# C/C++ 运行环境安装配置指南——初学者极简版

此指南主要目的是用**尽可能极简**的方式，帮助初学者**完成 C/C++ 的基础环境搭建**，三个动作搞定。

——**尤其是小学生小朋友** 😘💘🤗

好多家长朋友很头疼，孩子在学习 C/C++ 的第一步遇到的各种问题不知道怎么解决，例如：

- 有的是培训机构给了一个比较老旧的安装包，勉勉强强弄了能用，却隐藏了各种问题；
- 有的是按照某个教程一步步却走不通，要么下载遇到问题，要么注册环境变量遇到问题。

通过此指南，**用尽可能简单的方式**，指导初学者一步步完成 C/C++ 的编译器、开发环境等下载，**并进行安装、配置和更新。**

> 本教程仅针对 `Windows 10/11` 版本，不推荐 `Windows 7/XP`，本教程也不考虑 `MacOS`、`Linux` 等系统。
>
> 本教程发布和更新于作者陪同孩子学习 `Python`、`C/C++` 的 `Github` 项目： GitHub - coffeescholar/C_CPP-Learning 陪小学生学习 C/C++ 练习题
>
> 欢迎提出建议、意见，也欢迎赠送 Star~ ✨✨✨🤗❤️💕

---

**三个动作如下：**

1. **在线安装** `Scoop` —— 软件包部署和升级工具，以后经常会用到；
2. **自动安装设置** `gcc`，`gdb` —— `GNU` 的 `C/C++` 编译器和调试器； ——不需要设置环境变量等等，自动完成了
3. **测试一下**，确保安装正确
4. **后续可能用到的：**
    1. **更新已经安装的软件包**
    2. **解决访问不了** `Github.com` **的问题**：下载、解压缩、运行 `Fastgithub.UI`；
    3. **下载安装** `VSCode` **并配置** `C/C++ 扩展`，请自行查阅其它文档 ——需要的话，后续补充说明。

# 1. 在线安装 Scoop

## 1.1. 运行终端

有很多种方法，这里只说最简单的：

在 `Windows 10/11` 中，按下 `Win + x` 键，在弹出菜单里选中：`终端`（不要选 `终端管理员`）


## 1.2. 在线安装 Scoop

随后在打开的`终端（命令行）界面`，复制粘贴下面几行命令并按回车（`#` 后面是注释）：

```
# 设置 Powershell 的用户策略，如果提示确认请选择 'Y'
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
# 这里用国内镜像代替了官方的安装地址
iwr -useb https://gitee.com/glsnames/scoop-installer/raw/master/bin/install.ps1 |
iex
# 更新一下 scoop，执行后稍微等一等就好
scoop update
```

没有什么意外的话，稍微等一等就能看到提示完成后续安装。

用 `Scoop` 来安装一些开发常用的工具非常方便，后续会经常用到。

其功能非常强大，远远不止这一点点，建议以后有空多了解一下。

> `Scoop` 是一款适用于 `Windows` 系统的 `命令行软件（包）管理工具`，免费、开源：官网、`Github` 开源项目地址。
>
> 简单来说，就是可以通过命令行工具（`PowerShell`、`CMD` 等）实现软件（包）的安装管理等需求，**通过简单的一行代码实现软件的下载、安装、卸载、更新等操作。**
>
> 其灵感来源于 `macOS` 的 `Homebrew` 开源项目地址，`Mac` 用户就比较熟悉了。
>
> `Scoop` 不仅可用于安装软件包，还可以更新、批量更新，甚至导出安装列表，重装系统后一次性批量安装。
>
> 尤其**对于开发人员而言**，更方便安装指定版本、安装多个版本、运行指定版本，在很多情况下非常有用。

### 1.3. 切换为国内镜像（可选，推荐）

`Scoop` 正常情况下是连接到 `Github.com` 来获取相应的更新数据等等，

因为国内访问 `Github.com` 不稳定，切换为国内镜像会比较方便一些：

```
# 配置仓库为国内镜像
scoop config SCOOP_REPO https://gitee.com/glsnames/scoop-installer
# 配置主桶为国内镜像
cd $env:SCOOP\buckets\Main
git remote set-url origin https://gitee.com/scoop-bucket/main.git
```

后面会推荐访问 `Github.com` 的工具 `FastGithub`，虽然**访问速度不快**但至少**能访问。**

## 2. 自动安装设置 gcc, gdb

随后，用 `Scoop` 帮助我们自动下载安装 `GNU` 的 `C/C++ 编译器和调试器`：`gcc`，`gdb`，`g++` 已经包含在 `gcc` 包里。

在 `终端（命令行）界面` 粘贴或输入下面命令：

```
scoop install gcc, gdb
```

这个过程中，`Scoop` 会自动安装所需的 `7zip`，不需要设置环境变量等等，因为 `Scoop` 已经**自动完成**了。

---

## 3. 测试一下，确保安装正确

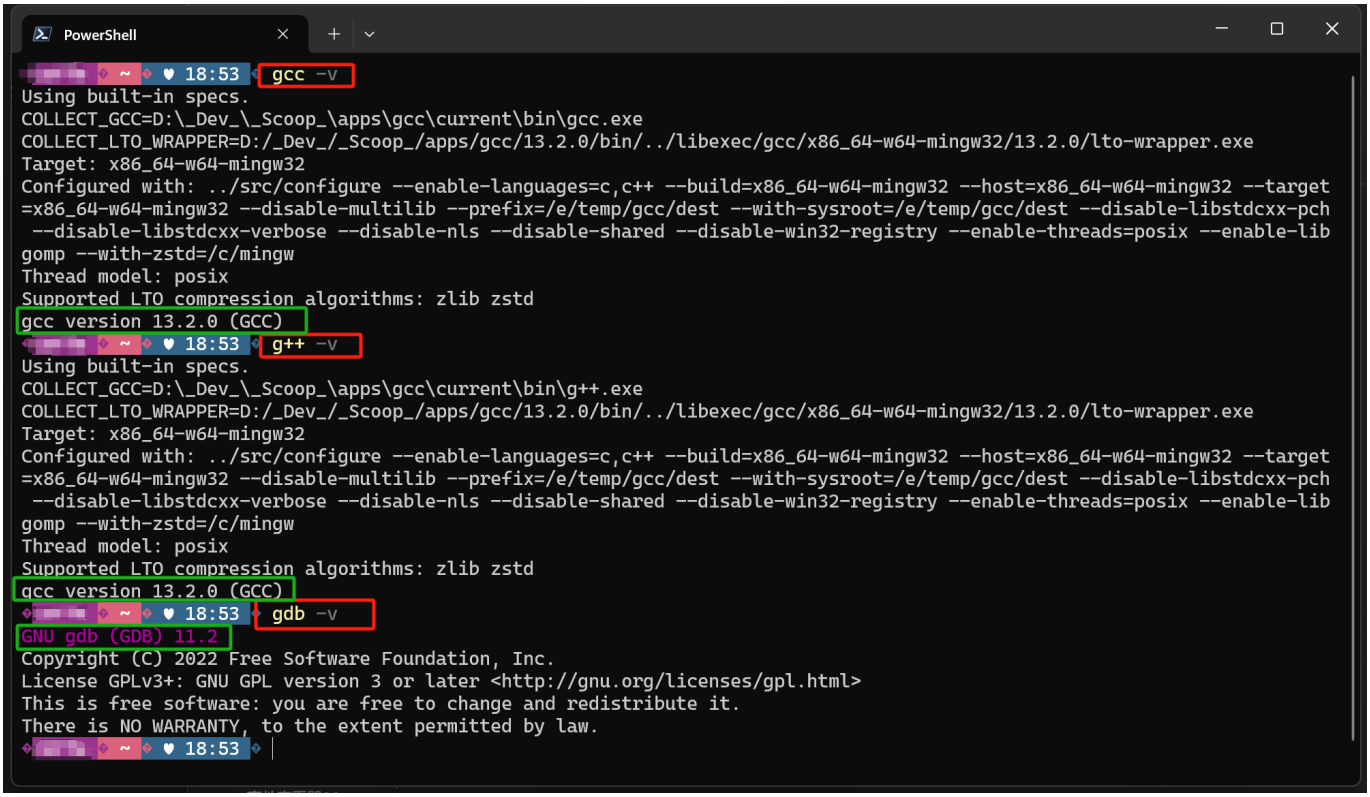
分别运行测试一下，在 `终端（命令行）界面` 中执行下面的命令：

```
gcc -v
g++ -v
gdb -v
```

如果安装失败，则会提示找不到对应的命令，如：

> gcc: The term 'gcc' is not recognized as a name of a cmdlet, function, script file, or executable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.

安装正确的情况下，在每个命令输出中能看到提示和版本号，如图中绿色部分：




# 4. 后续可能用到的

## 4.1. 更新已经安装的软件包

同样，以后可以在 `终端（命令行）界面` 中执行下面的命令，**更新所有安装了的工具包**：

```
# 第一次会自动安装 git
scoop update *
```

也可以**仅更新特定的软件包**：

```
# 可选，查看安装了哪些工具包
scoop list
# 更新特定的软件包，支持多个
scoop update git gcc gdb
```

## 4.2 （可选）下载、解压缩、运行 Fastgithub.UI

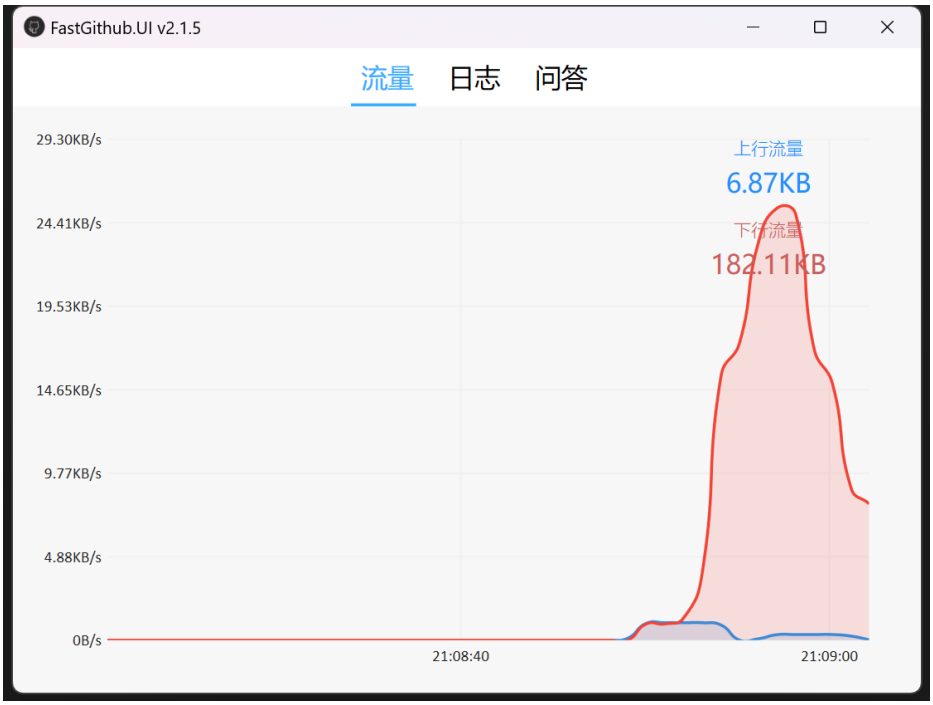
后续学习中，经常需要访问 `Github.com`，然而国内访问 `Github.com` 经常不稳定。

如果这时候总是提示网络问题，可以用 `FastGithub` 工具，开源、免费：

> GitHub - WangGithubUser/FastGitHub: GitHub加速神器，解决GitHub打不开、用户头像无法加载、Releases无法上传下载、git-clone、git-pull、git-push失败等问题

或者，直接下载最新版本的压缩包 `Windows 64 位` 版本：

> https://github.com/WangGithubUser/FastGitHub/releases/download/v2.1.5/fastgithub_win-x64.zip
>
> 解压缩到本地文件夹，双击运行其中的 `FastGithub.UI.exe`




很多文章中也会推荐其它类似 `FastGithub` 的工具，如 `Steam++` 等，相比之下 `FastGithub` 属于最简单、功能单一的一个。

另外，因为 `Steam++` 等的利益（广告），`FastGithub` 原作者关闭了原来在 `Github` 上的项目，

但幸好有喜欢这个工具的其它朋友备份并继续维护着 `FastGithub`，所以**上面分享的并不是原作者的项目地址，而是备份项目地址。**

**感谢原作者，也感谢这些热心的备份者和维护者（还有很多类似备份），谢谢你们** 👍❤️💕

> 注意：`FastGithub` 工具可以配置为系统服务，在 `Windows` 启动时自动运行，
>
> 这里暂不展开，有兴趣可自行查阅其项目说明，如运行原理、源代码等等：
>
> GitHub - WangGithubUser/FastGitHub: GitHub加速神器，解决GitHub打不开、用户头像无法加载、Releases无法上传下载、git-clone、git-pull、git-push失败等问题
>
> 请注意作者的**免责声明**：
>
> - fastgithub **不具备"翻墙"功能,也没有相关的计划**
> - fastgithub **不支持Windows7等已被发行方停止支持的操作系统，并且也不会主动提供支持**
> - fastgithub **不能为您的游戏加速**
> - fastgithub **没有主动在github之外的任何渠道发布**

## 4.3. 下载安装 VSCode，配置插件

编写代码，一般可以用记事本等 `文本编辑器`，更常见是使用专门为编程设计的 `代码编辑器`。

`代码编辑器` 有很多，其中功能相对比较全面、强大的，带有编译、调试、源代码管理甚至项目管理和团队协作等等的，称为 `集成开发环境（IDE，Integrated Development Environment）`。

`集成开发环境 IDE` 也有很多，其中比较出名和被广泛使用的主要有**开源、免费**的 `微软公司 VSCode` 和 `JetBrains` **商业授权**的 `CLion` 等等。

> 微软公司有**付费购买商业授权**的 IDE，被软件开发行业尊称/戏称为 `宇宙第一 IDE` 的 `Visual Studio`，非常强大，伴随了本教程作者的整个软件生涯。 这里的 `VSCode` 全称为 `Visual Studio Code`，正是微软公司后来推出的 `免费开源 IDE` ——但**两者是不同的软件**。 开源社区基于免费开源的 `VSCode` 还延申出了不同的开源版本，以后可以再深入了解。
>
> `JetBrains` 公司有很多非常棒的软件开发工具，比如各种编程语言的 `IDE`，如 `Java` 开发人员最熟悉的 `IntelliJ IDEA`，等等。 该公司的大部分软件需要**付费购买商业授权**才能使用，例如学习 `Python` 也可以用 `PyCharm`，非常不错。 **在校大学生**可以通过向 `JetBrains` 公司申请**教育授权**来免费使用，其它人则可以通过创建和维护开源项目、公益项目来**申请免费试用**。

对于初学者，推荐用 `微软公司` 的 `VSCode`（全称：`Visual Studio Code`）：

> VSCode **官网**： https://code.visualstudio.com/
>
> **直接下载地址**：Download Visual Studio Code - Mac, Linux, Windows

### 4.3.1. 下载安装、配置中文语言包、配置 C/C++ 扩展

1. 下载后安装，默认选择为当前用户安装，建议不要安装到默认的 `C` 盘；
2. 安装、配置中文语言包：
    - 安装完成后，打开 `VSCode`，按下快捷键：`Ctrl-Shift-X` 打开 `Extensions`（扩展）——在左边工具栏有个积木方块样子的图标；
    - 在顶部的搜索栏输入：`Chinese` 并稍等，选择下面的 `Chinese (Simplified) (简体中文) Language Pack for Visual Studio Code`，选中并在右侧点击 `安装`，安装中文语言包扩展；
    - 安装完成后，`VSCode` 会提示重新启动，确认，`restart`，然后就是中文环境了。
3. 安装、配置 `C/C++` 扩展：
    - 重启后，按下快捷键：`Ctrl-Shift-X` 打开 `Extensions`（扩展），在顶部的搜索栏输入：`C/C++`，选择并在右侧点 `安装`；
    - 同上，再安装扩展：`Code Runner`，用于运行 `C/C++`、`Python` 等等常用开发语言的代码。

基本足够了，以后还会有更多好用的扩展，慢慢熟悉、筛选、磨合。

#### 4.3.2. 编写、运行你的第一段 `C/C++` 代码

接下来，可以编写、运行你的第一段 `C/C++` 代码了：

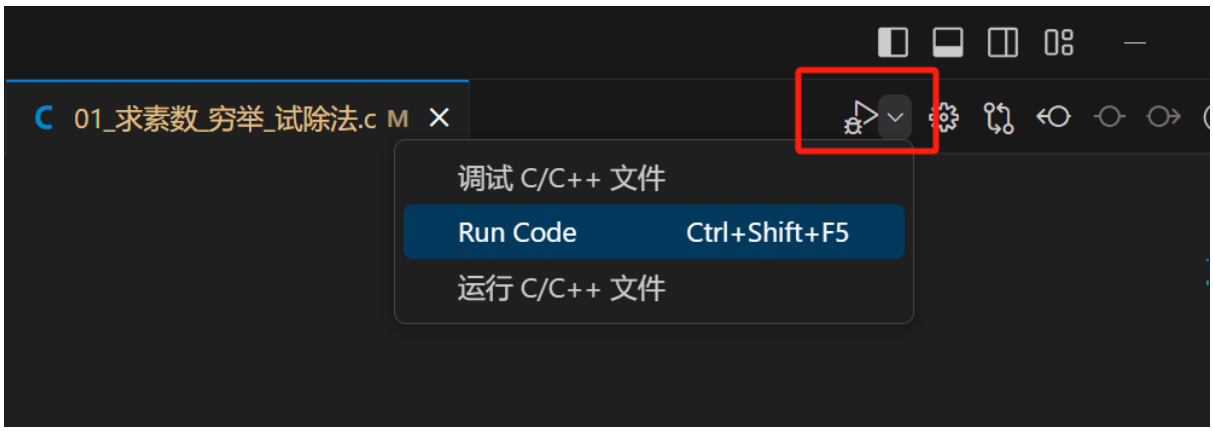
1. **新建文件夹：** 在 `VSCode` 中选择一个文件夹来保存你的代码，例如：`D:\MyCode\Learn_C`；

    > 注意：路径中不要带空格、中文，避免后续编译、调试时无法预料的错误。目前发现，主要是调试时对中文路径支持仍有问题。

2. **新建文件**：在 `VSCode` 左侧的 `资源管理器` 中，点右上角的 `新建文件`，并为创建的文件取个名字比如 `HelloWorld.c`（英文）——注意扩展名用 `.c` 或 `.cpp`，`VSCode` 会自动识别并启用对应的功能；
3. **编写代码**：然后在右侧编辑区域编写代码并保存，例如程序员打开数字世界的第一个口令：

    ```c
    #include <stdio.h>
    void main() {
        printf("Hello World");
    }
    ```

4. **运行代码**：点击编辑区域右上角，类似播放按钮旁的下拉菜单，选择 `Run Code`（记住快捷键更方便）：




5. **查看运行结果**：在底部输出窗口能看到类似下面的运行结果：


---

## 5. 大功告成

目前使用 `Code Runner` 已经能让你开始学习了，后续进一步还需要以下方面的配置：

> c_cpp_properties.json launch.json tasks.json

可参考其它教程，有很多很多。部分教程写的比较复杂，

尽量选择较新的学习，因为版本更新后一些配置可能变化了或者不需要了。

**注意：很多教程的方法比较复杂：**

> - 下载安装 `MinGW64` 或 `Cygwin` 或 `msys2`，下载安装和设置系统环境变量载时遇到问题、安装时也略微复杂，还要**设置系统环境变量** ——其实，本质是通过以上三个应用安装 `GNU` 的 `gcc/gdb`

**本教程简化之处**：

> - **直接安装** `GNU` 的 `gcc/gdb`，不需要通过以上三个或者其它应用；
> - 用 `Scoop` 自动完成下载安装和设置系统环境变量，**省去了复杂的设置操作**，以后还能**随时更新到最新版本。**