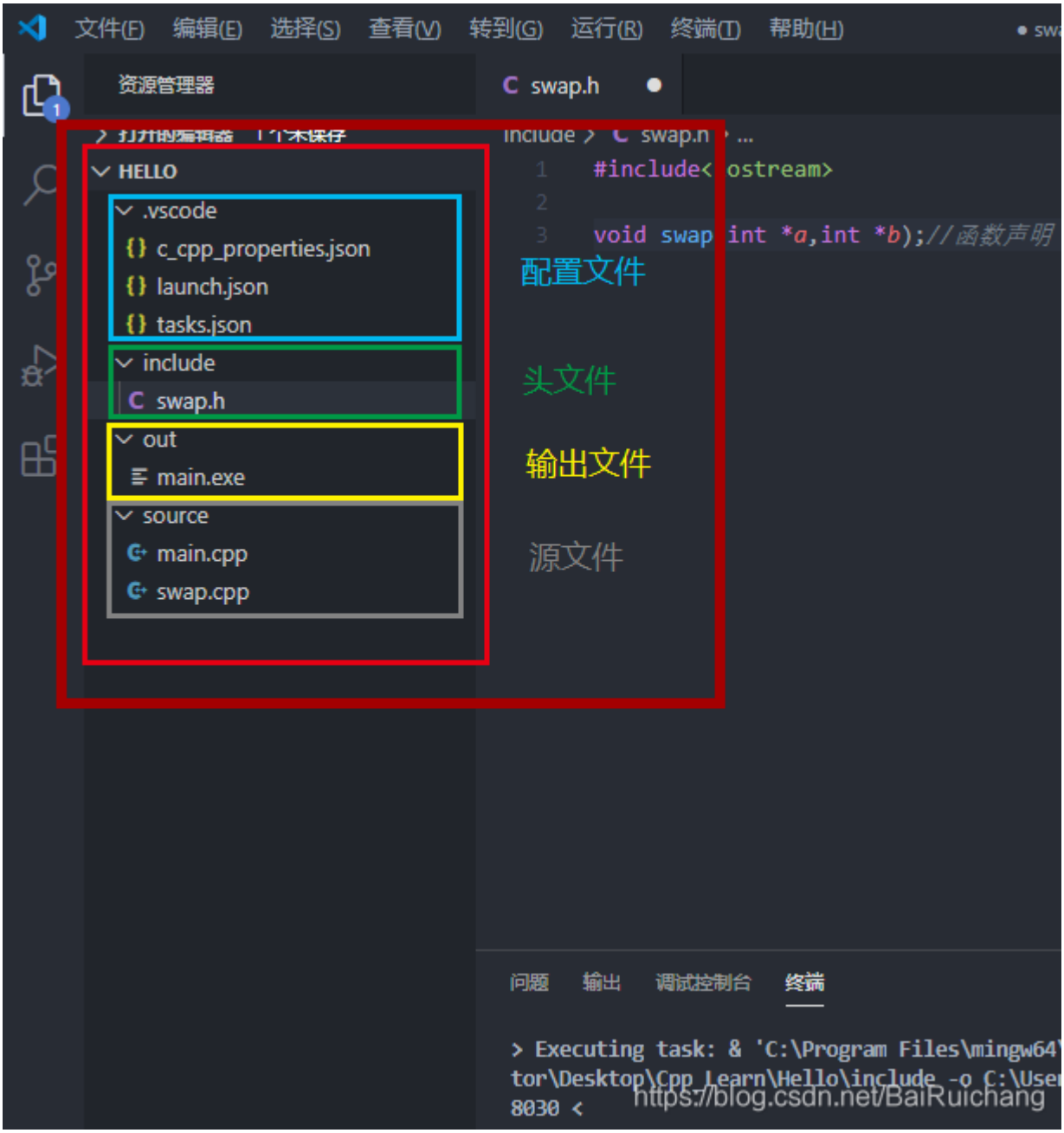
VSCode下c++多文件夹项目编译调试（非makefile）

2020-06-11 04:34 3071 查看

VScode默认配置文件只能编译单cpp文件。若是需要多文件编译或者需要分别设置Include头文件夹、Source资源文件夹（如下图），则需要修改配置三个.json文件(tasks.json、launch.json、c_cpp_properties.json)

> 注：个人习惯将.h头文件放到Include目录、.c.cpp源文件放到Source文件夹下面，输出文件.exe文件放到out文件夹下面



.vscode文件夹存放.json文件夹，实际使用中可以直接拷贝过来使用，而没必要每次都新建修改一遍

c_cpp_properties.json配置文件默认是不会产生的，快捷键ctrl+shift+p 再输入configuration便会出现

默认工作空间只有.vscode文件夹，.cpp文件直接放在工作空间根目录的。示例中include、source以及out文件夹均需要先分创建才能在.json中修改配置

修改.json文件

tasks.json文件

需要修改的地方主要有三处，可直接复制使用

1.修改 -g后面的目录

2. 修改-o后面的目录

3. 添加 -I 及后面的目录

[plain]

```
{
"tasks": [
{
"type": "shell",                        //任务执行的是shell命令
"label": "g++.exe build active file", //和launch.josn 中的 preLaunchTask 必须一样
"command": "C:\\Program Files\\mingw64\\bin\\g++.exe", //命令是g++，也可以直接写g++
"args": [
"-g",    //生成和调试有关的信息
"-Wall", // 开启额外警告
"${workspaceFolder}\\source\\*.cpp",  //当前工作空间下文件夹source目录名下的所有cpp文件。 source对应工程目录下的source文件夹名字，可自行修改
"-I","${workspaceFolder}\\include",      // 参数-I 和工程路径 指明了项目中要引用的非标准头文件的位置。 include对应工程目录下的include文件夹名字，可自行修改
"-o",
"${workspaceFolder}\\out\\${fileBasenameNoExtension}.exe", //指定输出的文件名为out，默认a.exe。out对应工程目录下的out文件夹名字，可自行修改
"-std=c++17",                                   //使用c++17标准编译
"-finput-charset=UTF-8",                        //输入编译器默认文本编码 默认为utf-8
"-fexec-charset=GB18030",                       //输出exe文件编码
],
// 所以以上部分，就是在shell中执行（假设文件名为main.cpp）  等同在shell中执行  g++ main.cpp -o filename.exe 等命令
"options": {
"cwd": "C:\\Program Files\\mingw64\\bin"
}
}
],
"version": "2.0.0"
}
```

launch.json文件

需要修改的地方有以下：

1. "program"目录
2. “cwd”

注意："miDebuggerPath"为自己编译器的bin文件目录，（就是添加到环境变量的那个目录）

[plain]

```
{
"version": "0.2.0",
```

```
"configurations": [
{
"name": "g++.exe",
"type": "cppdbg",
"request": "launch",

"program": "${workspaceFolder}\\out\\${fileBasenameNoExtension}.exe",  //需要运行/调试的是当前打开文件的目录中，名字和当前文件相同，但扩展名为exe的程序。和tasks.json中-o后面的目录一样的
"args": [],
"stopAtEntry": false,           //设为true时程序将暂停在程序入口处（即停止main函数开始），一般设置为false
"cwd": "${workspaceFolder}\\out", //调试程序时的工作目录 。out对应工程目录下的out文件夹
"environment": [],              //针对调试的程序，要添加到环境中的环境变量
"externalConsole": true,        //调试时是否显示外置控制台窗口（大黑框），一般设置为true显示控制台

"internalConsoleOptions": "neverOpen",   //可以没有
"MIMode": "gdb",                // VSCode要使用的调试工具or指示VS代码将连接到的调试器
"miDebuggerPath": "C:\\Program Files\\mingw64\\bin\\gdb.exe",   // miDebugger的路径，注意这里要与MinGw的路径对应 当未指定时，它将搜索操作系统的PATH变量来寻找调试器
"setupCommands": [
{
"description": "为 gdb 启用整齐打印",
"text": "-enable-pretty-printing",
"ignoreFailures": true
}
],
"preLaunchTask": "g++.exe build active file"  // 调试会话开始前执行的任务，一般为编译程序，c++为g++，c为gcc 这个名字一定要跟tasks.json中的任务名字大小写一致
}
]
}
```

c_cpp_properties.json

默认不会产生。快捷键ctrl+shift+p 再输入configuration便会出现。
修改的地方只有一处：

“includePath” 将include文件夹添加进去即可，注意格式！！
注：“compilerPath” 同launch.json。为编译器文件目录

[plain]

```
{
"configurations": [
{
"name": "Win32",
"includePath": [
"${workspaceFolder}/**",
"${workspaceFolder}\\include\\**"
],
"defines": [
```

```
"_DEBUG",
"UNICODE",
"_UNICODE"
],
"compilerPath": "C:\\Program Files\\mingw64\\bin\\gdb.exe",
"intelliSenseMode": "clang-x86"
}
],
"version": 4
}
```

总结

这三个配置文件可以各拷贝一份，新建工程时，直接放.vscode下面。软件在打开时会直接读取.json文件。

.vscode通常就是放配置文件的，除这三个常用的之外还有settings.json，用来配置编辑器等外观性质的东西。

VSCode下c++多文件夹项目编译调试还可以用makefile、cmake等工具实现，适用于大型项目文件时使用

标签：