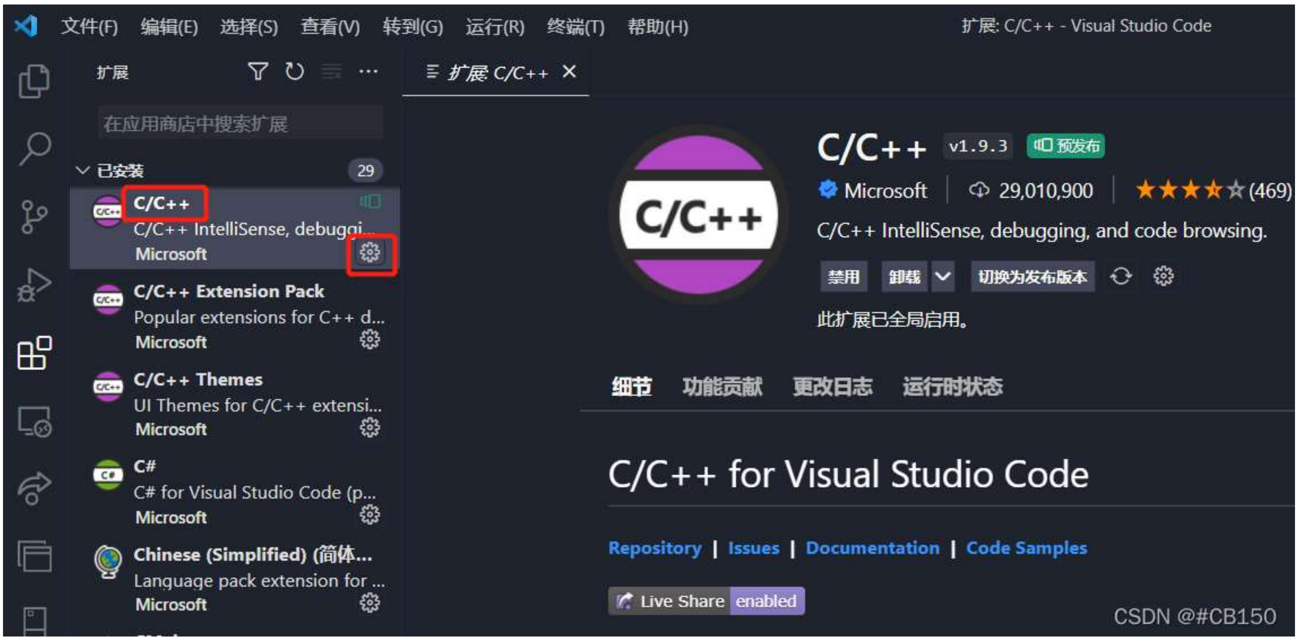
感谢试用沉浸式阅读器。请与我们分享你的反馈。

# VSCode快速配置C语言环境

## 1.在VSCode中安装C/C++插件


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
扩展: C/C++ - Visual Studio Code
扩展
在应用商店中搜索扩展
已安装 29
C/C++
C/C++ IntelliSense, debuggi...
Microsoft
C/C++ Extension Pack
Popular extensions for C++ d...
Microsoft
C/C++ Themes
UI Themes for C/C++ extensi...
Microsoft
C#
C# for Visual Studio Code (p...
Microsoft
Chinese (Simplified) (简体...
Language pack extension for ...
Microsoft
扩展 C/C++
C/C++
C/C++ v1.9.3 预发布
Microsoft 29,010,900 ★★★★☆ (469)
禁用 卸载 切换为发布版本
细节 功能贡献 更改日志 运行时状态
Repository | Issues | Documentation | Code Samples
Live Share enabled
CSDN @#CB150

C/C++ IntelliSense, debugging, and code browsing.

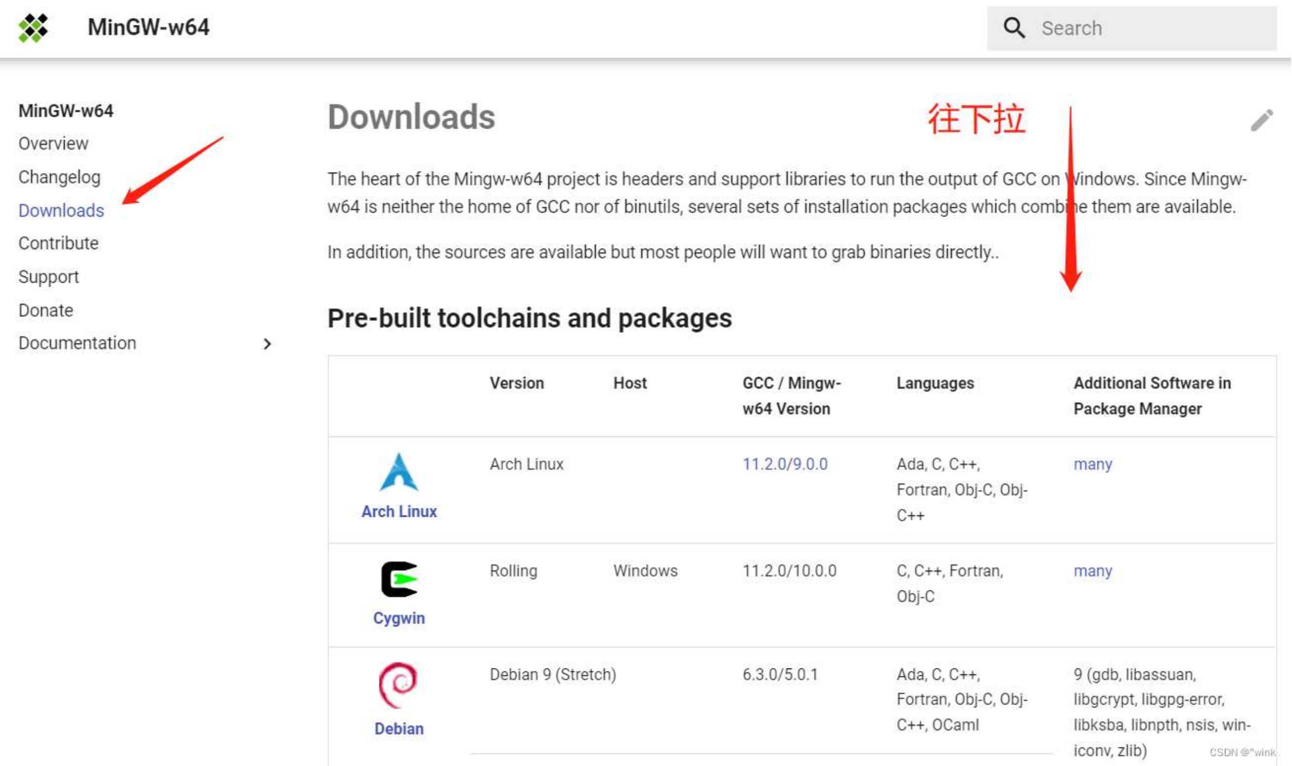
此扩展已全局启用。

C/C++ for Visual Studio Code

## 2.下载mingw64

MinGW-w64官方网站首页

### 1、找到downloads



## Downloads

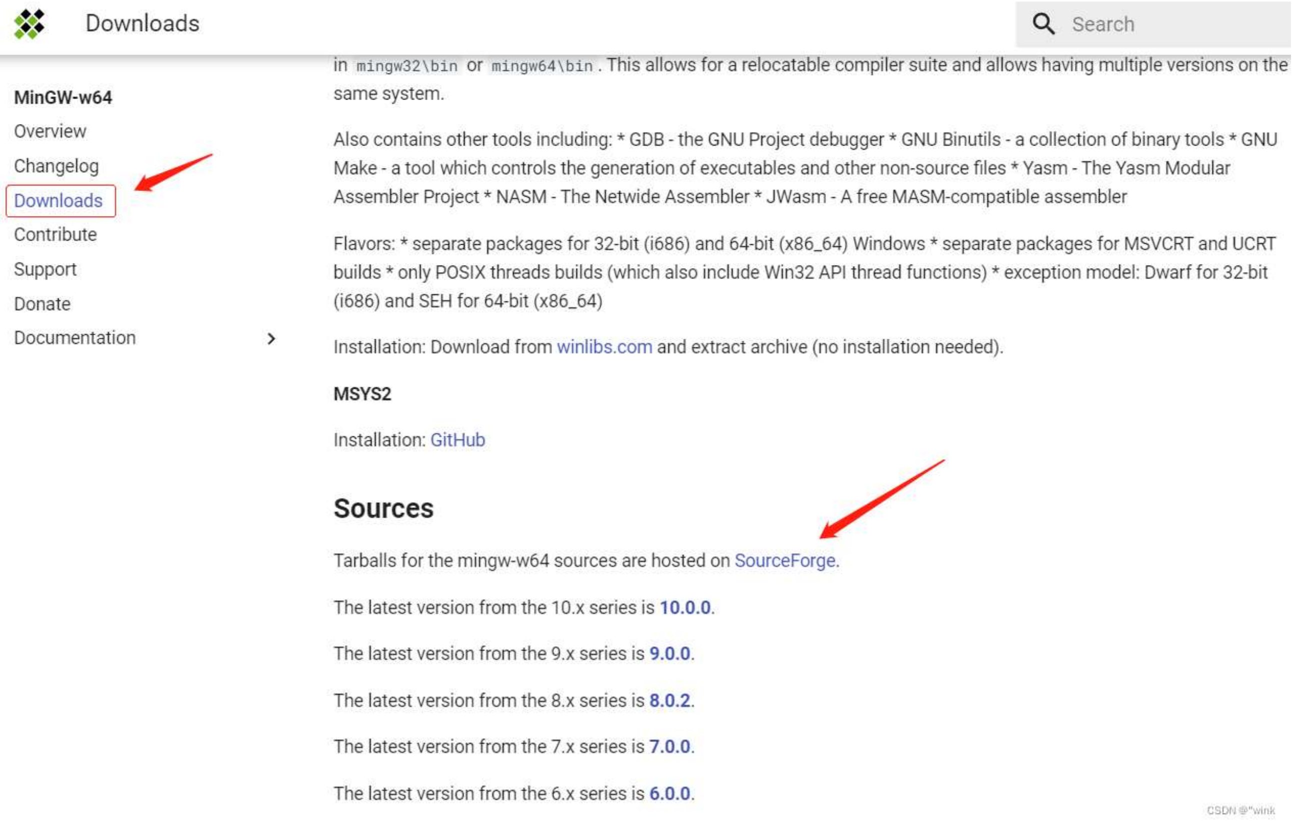
The heart of the Mingw-w64 project is headers and support libraries to run the output of GCC on Windows. Since Mingw-w64 is neither the home of GCC nor of binutils, several sets of installation packages which combine them are available.

In addition, the sources are available but most people will want to grab binaries directly..

### Pre-built toolchains and packages

| | Version | Host | GCC / Mingw-w64 Version | Languages | Additional Software in Package Manager |
|---|---|---|---|---|---|
| Arch Linux | Arch Linux | | 11.2.0/9.0.0 | Ada, C, C++, Fortran, Obj-C, Obj-C++ | many |
| Cygwin | Rolling | Windows | 11.2.0/10.0.0 | C, C++, Fortran, Obj-C | many |
| Debian | Debian 9 (Stretch) | | 6.3.0/5.0.1 | Ada, C, C++, Fortran, Obj-C, Obj-C++, OCaml | 9 (gdb, libassuan, libgcrypt, libgpg-error, libksba, libnpth, nsis, win-iconv, zlib) |

## 2、找到SourceForge



in `mingw32\bin` or `mingw64\bin`. This allows for a relocatable compiler suite and allows having multiple versions on the same system.

Also contains other tools including: * GDB - the GNU Project debugger * GNU Binutils - a collection of binary tools * GNU Make - a tool which controls the generation of executables and other non-source files * Yasm - The Yasm Modular Assembler Project * NASM - The Netwide Assembler * JWasm - A free MASM-compatible assembler

Flavors: * separate packages for 32-bit (i686) and 64-bit (x86_64) Windows * separate packages for MSVCRT and UCRT builds * only POSIX threads builds (which also include Win32 API thread functions) * exception model: Dwarf for 32-bit (i686) and SEH for 64-bit (x86_64)

Installation: Download from winlibs.com and extract archive (no installation needed).

**MSYS2**

Installation: GitHub

### Sources

Tarballs for the mingw-w64 sources are hosted on SourceForge.

The latest version from the 10.x series is **10.0.0**.

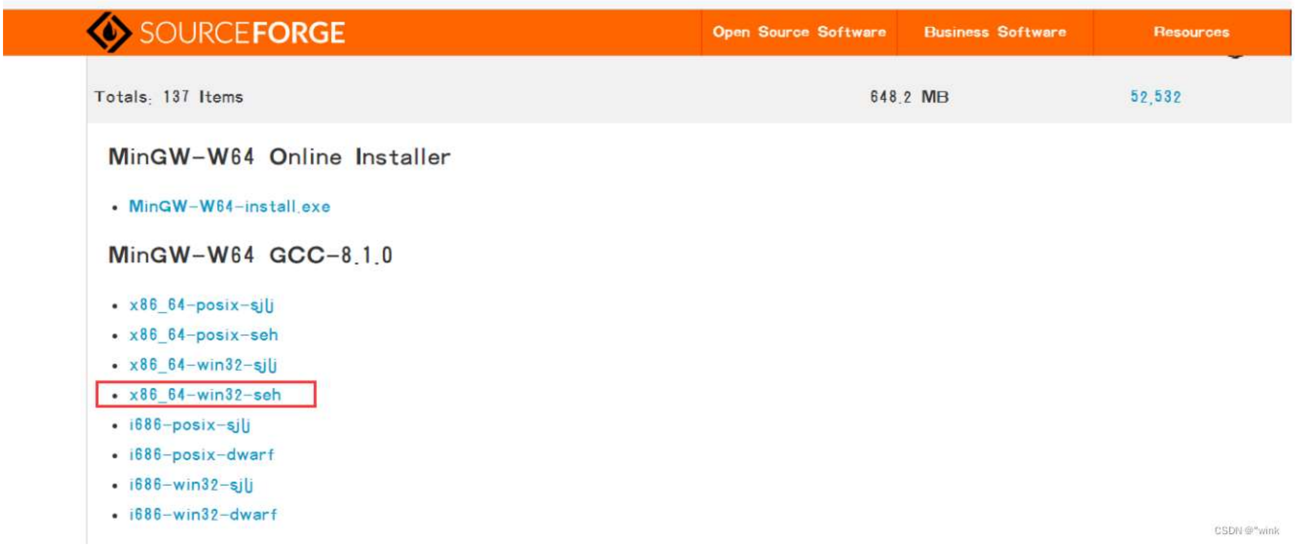
The latest version from the 9.x series is **9.0.0**.

The latest version from the 8.x series is **8.0.2**.

The latest version from the 7.x series is **7.0.0**.

The latest version from the 6.x series is **6.0.0**.

## 3、找到一个合适的版本（我这里是下拉找到免安装版）下载，其他的都试过了，都不行（可能是因为外网的关系连接不稳定）



Totals: 137 Items 648.2 MB 52,532

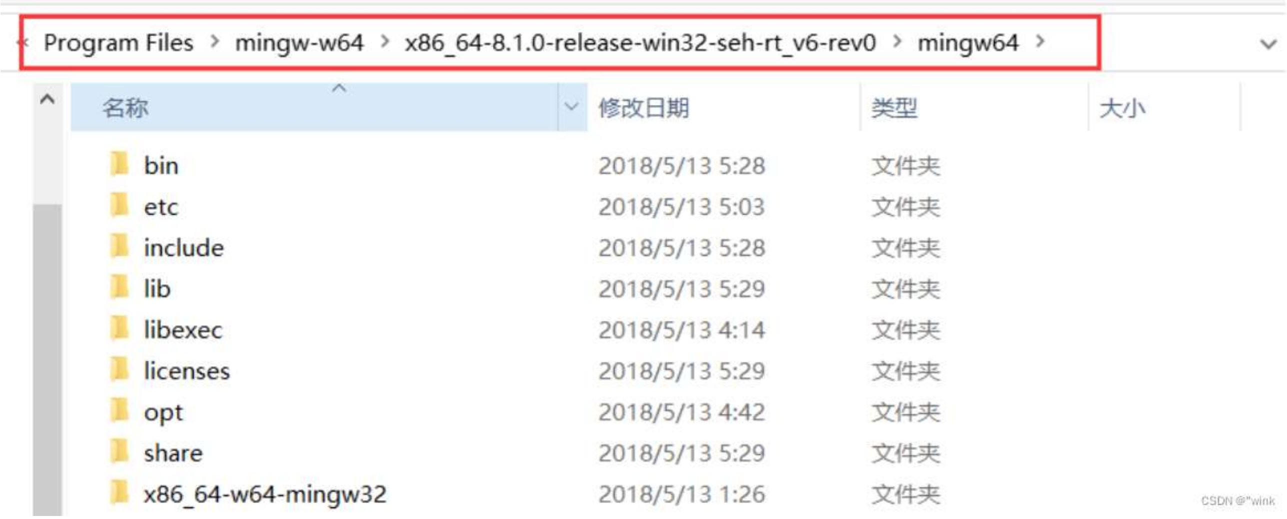
MinGW-W64 Online Installer

- MinGW-W64-install.exe

MinGW-W64 GCC-8.1.0

- x86_64-posix-sjlj
- x86_64-posix-seh
- x86_64-win32-sjlj
- x86_64-win32-seh
- i686-posix-sjlj
- i686-posix-dwarf
- i686-win32-sjlj
- i686-win32-dwarf

## 4、直接解压到想要的位置

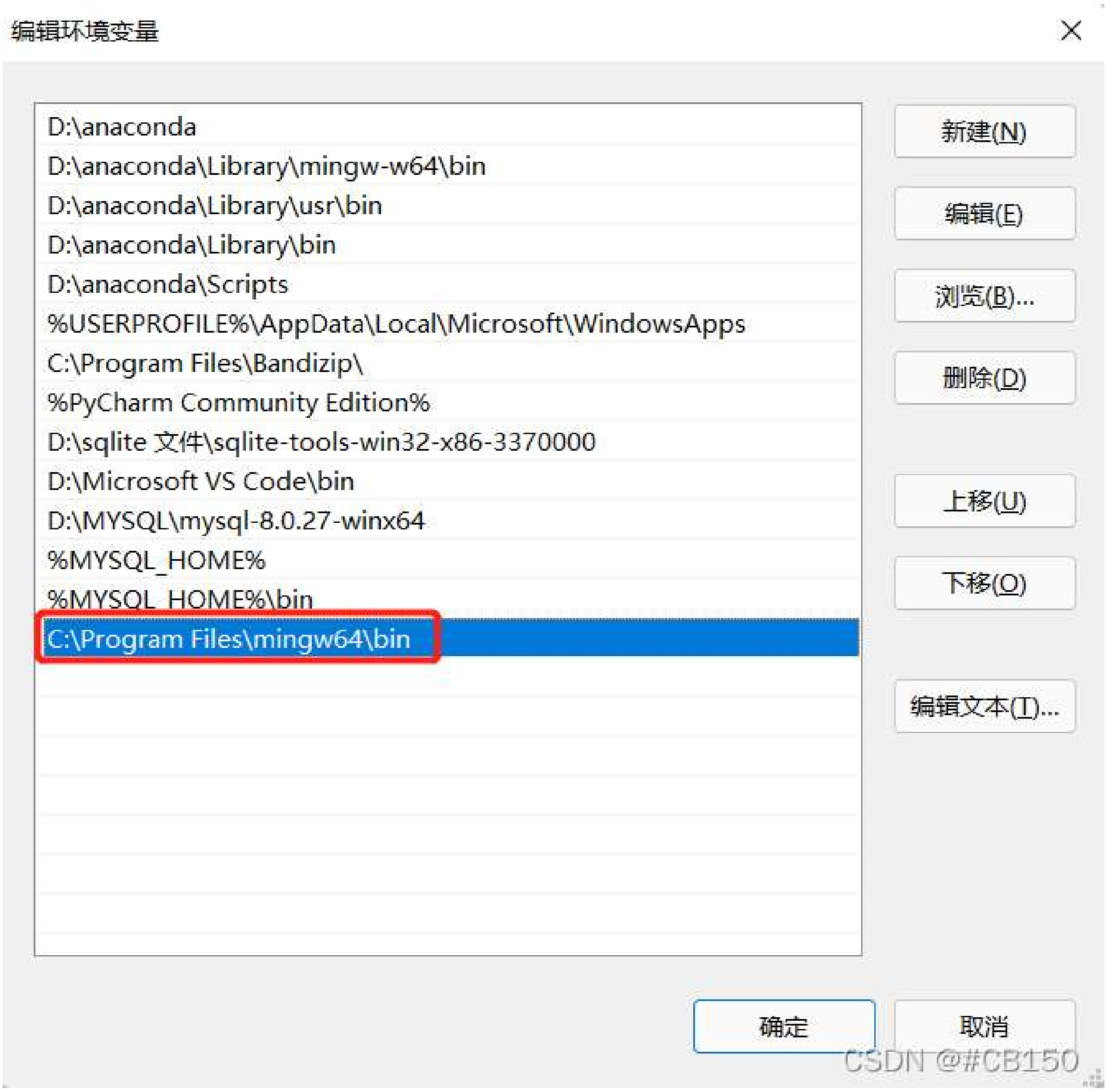


Program Files > mingw-w64 > x86_64-8.1.0-release-win32-seh-rt_v6-rev0 > mingw64 >

| 名称 | 修改日期 | 类型 | 大小 |
|---|---|---|---|
| bin | 2018/5/13 5:28 | 文件夹 | |
| etc | 2018/5/13 5:03 | 文件夹 | |
| include | 2018/5/13 5:28 | 文件夹 | |
| lib | 2018/5/13 5:29 | 文件夹 | |
| libexec | 2018/5/13 4:14 | 文件夹 | |
| licenses | 2018/5/13 5:29 | 文件夹 | |
| opt | 2018/5/13 4:42 | 文件夹 | |
| share | 2018/5/13 5:29 | 文件夹 | |
| x86_64-w64-mingw32 | 2018/5/13 1:26 | 文件夹 | |

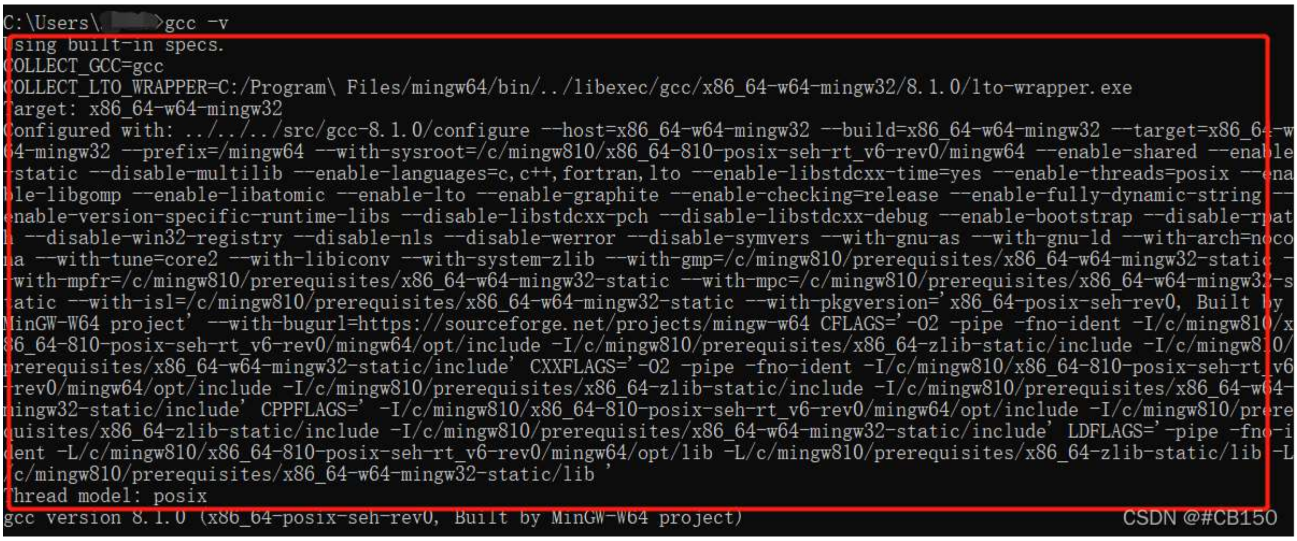
# 5.将mingw64添加到环境变量

将下载好的mingw64解压到指定的目录,我的路径是C:\Program Files\mingw64,然后配置环境变量 将目录C:\Program Files\mingw64\bin加入到环境变量path里（注意修改为自己的mingw64路径，具体到bin文件夹）



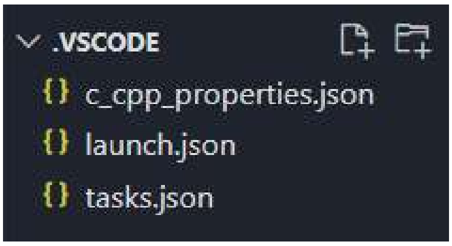
D:\anaconda
D:\anaconda\Library\mingw-w64\bin
D:\anaconda\Library\usr\bin
D:\anaconda\Library\bin
D:\anaconda\Scripts
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
C:\Program Files\Bandizip\
%PyCharm Community Edition%
D:\sqlite 文件\sqlite-tools-win32-x86-3370000
D:\Microsoft VS Code\bin
D:\MYSQL\mysql-8.0.27-winx64
%MYSQL_HOME%
%MYSQL_HOME%\bin
C:\Program Files\mingw64\bin

打开命令行输入`gcc -v`，出现以下信息说明mingw64配置成功



```
C:\Users\[redacted]>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=C:/Program\ Files/mingw64/bin/../libexec/gcc/x86_64-w64-mingw32/8.1.0/lto-wrapper.exe
Target: x86_64-w64-mingw32
Configured with: ../../../src/gcc-8.1.0/configure --host=x86_64-w64-mingw32 --build=x86_64-w64-mingw32 --target=x86_64-w
64-mingw32 --prefix=/mingw64 --with-sysroot=/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64 --enable-shared --enable
-static --disable-multilib --enable-languages=c,c++,fortran,lto --enable-libstdcxx-time=yes --enable-threads=posix --ena
ble-libgomp --enable-libatomic --enable-lto --enable-graphite --enable-checking=release --enable-fully-dynamic-string --
enable-version-specific-runtime-libs --disable-libstdcxx-pch --disable-libstdcxx-debug --enable-bootstrap --disable-rpat
h --disable-win32-registry --disable-nls --disable-werror --disable-symvers --with-gnu-as --with-gnu-ld --with-arch=noco
na --with-tune=core2 --with-libiconv --with-system-zlib --with-gmp=/c/mingw810/prerequisites/x86_64-w64-mingw32-static -
-with-mpfr=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-mpc=/c/mingw810/prerequisites/x86_64-w64-mingw32-s
tatic --with-isl=/c/mingw810/prerequisites/x86_64-w64-mingw32-static --with-pkgversion='x86_64-posix-seh-rev0, Built by
MinGW-W64 project' --with-bugurl=https://sourceforge.net/projects/mingw-w64 CFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x
86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86_64-zlib-static/include -I/c/mingw810/
prerequisites/x86_64-w64-mingw32-static/include' CXXFLAGS='-O2 -pipe -fno-ident -I/c/mingw810/x86_64-810-posix-seh-rt_v6
-rev0/mingw64/opt/include -I/c/mingw810/prerequisites/x86_64-zlib-static/include -I/c/mingw810/prerequisites/x86_64-w64-
mingw32-static/include' CPPFLAGS=' -I/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/include -I/c/mingw810/prere
quisites/x86_64-zlib-static/include -I/c/mingw810/prerequisites/x86_64-w64-mingw32-static/include' LDFLAGS='-pipe -fno-i
dent -L/c/mingw810/x86_64-810-posix-seh-rt_v6-rev0/mingw64/opt/lib -L/c/mingw810/prerequisites/x86_64-zlib-static/lib -L
/c/mingw810/prerequisites/x86_64-w64-mingw32-static/lib '
Thread model: posix
gcc version 8.1.0 (x86_64-posix-seh-rev0, Built by MinGW-W64 project)
```

## 4.新建.vscode文件夹 在该文件夹创建三个文件



.VSCODE

c_cpp_properties.json

launch.json

tasks.json

点这里可以直接下载 再修改路径

.vscode文件下载

# 注意：c_cpp_properties.json和launch.json 需要修改路径

## （1）c_cpp_properties.json配置

```
{
    "configurations": [
        {
            "name": "Win32",
            "includePath": [
                "${workspaceRoot}",
                "C:\\Program Files\\mingw64"
            ],
            "defines": [
                "_DEBUG",
                "UNICODE",
                "__GNUC__=6",
                "__cdecl=__attribute__((__cdecl__))"
            ],
            "intelliSenseMode": "windows-gcc-x64",
            "browse": {
                "limitSymbolsToIncludedHeaders": true,
                "databaseFilename": "",
                "path": [
                    "${workspaceRoot}",
                    "C:\\Program Files\\mingw64"
                    //此处修改为自己mingw64的路径
                ]
            }
        }
    ],
    "version": 4
}
```

## （2）launch.json配置

```
{
    "version": "0.2.0",
    "configurations": [
        {
            "name": "(Windows) Launch",
            "type": "cppvsdbg",
            "request": "launch",
            "program": "cmd",
            "preLaunchTask": "echo",
            "args": [
                "/C",
                "${fileDirname}\\${fileBasenameNoExtension}.exe",
                "&",
                "echo.",
                "&",
                "pause"
            ],
            "stopAtEntry": false,
            "cwd": "${workspaceFolder}",
            "environment": [],
            "externalConsole":true
        },
        {
            "name": "(gdb) Launch",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/${fileBasenameNoExtension}.exe",
            "args": [],
            "stopAtEntry": false,
            "cwd": "${workspaceFolder}",
            "environment": [],
            "externalConsole": true,
            "MIMode": "gdb",
            "miDebuggerPath": "C:\\Program Files\\mingw64\\bin\\gdb.exe",//修改为自己电脑的gdb路径
            "preLaunchTask": "echo",//这里和task.json的label相对应
            "setupCommands": [
                {
                    "description": "Enable pretty-printing for gdb",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": true
                }
            ]

        }
    ]
}
```

### （3）tasks.json配置

```
{
    // See https://go.microsoft.com/fwlink/?LinkId=733558
    // for the documentation about the tasks.json format
    "version": "2.0.0",
    "tasks": [
        {
            "label": "echo",
            "type": "shell",
            "command": "gcc",
            "args": [
                "-g",
                "${file}",
                "-o",
                "${fileBasenameNoExtension}.exe",
                "-fexec-charset=GBK"//解决中文乱码
            ]
        }
    ],
    "presentation": {
        "echo": true,
        "reveal": "always",
        "focus": false,
        "panel": "shared",
        "showReuseMessage": true,
        "clear": false
    }
}
```

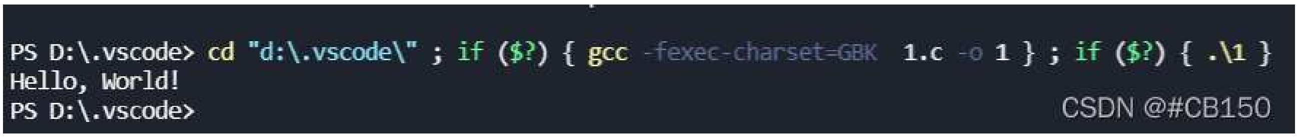
## 5.代码编译与运行

```
#include <stdio.h>

int main()
{
   printf("Hello, World! \n");

   return 0;
}
```

## ctrl+alt+n 运行



```
PS D:\.vscode> cd "d:\.vscode\" ; if ($?) { gcc -fexec-charset=GBK  1.c -o 1 } ; if ($?) { .\1 }
Hello, World!
PS D:\.vscode>
```

CSDN @#CB150

## vscode配置c语言环境就完成了！

如果打印中文乱码，请看下面这篇文章

vscode 解决C语言printf打印中文乱码问题