如何更高效地开发

——X · Lab 硬件技术团队第一次内训

张卓雨

浙江大学启真交叉学科创新创业实验室

硬件技术团队

2025 年 4 月 20 日

浙江大学启真交叉学科创新创业实验室

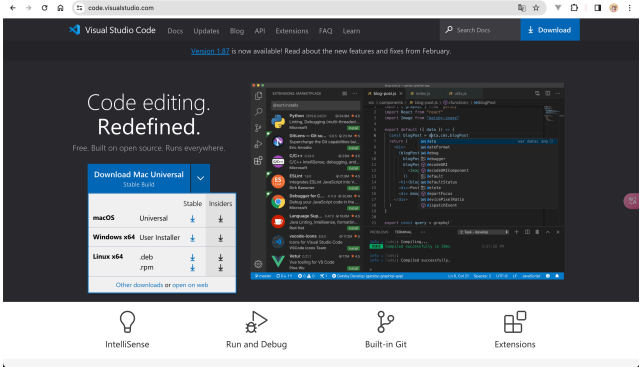
ZJU Inspiration Interdisciplinary Innovation& Entrepreneurship Laboratory

为什么是 VSCode?

- 代码高亮、代码补全、代码格式化等辅助功能做的比大多数 IDE 要好
- 支持多种语言的代码编辑，不必写一个语言开一个 IDE，这对以后用的语言多了很有帮助
- 内置命令行交互（在今天之后要习惯用命令行）
- 它的上限不取决于它本身，而是取决于插件

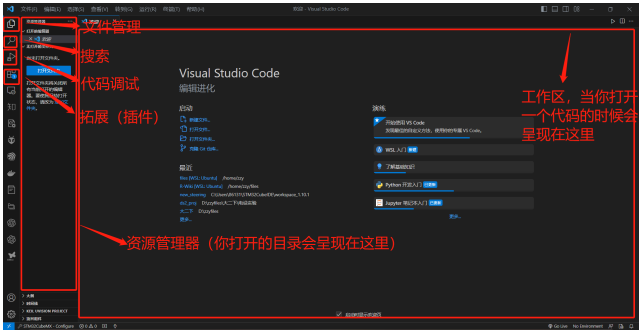
VSCode 的下载和安装

官网点击下载安装即可。



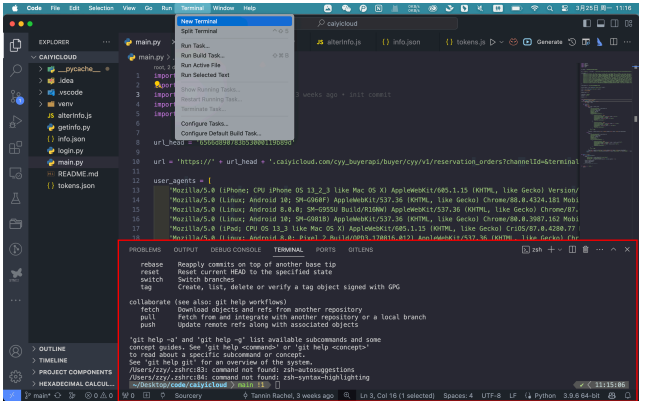
界面介绍

- 第一次安装完进来是英文的，且左侧的侧边栏不会有这么多东西
- 可以在左侧的拓展中搜索 Chinese 来使用中文版
- 暂时只需要了解这几个部分，其他的部分我们边讲边说



命令行

点击上方【终端】，【新建终端】，即可在下方出现一个当前目录下的命令行界面，这个与 Windows 系统下 win+R 后输入 cmd 调用出命令行是一样的。



几个概念的区分

- 编辑器：用来处理文本，例如你电脑自带的文本编辑器、我们正在讲的 VSCode、甚至是 Microsoft Word，它的拓展功能也许能做到智能提示、代码高亮、智能格式化等，但做不到将你的源码文件 (.c) 进行后续的预处理、编译、汇编、连接生成.exe 文件的过程。
- 编译器：用来做将源码文本文件变成计算机理解的二进制文件、可执行文件的过程。
- 集成开发环境（IDE）：用于提供程序开发环境的应用程序，包括了代码编辑器、编译器、调试器和图形用户界面工具。集成了代码编写、分析、编译、调试等一整套工具链。

VSCode 属于编辑器，Dev C++、Visual Studio、PyCharm、Keil、STM32CubeIDE 等属于 IDE。

Dev C++ 帮我们做了哪些事情？

- 预处理：删掉所有的注释、宏拓展、文件包含，处理结束后会产生一个后缀为.i 的临时文件
- 编译：将.i 文件转换成.s 文件（汇编语言）
- 汇编：使用汇编程序将.s 文件转换为机器码 (.o 文件)
- 连接：由.o 文件生成可执行文件（.exe）

不用 Dev C++ 怎么进行这个过程？

以 hello world 程序为例，C 语言代码如下：

```c
#include <stdio.h>
int main() {
    printf("Hello, World!\n");
    return 0;
}
```

接下来我们在命令行中编译运行这个程序：

```
gcc -save-temps demo.c -o demo
```

其中 gcc 是编译器的名称，-save-temps 表示保留所有编译过程中产生的文件（因此在不需要保留这些文件的时候可以不加这句话），demo.c 是你的文件名称，-o demo 表示生成的可执行文件名称叫做 demo。
之后我们再执行这个可执行文件，即在命令行中输入：

```
./demo.exe
```

用 Visual Studio Code 简化上述流程

- 安装 C/C++ 插件
- 安装 Code Runner 插件
- 在 VSCode 中打开一个文件夹，新建一个.c 文件，写入代码
- 点击右上角的运行按钮，或者按下 Ctrl+Alt+N，即可在下方的终端中看到运行结果

为什么要多文件？

- 代码复用：将一些常用的函数写在一个文件中，需要的时候直接调用
- 代码结构清晰：将不同功能的代码写在不同的文件中，方便维护
- 编译速度：只有修改的文件需要重新编译，不需要每次都编译整个项目

打造一个规范的工程模板

一个规范的工程模板应该包含以下几个文件夹/文件：

- inc 文件夹：头文件 (.h 文件)
- src 文件夹：源文件 (.c 文件)
- lib 文件夹：库文件
- bin 文件夹：可执行文件
- build 文件夹：编译过程中生成的文件
- docs 文件夹：文档
- CMakeLists.txt：CMake 的配置文件/Makefile：Make 的配置文件
- README.md：项目说明

“以人为本”的程序设计

“编写程序应该以人为本，计算机第二。”
——Steve McConnell《代码大全》

- 清晰第一：一般情况下，代码的可读性比性能更重要，只有确定性能有问题的时候才需要考虑性能问题。
- 简洁为美：代码越长越难以看懂，也就越容易在修改时引入错误。废弃的代码 (没有被调用的函数和全局变量) 要及时清除，重复代码应该尽可能提炼成函数。
- 选择合适的风格，与代码原有风格保持一致。

头文件规范

- 头文件中只包含函数声明、宏定义、结构体声明、全局变量声明等，不包含函数定义。
- 头文件中不要包含其他头文件，只包含当前文件需要的头文件。
- 头文件中不要定义全局变量，全局变量应该在.c 文件中定义。
- 头文件中不要定义宏，宏应该在.c 文件中定义。
- 要记得加上头文件保护宏，防止头文件被重复包含。

函数规范

函数设计的精髓：编写整洁函数，同时把代码有效组织起来。

- 一个函数只做一件事，同时函数的名字要能够有效地体现出函数的功能。
- 重复代码应该尽可能提炼成函数。
- 避免函数过长，新增函数不超过 50 行（非空非注释行）。
- 避免函数的代码块嵌套过深，新增函数的代码块嵌套不超过 4 层。

给大家看个反例。

变量规范

- 变量名要有意义，不要使用无意义的变量名，如 a、b、c 等。
- 变量名要尽量简洁，同时要能够有效地体现出变量的含义。
- 变量名要使用小写字母，单词之间使用下划线分隔（也可以是驼峰式命名，如：addItem、isChineseStudent 等）。
- 变量名要尽量避免使用缩写，除非是广泛使用的缩写。
- 不用或者少用全局变量，全局变量会增加代码的耦合性，使代码难以维护。
- 防止局部变量与全局变量同名。
- 明确全局变量的初始化顺序，避免跨模块的初始化依赖。

注释规范

- 注释要写在需要解释的代码上方或右侧，而不是下方。
- //和/* 后要有一个空格。
- 大段注释使用/* */，小段注释使用//。
- 一个函数的注释应该包括函数的功能、输入参数、输出参数、返回值等，一个具体的模板如下：

```
1 /**
2 * @brief 函数功能描述
3 * @param param1 参数1的描述
4 * @param param2 参数2的描述
5 * @return 返回值的描述
6 */
7 int func(int param1, int param2);
```

推荐插件：Perttier

Makefile 是什么？

Makefile 是一种构建工具，它可以自动化执行编译、链接等操作，使得代码的编译和运行更加方便。

Makefile 的基本结构如下：

- 目标：依赖
- 　　命令

例如：

```
1 main: main.o func.o
2     gcc -o main main.o func.o
3 main.o: main.c
4     gcc -c main.c
5 func.o: func.c
6     gcc -c func.c
```

CMake 是什么？

CMake 是一个跨平台的构建工具，它可以自动生成 Makefile、Visual Studio 项目、Xcode 项目等，使得代码的编译和运行更加方便。

CMake 的基本结构如下：

```
1 PROJECT (MAIN)
2 SET(SRC_LIST main.c print1.c print.c)
3 add_executable(main ${SRC_LIST})
```

按照规范工程模板写一个 CMake

```
project (main)
set (EXECUTABLE_OUTPUT_PATH ${PROJECT_SOURCE_DIR}/bin)
aux_source_directory (src SRC_LIST)
include_directories (inc)
add_executable (main ${SRC_LIST})
```

CMake 的编译流程

- 进入到 build 目录下：cd build
- 运行 cmake 的命令：cmake ..（.. 是指上一级目录，也就是说，我们要在 build 目录下运行它上一级目录也就是我们的工程文件夹下的 cmake，这样做的目的是把 cmake 生成的乱七八糟的文件都放在 build 目录下）
- 这时已经可以看到 build 目录下生成了 Makefile 文件，之后我们去运行这个 Makefile
- 最后我们即可发现在 bin 目录下生成了我们需要的可执行文件

不要让你的路径出现中文，不然你会像我一样调一晚上也不知道哪出了问题

推荐插件：CMake

先看两个例子

我的 C 大程 Project：

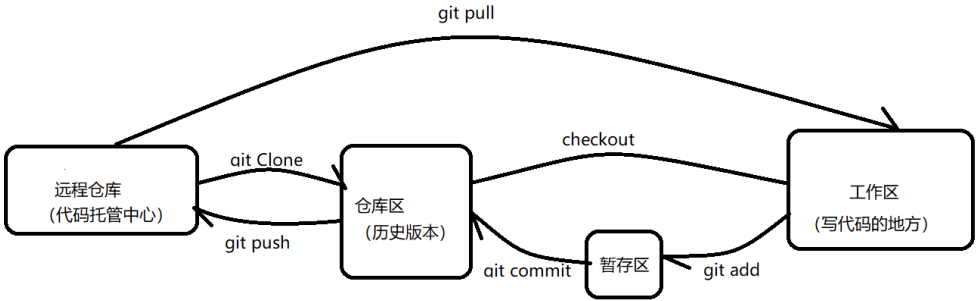
名称	修改日期
.vscode	2023/3/16 21:34
mario	2023/3/16 21:34
v1.1	2023/3/16 21:34
v1.3	2023/3/16 21:34
超级玛丽	2024/6/28 1:07
大程序相关	2023/3/16 21:35
倒数第二版	2023/3/16 21:35
第二版	2023/3/16 21:35
最最终版	2024/6/28 1:09

我朋友的一个 Project：

0.0.0 发送验证码，手动输入验证码登录	24/4/1 23:02
0.1.0 发送验证码，手动输入验证码登录	23/11/18 22:43
0.2.0 接口测试	23/12/6 22:18
0.2.1 增加多线程发送请求	23/12/7 22:36
0.2.2 增加指定时间启动	23/12/9 22:49
0.2.3 增加添加多个观演人的功能	23/12/8 23:16
0.3.0 重构代码删除废弃接口	23/12/10 23:44
0.3.1 优化CPU占用	23/12/10 23:44
0.3.2 优化内存占用	23/12/29 11:17
0.3.3 使用异步请求	23/12/29 9:30
0.3.4 使用进程池	23/12/29 16:11
0.4.0 限制请求频率	24/2/23 0:18
0.5.0 添加预约状态接口	24/3/2 14:05

git 是什么？

- git 是一个免费的、开源的分布式版本控制系统，可以高速处理从小型到大型的各种项目。
- 版本控制：是一种记录文件内容变化，以便将来查阅特定版本修订情况的系统。
- 工作机制：



git 的安装与使用

现场演示一手。

扩个列（备注如24-摸鱼学-张三）：

Thanks!