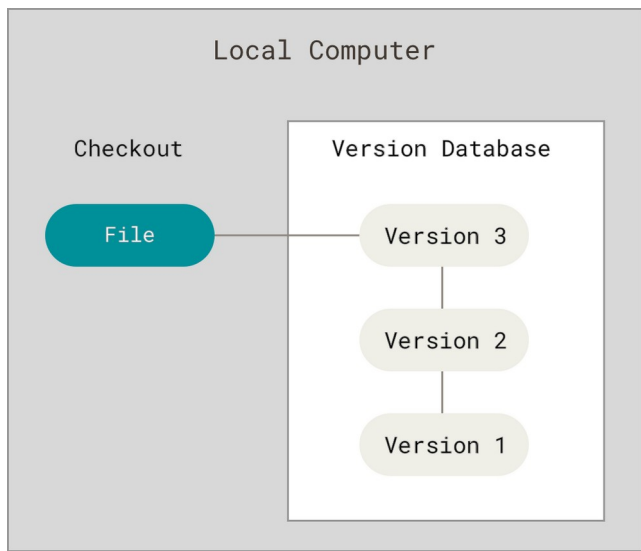
- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 Vim

为什么要用命令行

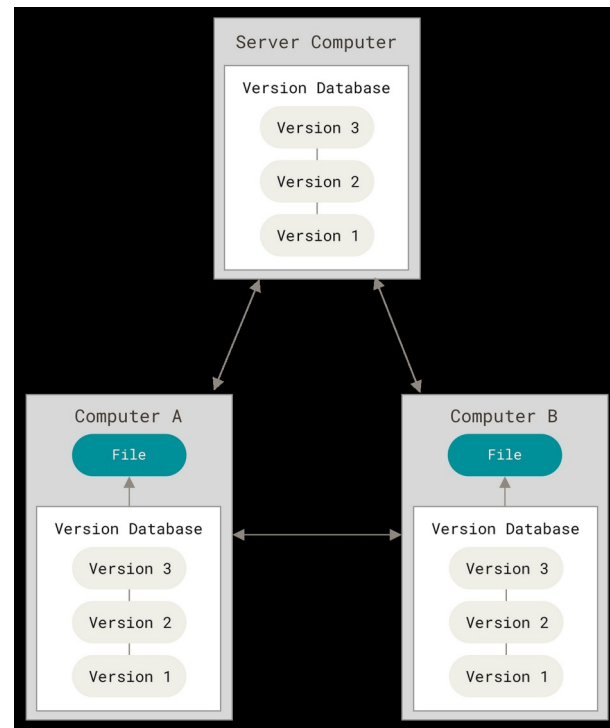
作为计算机的直接操控者，CLI 会给你提供更加直接的操作（想想修改系统某处的时候，你是会选择手忙脚乱的翻 GUI, 还是几行命令解决？）
更何况，CLI 其实更加方便。当你熟练使用，并且配置舒服后，你会喜欢上 CLI 的。

Git

- Git 是分布式版本控制系统的一大代表
- 本地版本控制系统
- 集中化的版本控制系统



- 分布式版本控制系统

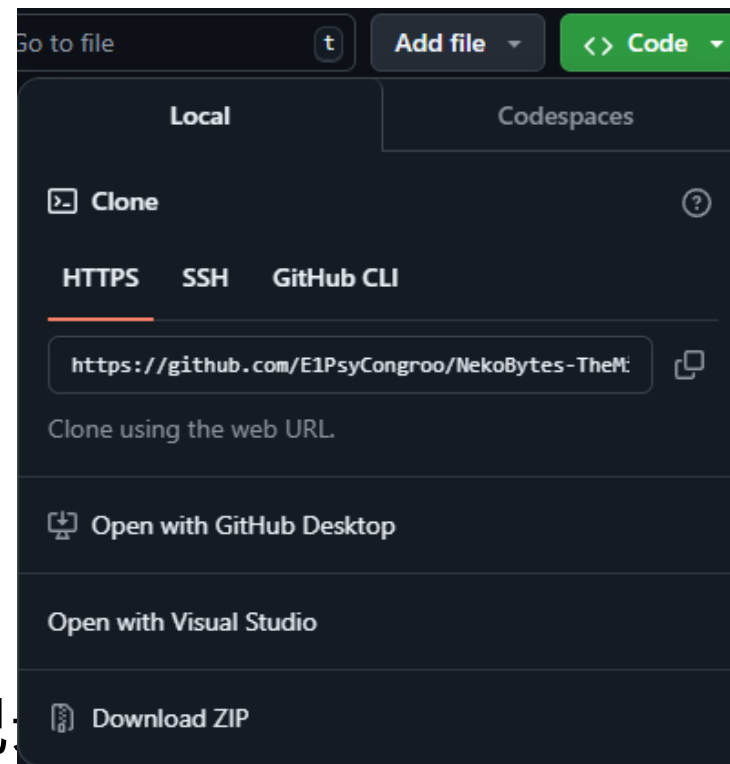


Git 的命令行接口

NekoBytes-TheMissing
2024

NekoBytes

- `sudo apt install git`
- `git help <command>`
- `git init`
- `git status`
- `git add <filename>`
- `git commit`
- `git log`
- `git log --all --graph --decorate`: 可视化历史记录
- **`git clone <url>`**



怎么学 git

- 自己动手实操
 - [Learn Git Branching https://learngitbranching.js.org/](https://learngitbranching.js.org/)（没有项目也能实操）
- 推荐书籍：
 - Pro Git <https://git-scm.com/book/en/v2>

tar

文件后缀

- .tar
- .tar.gz
- .tar.bz2
- .tar.xz

- 怎么来的?
RTFM !

解压命令

- tar -xvf <filename>
- tar -xzvf <filename>
- tar -xjvf <filename>
- tar -xJvf <filename>

GCC

- gcc 全称 GNU project C and C++ compiler
- gcc 是 C 语言的一种编译器
- 易混淆的三样东西
 - Visual Studio——IDE
 - Visual Studio Code—— 文本编辑器
 - GCC—— 编译器

GCC 的命令行接口

- 功能参数
 - -o 指定输出文件名
 - -Wall 开启警告
 - -O(2/3, ...) 优化
- 完整命令
 - `gcc -o xxx xxx.c`
- 运行
 - `./xxx`
- 进阶用法
 - Make 与 Makefile

Agenda

- 如何科学地提问
- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

VSCode 环境配置

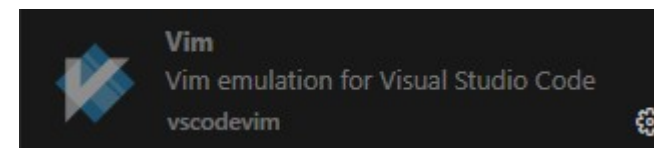
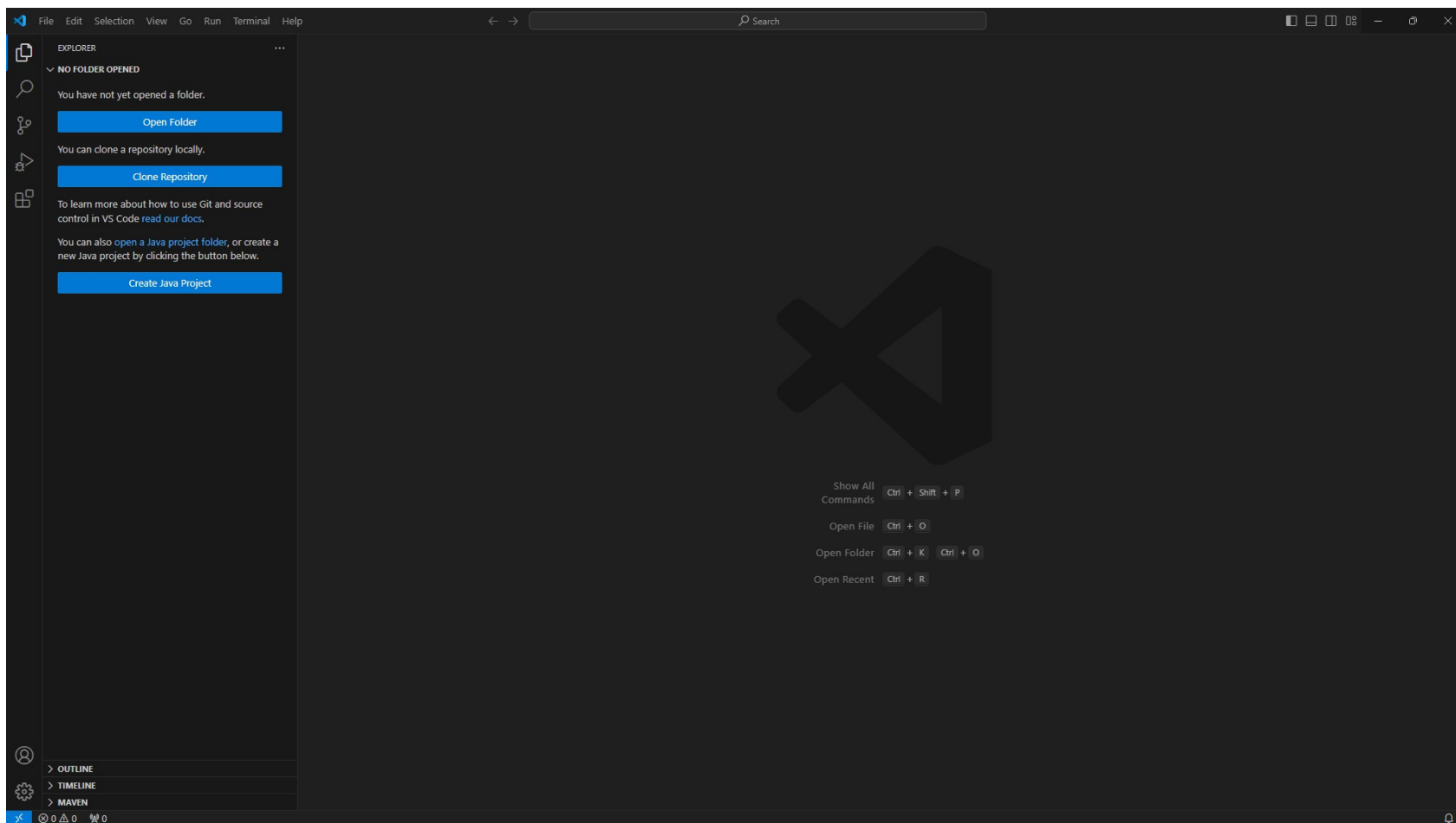
检查你的 week0

适合自己的才是最好的

VsCode 中的 Vim 插件

NekoBytes-TheMissing
2024

NekoBytes



Vim 的哲学

- 在编程的时候，你会把大量时间花在阅读 / 编辑而不是在写代码上。所以， Vim

