* 图片名 命名规则：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pictures** | Albums | Camera | Comics | Images | Photos |
| 相册 | 纪实 | 漫画 | 唯美 | 设计 |

1. [四个汉字] 作为帐号头像、照片墙、背景等特定用途，其它不可与之冲突。
2. [人/物名] 动漫人物、娱乐明星、美眉才俊、艺术作品、创意设计等；

[人/物名01……] 相同人/物的不同照片（起始为0，到99；最多可存储100张）；

[人/物名+/ /+原名或外文名] 避免四字人/物名与特定的四个汉字文件名相冲突。

1. [生活朋友名+/~/昵称+/-/住址所在省市] 纪实生活中接触到的人。
2. [任意名] 以最为体现图片内容、最为切实合理分类为主。

e.g.

樱梦雪儿.jpg

艾米莉娅炭.jpg，艾米莉娅炭01.jpg；栗山未来 くりやま みらい.jpg，栗山未来01.jpg

史海燕~小乖-河南洛阳.jpg

高音八孔竖笛指法表（德国式）.jpg

* 影视内容 编辑规则：

World Opera.docx

影视结构：<>说明用语 []可选操作

<标题>：中文名在书名号，原文名在小括号，作者国在中括号，原作导演优先分。

[<设定>]：现实名词加底纹，二次元语用原样。

[<剧集>]：两位数保持工整，中文名带原语名。

<人物>：尽量括号带原名，关系添加下划线：情侣中间用红心，等号左右为称谓。

<语句>：数字顿号标号排，同为一人放一块，引用人名放在前，可能带上原语言。

[<感悟>]：人物感言加冒号，对其评语加括号，本身体会直接写，句末换行与句号。

[<插图>]：原图居中不缩放，体现场景或人物。

1. 标题格式：**《影视名》 [国籍]作者1、作者2、……**
2. 作品名称：**《汉语名 ±外文名》（原版名）**
3. 注释符号：“绰号”，（注解），[愿望/国籍/地名]，〔武器或技能〕，<绝招>
4. 名言警句：1）中文：宋体；2）英文：Times New Roman；3)日文：MS Mincho

……。——人名

原文语言（或翻译外文）

1. X引用Y的话：Y：……。——X
2. ~~红色删除线~~：人物消失
3. 默认字体：宋体，五号字，单倍行距；影视标题：宋体，四号字，加粗

e.g. **《花开伊吕波》**

**四一、 《香肠派对》（Sausage Party） [美]康拉德•弗农、格雷**

**格•蒂曼**

**五〇、 《Re：从零开始的异世界生活》（Re：ゼロから始める異世界生活）（Re: Life in a different world from zero） [日]長月達平**

* 1. 决定成败的是人信念的强弱……而不是帮手的多少。——~~莱姆斯·约翰·卢平~~

It is the quality of one’s convitions that determine his success…… not the number of followers.——Remus John Lupin

* 1. 玛莎：没有东西是比免费更贵的。——不动游星
  2. 憎恨的尽头是一片虚无。——武藤游戏

憎しみの先には何もないよ。——むとう　ゆぎ

* 音乐名 命名规则：

Music

1. 歌曲原唱：[演唱者+/ - /+歌曲名]
2. 歌曲翻唱：[演唱者+/ - /+歌曲名+/ = /+原唱]

e.g. 田馥甄 - 小幸运.flac

冯提莫 - 刚好遇见你 = 李玉刚.flac

* 编程语言 命名规范：

1. **常量**：字母全部大写，多个单词则中间用下划线“\_”连接。宏定义的符号常量 “#define 大写宏名 字符值”，const修饰的常量 “const 数据类型 大写字母 = 数值;”。
2. **变量**：使用骆驼命名法，舍弃匈牙利命名法，不加类型前缀（除指针加“p”，引用加“r”外），数组多使用复数；局部变量直接以有意义的单词命名，多个单词则从第二个开始首字母大写；全局变量加上“g\_”前缀，单个单词用小写，多个则每个单词首字母大写；可以考虑使用小写字母+下划线的形式命名。
3. **函数名**：单个单词用小写，多个单词每个首字母均大写，采用动宾形式；可以考虑使用小写字母+下划线的形式命名。
4. **类**：
5. **类名**，默认只一个首字母大写的单词，可以例外；
6. **数据成员**，默认只一个字母小写的单词，多个则与局部变量类似，但是末尾要加后缀下划线“\_”以示与普通局部变量相区别；
7. **数据成员的get set方法**，使用get/set+数据成员名（不加后缀下划线，首字母大写）的形式，get方法定义函数体为const，set方法定义形参多为const的引用（常见于类），可以例外。
8. **构造函数**，与类名相同，考虑带默认形参值减少函数定义，与声明分离的实现不能再带有默认形参值；
9. **复制构造函数**，与类名相同，函数形参定义为类的常引用 “const 类& 变量名”
10. **析构函数**，采用“virtual”修饰的虚析构函数；
11. **成员函数**，一般使用两个且首字母均大写的单词，动宾形式。
12. **结构体/共用体**：类型说明名与类类似，数据变量名与局部变量类似。
13. **枚举**：类型说明名与类类似，枚举元素采用单个首字母大写的形式。
14. **语句标准**：操作符与操作数以空格分隔（单目操作符、指针->除外）；函数定义之间至少一个空行；函数形参表多个变量以空格分开；if while for与( )以空格间隔。
15. **注释要求**：尽量使用Tab跳格；变量定义后面的说明用“//”；函数定义前面的说明用“/\*\*/”；分阶段算法在之前一行同一列使用“//”简要阐述。

e.g.

#define WIDTH 800 //窗口的宽（默认值） 800

const double PI = 3.14159265358979323846; //圆周率

const int RECT\_WIDTH = 200; //圆角矩形按钮的宽 200

int copy = 1;

vector<Poker> pokersTemp; //临时卡组存储

POINT pts\_pentagon[5];

int g\_hurt; //伤害计数器

RECT g\_rectOtherLife = { 20, 24, 200, 84 }; //对方生命数值区

TCHAR g\_SelfName[20] = \_T("Jack"); //己方玩家名字

extern void ExitProgram(int status = 0); //退出程序

extern void transp\_image(int dstX, int dstY, IMAGE \*pimg, int avoid\_color, int deviation, int tp);

class Poker //扑克牌Poker类

{

public:

//构造函数

Poker() {}

Poker(LPCTSTR pImgFile, int nWidth = 0, int nHeight = 0); //构造时获取图像

//复制构造函数

Poker(const Poker& rPoker) {

point\_ = rPoker.point\_;

color\_ = rPoker.color\_;

image\_ = rPoker.image\_;

//pImageFile\_ = rPoker.pImageFile\_;

}

//数据成员的 get set 方法

int getPoint() const { return point\_; } //point\_

void setPoint(const int point) { point\_ = point; }

int getColor() const { return color\_; } //color\_

void setColor(const int color) { color\_ = color; }

IMAGE getImage() const { return image\_; } //image\_

void setImage(const IMAGE& image) { image\_ = image; }

//LPCTSTR getImageFile() const { return pImageFile\_; } //pImageFile\_

//void setImageFile(const LPCTSTR pImgFile){ pImageFile\_ = pImgFile; }

void ObtainCard(LPCTSTR pImgFile, int nWidth = 0, int nHeight = 0); //获取卡牌

void PlaceCard(int dstX, int dstY, int dstWidth = 80, int dstHeight = 120); //放置卡牌（点的坐标）

void PlaceCard(const CardButton& rCbutton, int dstWidth = 80, int dstHeight = 120); //放置卡牌（CardButotn对象）

//bool operator ==(const Poker& rPoker); //等于==操作符重载，实现图像匹配

private:

int point\_; //扑克牌点数

int color\_; //扑克牌花色

IMAGE image\_; //扑克牌图像

};

const enum Color { Spade = 1, Heart = 2, Club = 3, Diamond = 4 }; //扑克牌花色

/\* 图像透明处理

排除颜色为avoid\_color，容差为deviation，透明度tp(transparency)从0到100 \*/

void transp\_image(int dstX, int dstY, IMAGE \*pimg, int avoid\_color, int deviation, int tp)

{ }

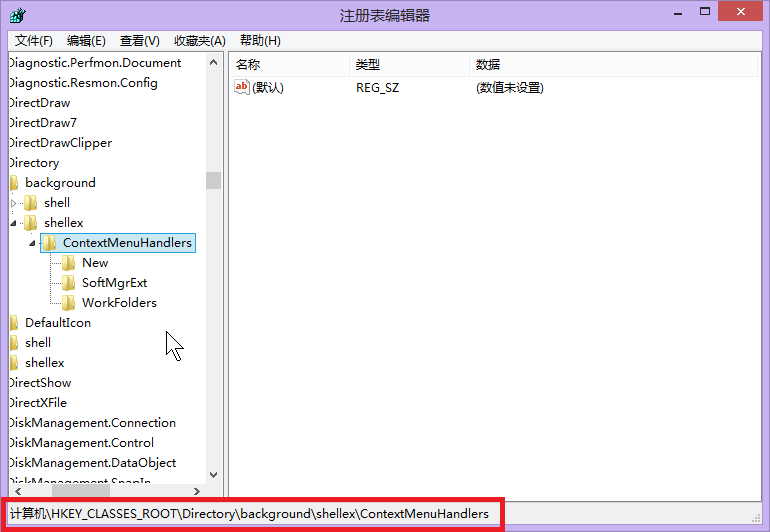
for (y = 0; y < pimg->getheight(); y++)

{ }

if (false == g\_cbOtherHand[show].getExit())

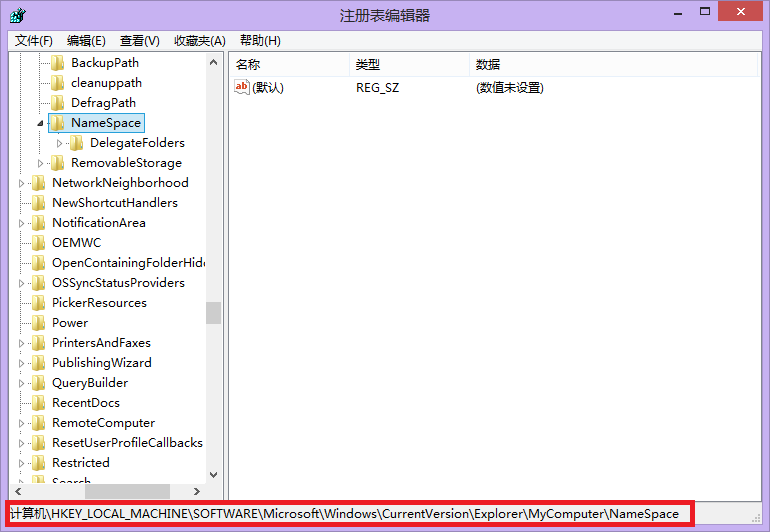
{ }

1. **删除右键图形选项&属性&显卡(Windows8.1 X64)**



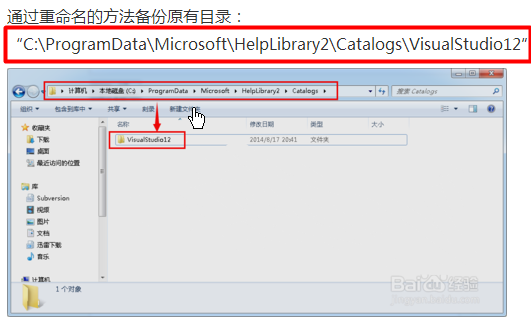
计算机\HKEY\_CLASSES\_ROOT\Directory\Background\shellex\ContextMenuHandlers

1. **删除资源管理器六个文件夹(Windows8.1 X64)**

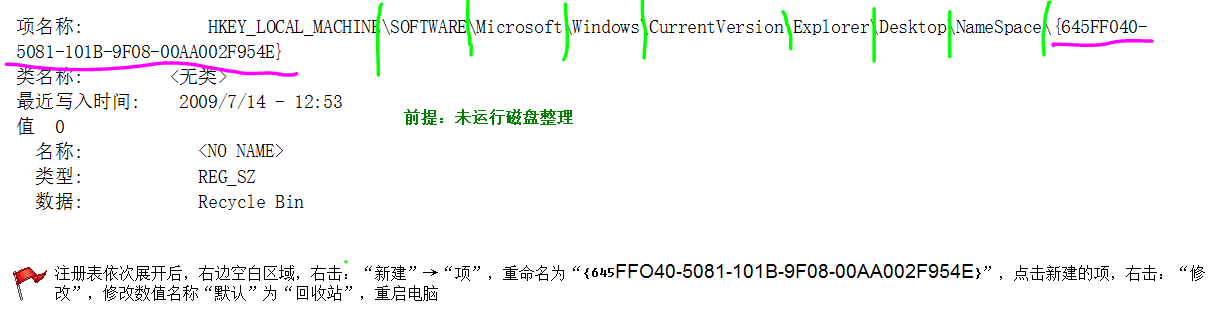


计算机\HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\MyComputer\NameSpace

1. **VS2013离线版MSDN安装**



1. **恢复误删且清空回收站的文件**



1. **修复文件”\*.dll”缺失**

①缺什么在“找DLL下载站<http://www.zhaodll.com/>”中下载，

②存放至系统目录下

C:\Windows\System (Windows 95/98/Me)

C:\WINNT\System32 (Windows NT/2000)

C:\Windows\System32 (Windows XP, Vista)

C:\Windows\System32 (Windows 7/8/2008r2)

C:\Windows\SysWOW64 (Windows 7/8/2008r2)，

③按”Windows+R”在运行框中输入”regsvr32 <\*.dll>”并确定。

如：系统缺失文件”SbieDll.dll”，“运行“后输入”regsvr32 SbieDll.dll”确定即可。

1. **解决component ‘\*.OCX’控件出错**

①下载\*.OCX文件（百度百科文末附有文件绿色纯净版链接），

②32位系统将其复制到 C:\windows\system32\ 目录下，

64位系统将其复制到 C:\Windows\SysWOW64\ 目录下

③用“管理员身份”运行“命令提示符”，

④32位系统输入 regsvr32 C:\Windows\system32\comdlg32.ocx 回车，

64位系统输入 regsvr32 C:\Windows\SysWOW64\comdlg32.ocx 回车

如：控件tabctl32.ocx出错，回车后弹出提示框



1. **驱动精灵硬件检测报告**

版本:2014.12.25.1317(8.0.1225.1317)

======================================================================

**硬件概要**

CPU: 英特尔 Core i5-3320M (双核)

主板: LENOVO 68852F7 (英特尔 HM77 (Panther Point))

内存: 4 GBytes

显卡: 英特尔 HD Graphics 4000, 1678156 KBytes

硬盘: WDC WD5000LPVX-08V0TT5

显示器: 联想 N156B6-L07

网卡: 博通 BCM43142 802.11b/g/n Wireless Network Adapter

声卡: 英特尔 Panther Point PCH - High Definition Audio Controller [C1]

**处理器信息**

处理器: 英特尔(R) Core(TM) i5-3320M CPU @ 2.60GHz

运行速度: 2600.0 MHz

核心/线程: 双核, 四线程

核心代号: Ivy Bridge-MB PGA SV

功耗: 35.0 W

插座: rPGA988B

一级缓存: 指令: 32 KBytes, Data: 32 KBytes

二级缓存: 集成: 256 KBytes

三级缓存: 3 MB

特性: MMX　SSE　SSE-2　SSE-3　SSSE-3　SSE4.1　SSE4.2　AVX　EMT64　VT　SMX　EIST　TM1　TM2　Turbo Boost

**主板信息**

主板厂商: 联想

主板型号: 联想 68852F7

芯片组: 英特尔 HM77 (Panther Point)

主板插槽: 3xPCI Express x1, 1xPCI Express x8

USB支持: v3.0

PCI-E支持: v3.0

BIOS版本: HEET42WW (1.23 )

BOIS日期: 01/27/2014

**内存信息(总计: 4 GBytes)**

内存大小: 4096 MB

内存类型: DDR3 SDRAM

制造商: Micron Tech.

制造日期: 2014年第24周

**显卡信息**

显卡1:

显卡芯片: 英特尔 HD Graphics 4000

显卡型号: 英特尔 Ivy Bridge-MB GT2 - Integrated Graphics Controller [E1/L1/N0/P0] [Lenovo]

显存大小: 1678156 KB

显卡BIOS版本: 2130 PC 14.34 12/23/2011 01:15:27

频率: 350.0 MHz

显卡2:

显卡芯片: NVIDIA（英伟达） GeForce GT 740M

显卡型号: nVidia GeForce GT 740M [联想]

显存大小: 2048 MB

显卡BIOS版本: 2130 PC 14.34 12/23/2011 01:15:27

**显示器信息**

显示器: 联想 N156B6-L07

生产日期: 2011年第1周

信号输入方式: Digital

屏幕水平尺寸: 34 cm

屏幕垂直尺寸: 19 cm

屏幕对角尺寸: 15英寸

**存储信息**

存储器1:

控制器: Serial ATA 6Gb/s

型号: WDC WD5000LPVX-08V0TT5

容量: 476,940 MB (500 GB)

转速: 5400 RPM

缓存: 8192 KBytes

NCQ功能: 支持, Max. Depth: 32

S.M.A.R.T.: 存在, Active

48bit LBA: 支持, Active

存储器2:

型号: HL-DT-ST DVDRAM GU90N

**网卡信息**

网卡1:

硬件名称: 博通 BCM43142 802.11b/g/n Wireless Network Adapter

MAC地址: 18-CF-5E-01-C6-7C

最大带宽: 13 Mbps

网卡2:

硬件名称: 瑞昱 Semiconductor RTL8168/8111 PCI-E Gigabit Ethernet NIC

MAC地址: 28-D2-44-B9-24-13

最大带宽: 1000 Mbps

**声卡信息**

硬件名称: Conexant

解码器: 英特尔 Panther Point PCH - High Definition Audio Controller [C1]

**电池信息**

电池制造商: SMP

电池型号: 45N1045

化学性质: 锂电池

设计容量: 42770 mWh

完全充电容量: 43730 mWh

损耗率: 0.0 %

**USB设备**

USB设备1:

厂商: 无

设备名称: USB 光电鼠标

USB版本: 1.10

驱动描述: USB

1. **Android odex反编译工程**

①新建文件夹，如：123，将要反编译的odex文件，如：Calculator.odex，和baksmali.jar, smali.jar两个jar包放在同一目录下（baksmali.jar和smali.jar下载链接：<https://bitbucket.org/JesusFreke/smali/downloads>）。

②为解决”Cannot locate boot class path file /system/framework/core.odex”问题，需要在当前文件夹下新建system/framework/目录，并拷贝手机system/framework/下的全部\*.odex文件到此目录下（手机上安装Root Explorer文件管理器，已被Root过，并授予RE文件管理器以Root权限，即可访问到此目录）。

③Windows+R->“运行”->cmd->进入到文件目录123下，输入” java -jar baksmali.jar -d ./system/framework –x Calculator.odex”，则在当前目录123下生成out文件夹；

④输入”java -jar smali.jar out -o classes.dex”，则在当前目录123下生成classes.dex文件

⑤用dex2jar将\*.dex文件反编译成\*.jar文件：

下载dex2jar-2.0，<https://sourceforge.net/projects/dex2jar/>。使用其中的bat命令：d2j-dex2jar.bat，在cmd下在当前目录下，输入” C:\Users\Snow-Angel\Desktop\dex2jar-2.0\dex2jar-2.0\d2j-dex2jar.bat classes.dex”，得到” dex2jar classes.dex -> .\classes-dex2jar.jar”结果，表示已反编译成jar文件。

⑥用jd-gui-windows-1.4.0打开class-dex-dex2jar-2.0.jar文件，即可得到原Java实现代码（jd-gui下载链接：<http://jd.benow.ca/>）。

1. **更改Windows桌面位置为D盘**

①在D盘新建目录：D:\Documents\Desktop；

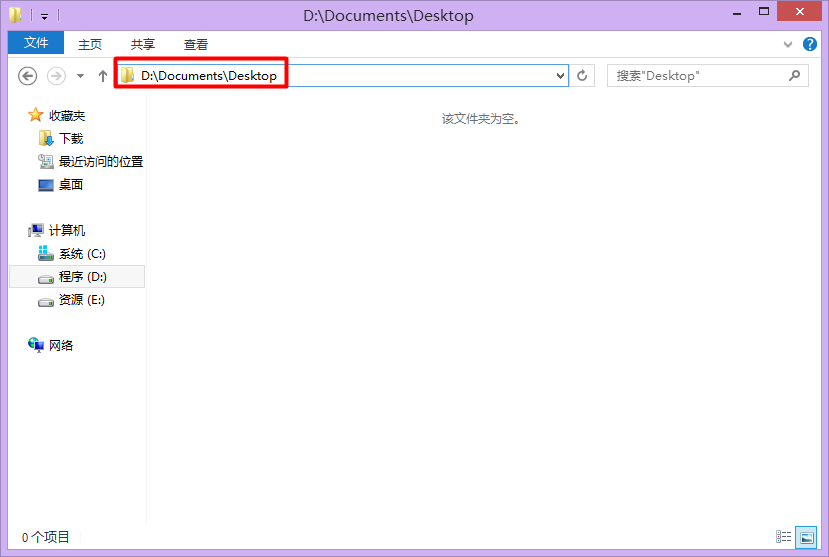
②进入注册表编辑器所示路径下：

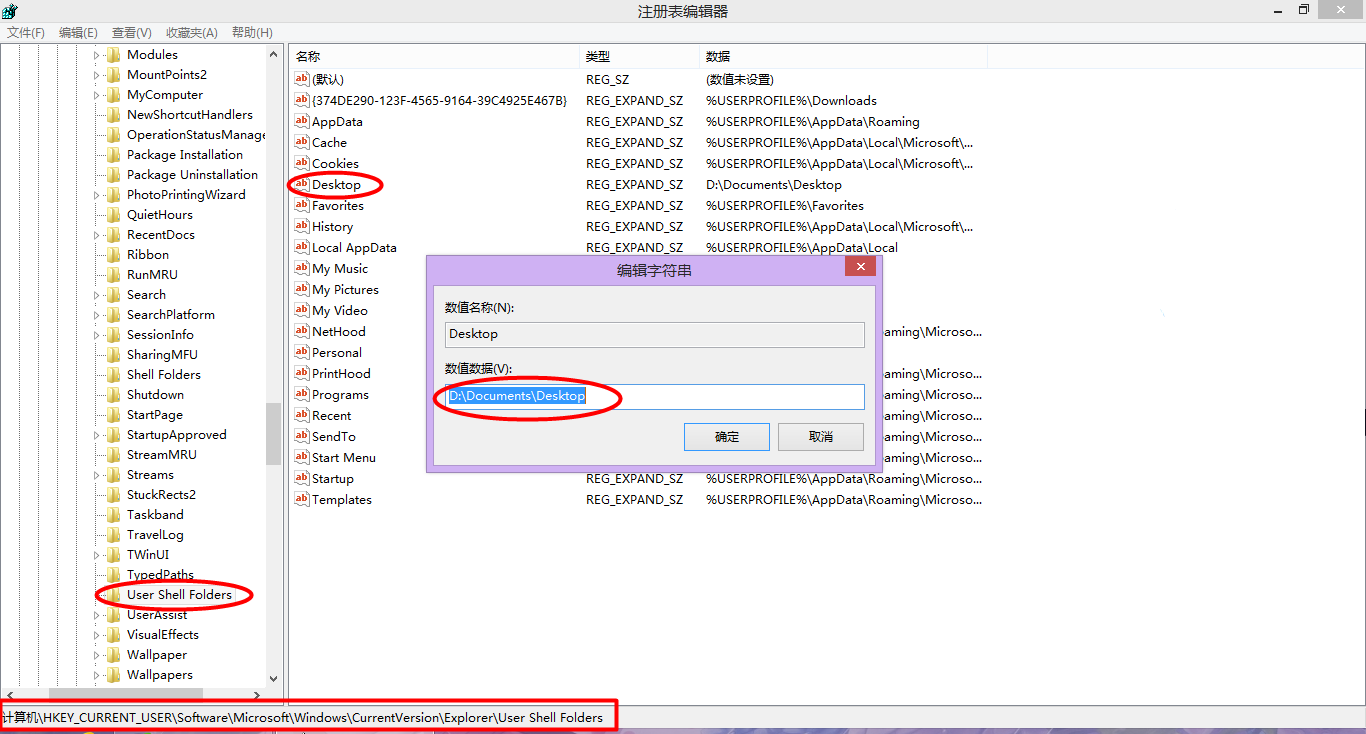
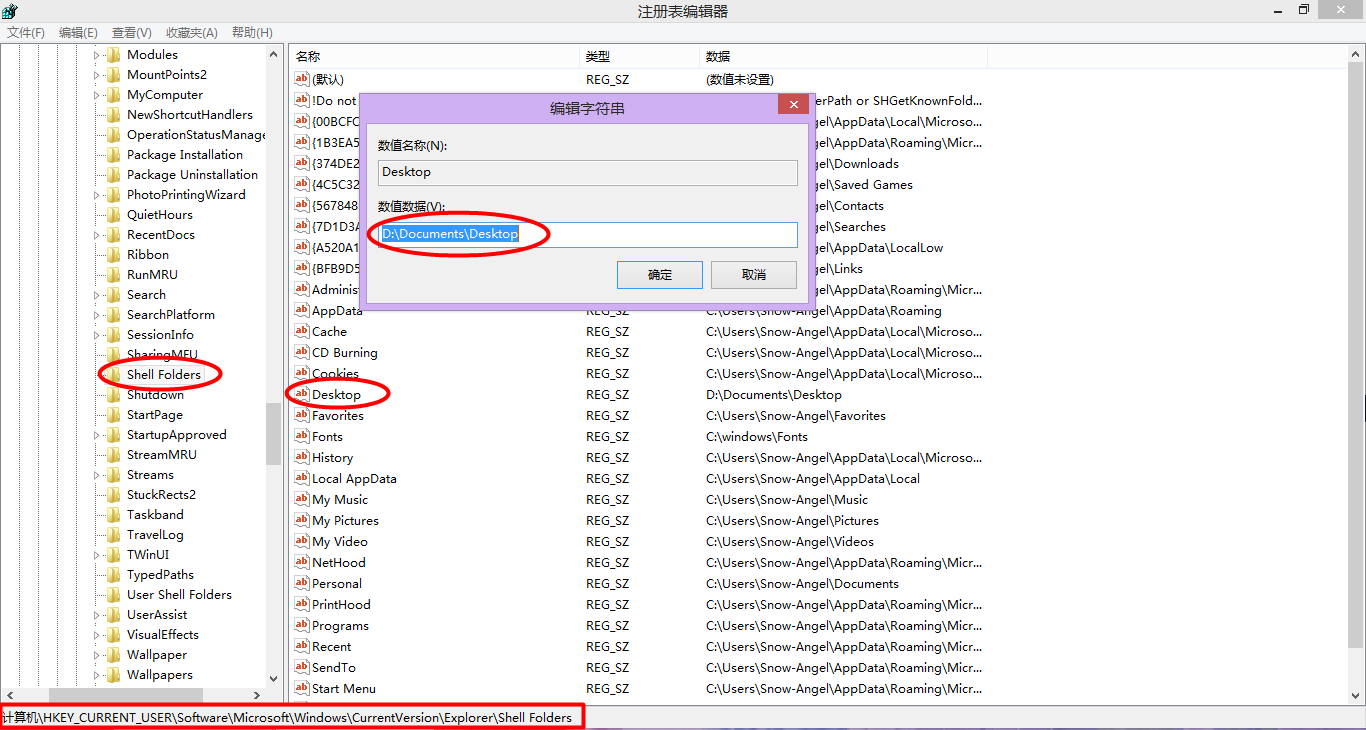
修改Shell Folders项的Desktop键的数值数据为D:\Documents\Desktop

（原始值为 C:\Users\Snow-Angel\Desktop）；

修改User Shell Folders项的Desktop键的数值数据为D:\Documents\Desktop。

（原始值为 %USERPROFILE%\Desktop）。





1. **"index of"直接访问网站内部文件**

index of是一串网络下载口令，可以直接进入网站首页的所有文件和文件夹，不必再通过HTTP的网页形式，从而突破网站下载限制。

搜索完整内容，使用“""”。

举例如下：在搜索框中输入

“"index of /" inurl:lib”，按搜索进入图书网站的资源列表；

“"index of /" cnki”，按搜索进入图书馆的CNKI、VIP、超星等入口。

百度搜索排名关键字：mpeg4，mp3，cnki，rmvb，rm，movie，swf，jpg，admin，pdf，doc，wmv，mdb，mpg，mtv，software，mov，asf，lib，vod，rar，exe，iso，video，book，soft

，chm，password，game，music，dvd，mid，book，download，……

1. **VS2017+OpenGL环境的配置**

OpenGL=Open Graphics Library开放图形库

(1) 配置环境

①下载OpenGL所需要的文件（glutdlls37beta.zip）：

http://www.opengl.org/resources/libraries/glut/glutdlls37beta.zip

然后得到5个文件：glut.dll glut32.dll glut.lib glut32.lib glut.h。

②找到目录..\Microsoft VisualStudio\2017\Professional\VC\Tools\MSVC\14.10.25017\include\gl（没有的话就建立一个），将glut.h放到里面；

找到目录..\Microsoft Visual Studio\2017\Professional\VC\Tools\MSVC\14.10.25017\lib\x86，将glut.lib和glut32.lib放到里面；

最后把glut.dll和glut32.dll放到

C:\Windows\system32（32位系统）或

(2) 隐藏控制台黑窗口

Console Application下不显示控制台黑窗口只显示Windows窗口。

方法一、使用#pragma预处理命令：

在程序开头的#include后面加上

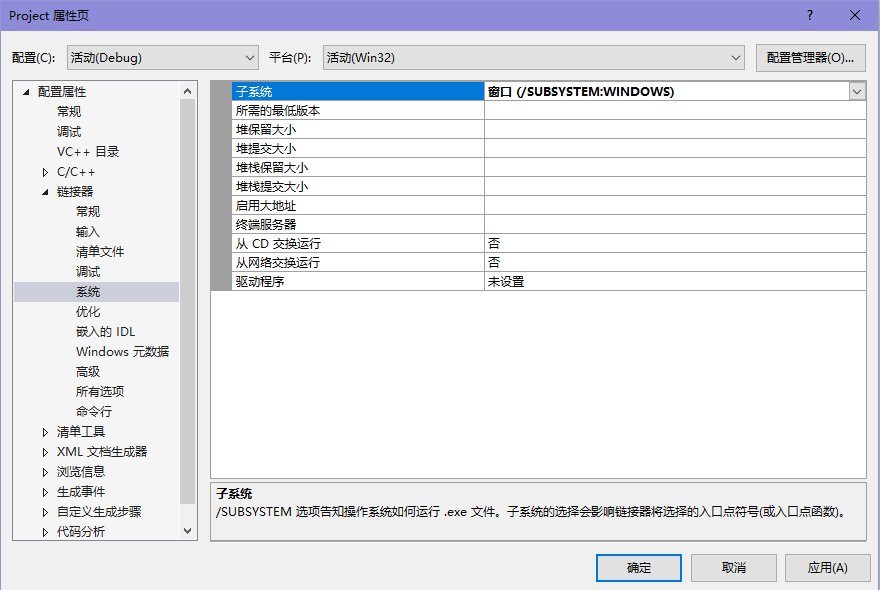
#pragma comment(linker, "/subsystem:windows /entry:mainCRTStartup")

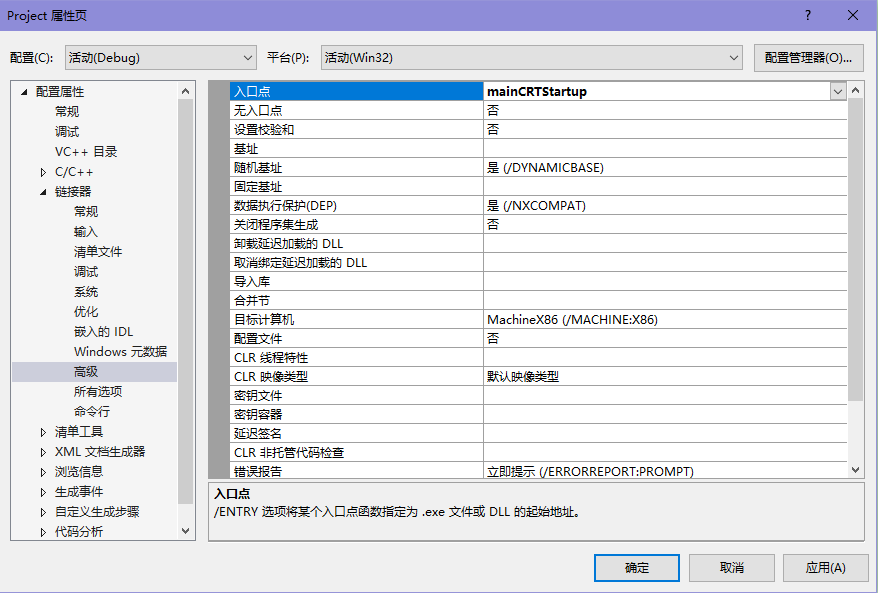
方法二、修改项目属性：

项目->配置属性->链接器，

在“系统”选项中，修改“子系统”选择为“窗口 (/SUBSYSTEM:WINDOWS)”；

在“高级”选项中，修改“入口点”填写为“mainCRTStartup”。





说明：

Windows下，不同的子系统会链接不同的主函数，控制台的会链接main,窗口的会链接WinMain，如果不匹配会链接失败。

测试代码：gl.cpp //画五角星

#include <GL/glut.h>

#include <math.h>

#pragma comment(linker, "/subsystem:windows /entry:mainCRTStartup")

const GLfloat Pi = 3.1415926536f;

void pentagon()

{

GLfloat a = 1 / (2 - 2 \* cos(72 \* Pi / 180));

GLfloat bx = a \* cos(18 \* Pi / 180);

GLfloat by = a \* sin(18 \* Pi / 180);

GLfloat cy = -a \* cos(18 \* Pi / 180);

GLfloat PointA[2] = { 0, a };

GLfloat PointB[2] = { bx, by };

GLfloat PointC[2] = { 0.5, cy };

GLfloat PointD[2] = { -0.5, cy };

GLfloat PointE[2] = { -bx, by };

//按照A->C->E->B->D->A的顺序，一笔画出五角星

glClear(GL\_COLOR\_BUFFER\_BIT);

glBegin(GL\_LINE\_LOOP);

glVertex2fv(PointA);

glVertex2fv(PointC);

glVertex2fv(PointE);

glVertex2fv(PointB);

glVertex2fv(PointD);

glEnd();

glFlush();

}

int main(int argc, char \*argv[])

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutInitWindowPosition(100, 100);

glutInitWindowSize(400, 400);

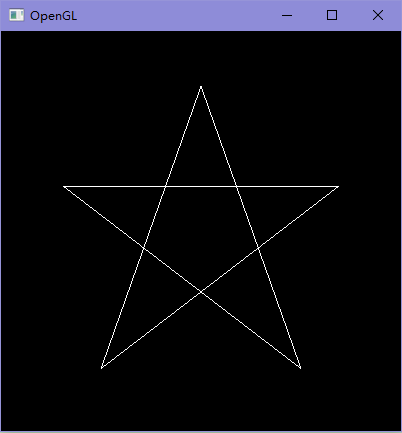
glutCreateWindow("OpenGL");

glutDisplayFunc(pentagon);

glutMainLoop();

return 0;

}



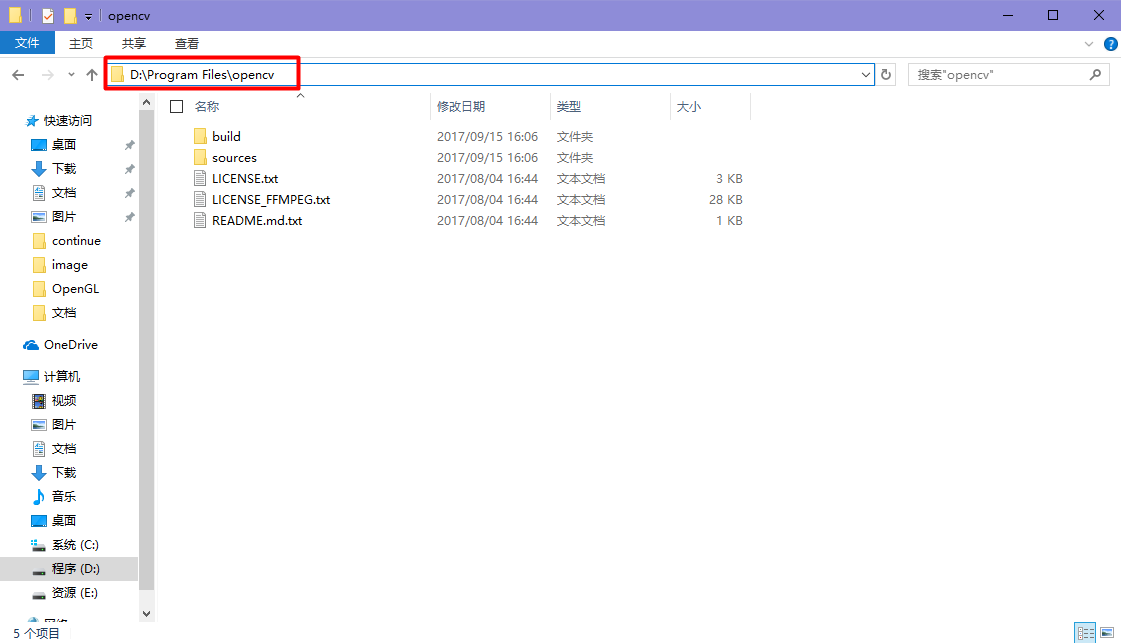
1. **VS2017+OpenCV环境的配置**

OpenCV=Open Source Computer Vision Library开源计算机视觉库

1. 下载OpenCV文件（OpenCV 3.3.0）：

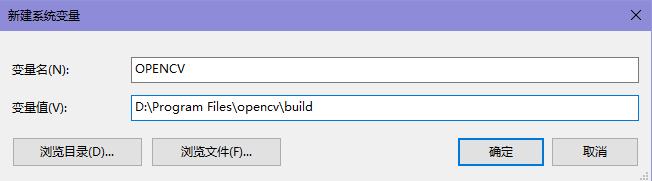
<http://opencv.org/releases.html>

双击安装。（本例将文件安装在D:\Program Files目录下）

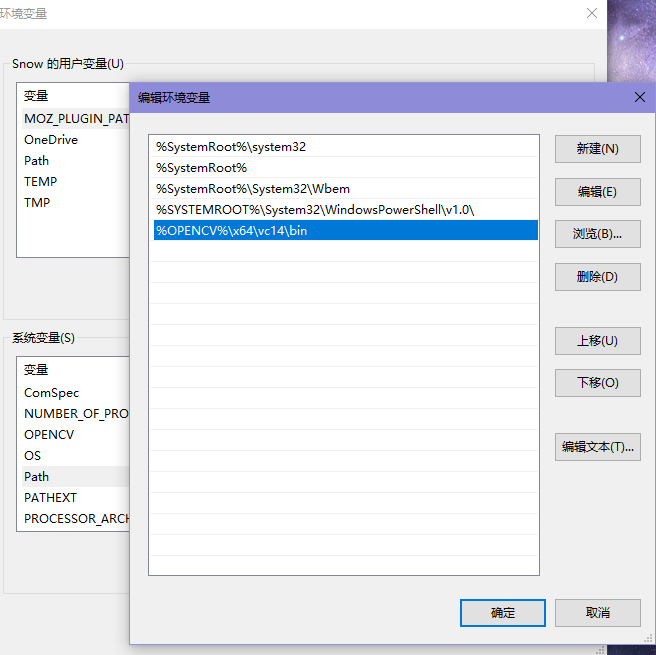


1. 设置环境变量：

新建系统变量

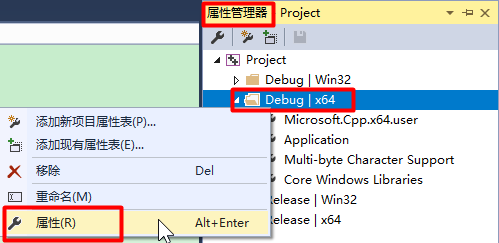


编辑系统变量Path，新建路径值



1. 新建Visual Studio空项目；

选择菜单“视图”->“其他窗口”->“属性管理器”；在“Debug|x64”上右击并选择“属性”；



在“通用属性”->“VC++目录”的“包含目录”中添加如下路径：

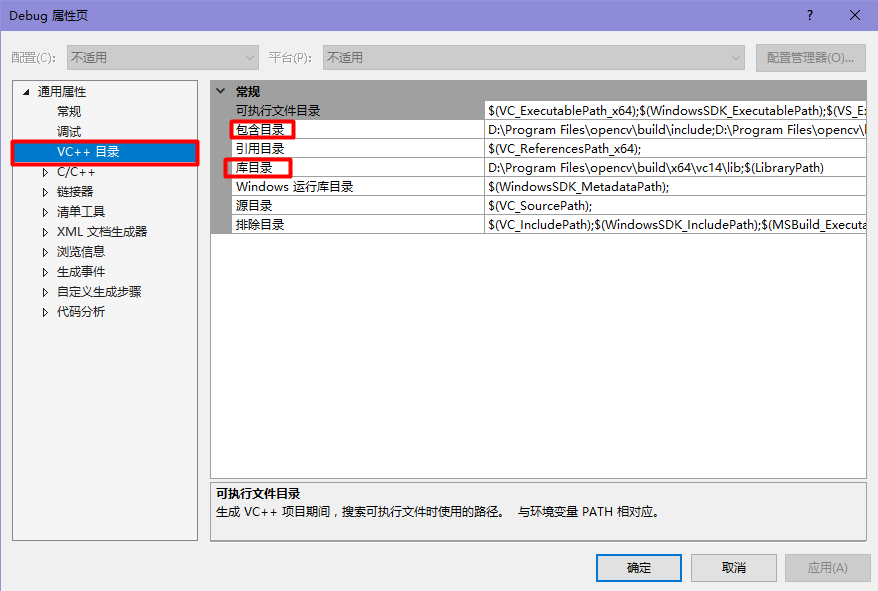
D:\Program Files\opencv\build\include

D:\Program Files\opencv\build\include\opencv

D:\Program Files\opencv\build\include\opencv2

在“库目录”中添加如下路径：

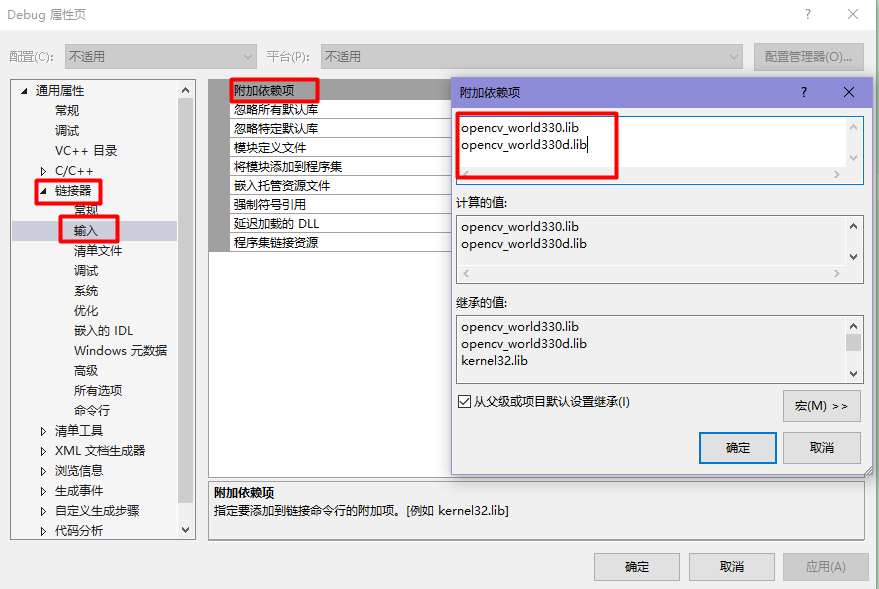
D:\Program Files\opencv\build\x64\vc14\lib



在“链接器”->“输入”的“附加依赖项”中填写如下内容：

opencv\_world330.lib

opencv\_world330d.lib



1. 工具栏的“配置管理器”选择“Debug”和“x64”。
2. 隐藏控制台黑窗口，在“属性管理器”->“Debug|x64”->“属性”中的“Debug属性页”，参照“VS2017+OpenGL环境的配置：（2）隐藏控制台黑窗口：方法二”进行配置。

测试代码：open.cpp //显示图片

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

using namespace cv;

int main()

{

IplImage\* img = cvLoadImage("E:\\image\\AIR.jpg");

cvNamedWindow("显示图片", CV\_WINDOW\_AUTOSIZE);

cvShowImage("显示图片", img);

cvWaitKey(0);

cvReleaseImage(&img);

cvDestroyAllWindows();

}

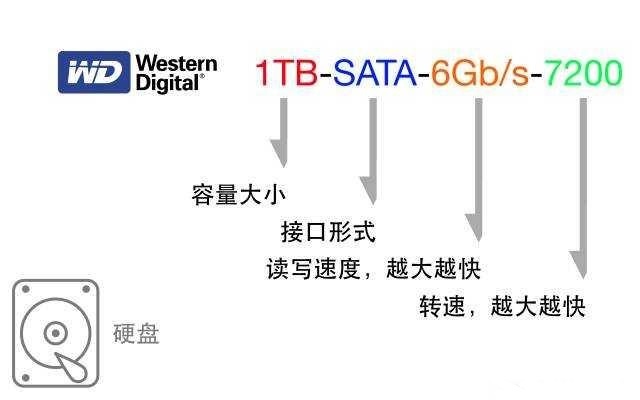
1. **计算机硬件配置解读**

* CPU

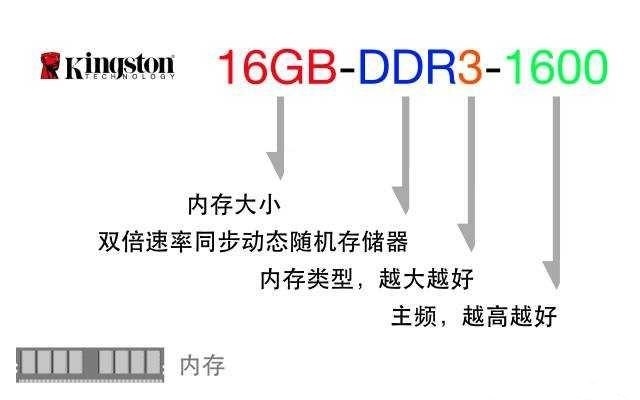




* 硬盘



* 内存



* 显卡





1. **修改CMD默认代码页为UTF-8（重新启动仍有效）**

“Windows+R”=>运行“regedit”=>打开“注册表编辑器”，

进入“计算机\HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Command Processor”路径（全局设置），

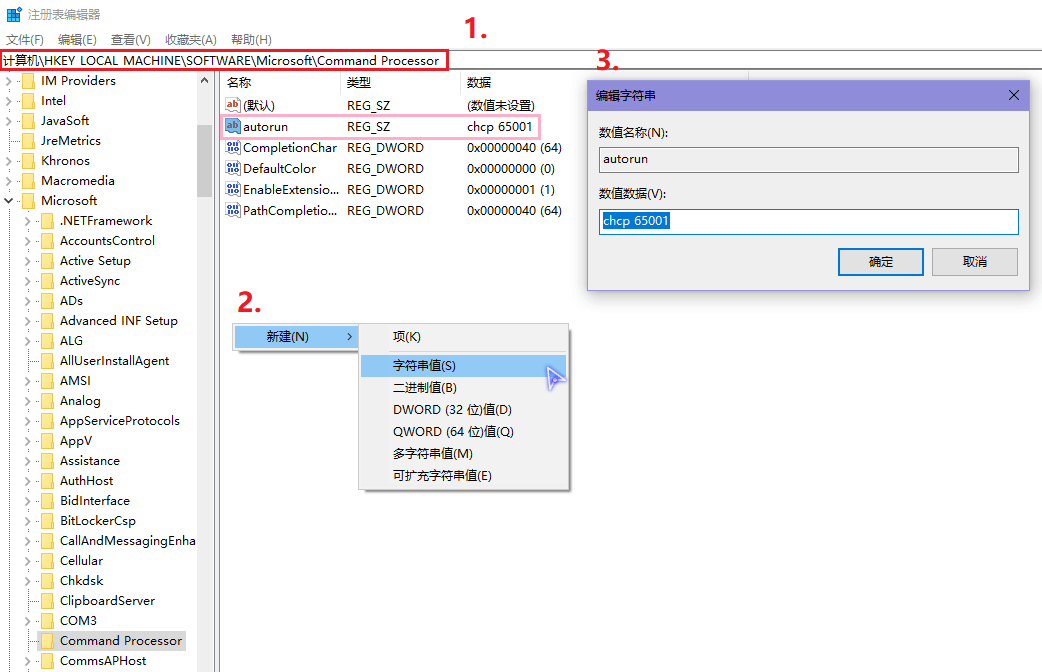
右键“新建|字符串值”，输入“autorun”；

选中autorun，右键“修改…”，打开“编辑字符串”对话框，“数值数据”栏中输入“chcp 65001”。

常用活动代码页 Active code page：

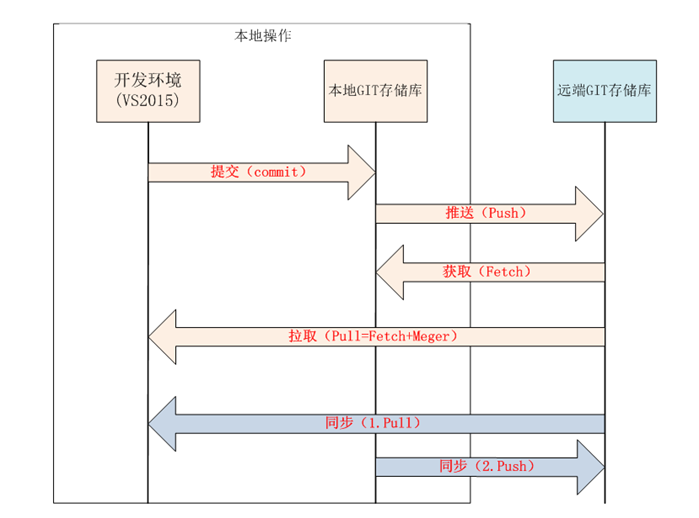
chcp 437 /\*英文\*/， chcp 932 /\*日文\*/， chcp 936 /\*简体中文\*/，

chcp 949 /\*韩文\*/， chcp 950 /\*繁体中文\*/， chcp 65001 /\*UTF-8\*/



1. **GitHub托管配置**

工作流程：



“先拉后推”：拉取→提交→推送→同步。

可以上传任何文件（文档、图片、音乐、视频等均可）。

**拉取（Pull）：**将远程版本库合并到本地版本库，=Fetch+Meger；

**获取（Fetch）**：从远程版本库获得最新版本；

**合并（Meger）：**将两个版本库进行合并操作；

**提交（Commit）：**将所做的更改，存入本地暂存库；

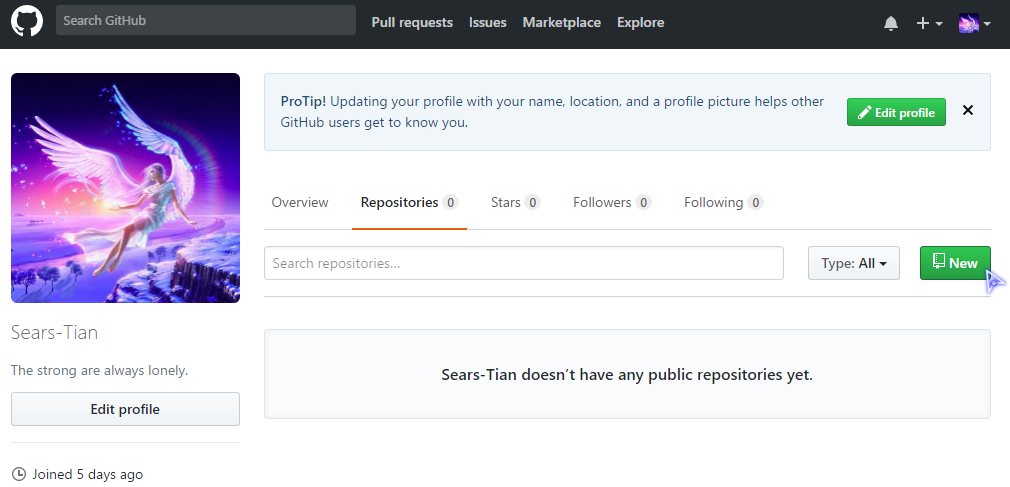
**推送（Push）**：将所做的更改，存入远程版本库；

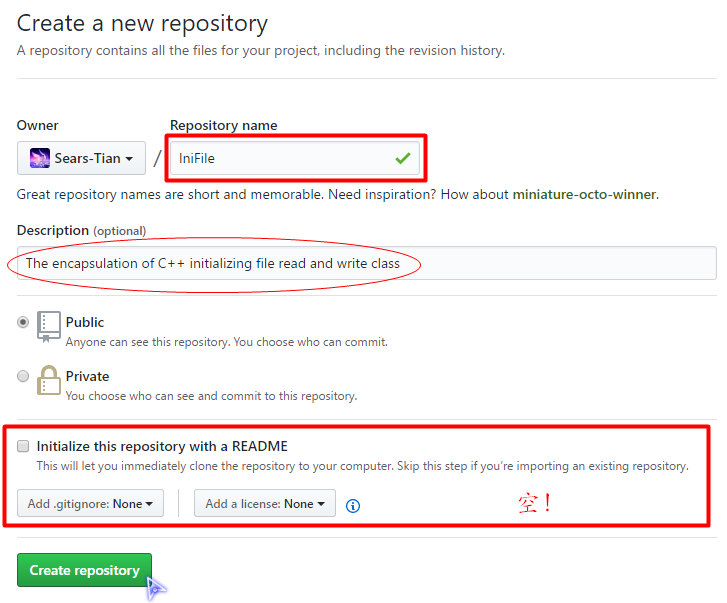
**同步（Syschro）：**先将远程库拉取，再将本地库推送到远程库，=Pull+Push；

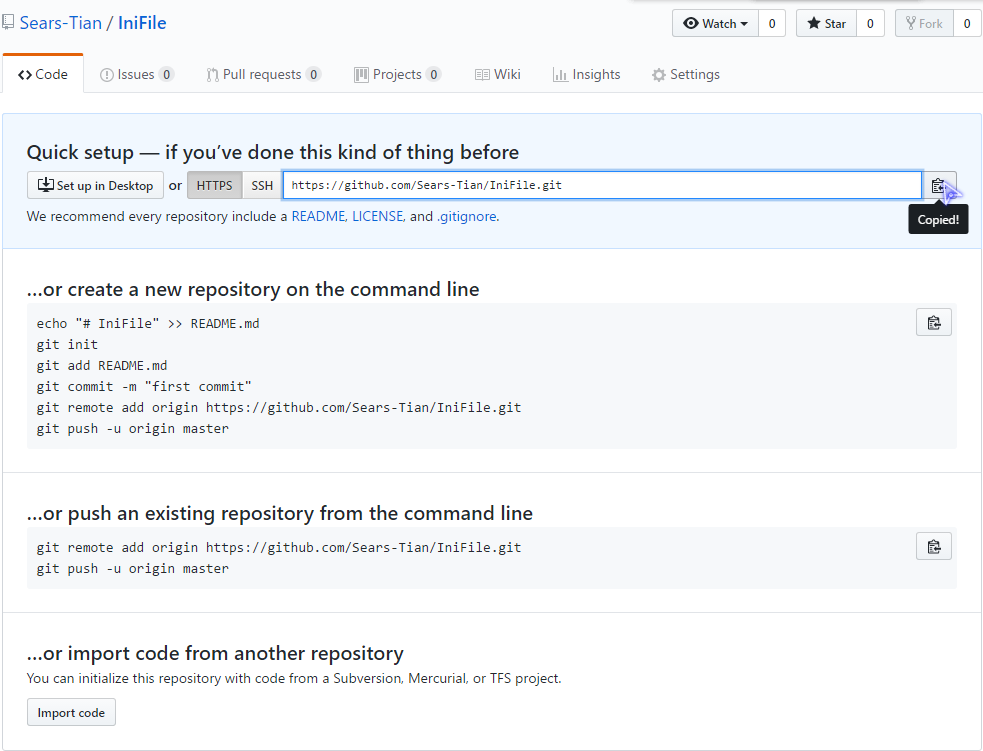
**变基到（Switch）：**切换分支（双击即可切换）。

创建空的仓库！否则会报错，可能需要更新VS版本：









**(1) Visual Studio 2017**

* 本地已有项目

①打开项目，“添加到源代码管理”，



②利用“更改”快捷按钮，更改Git用户的“名称”和“电子邮件地址”作为全局设置，



③利用“同步”快捷按钮，填写“推送到远程存储库”的URL并“发布”，



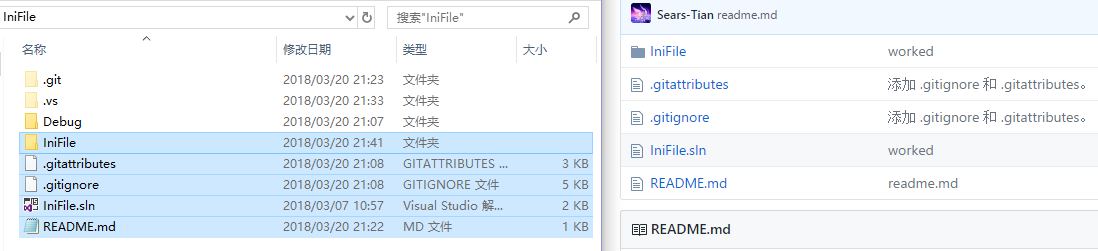
④开始与云端“同步”。

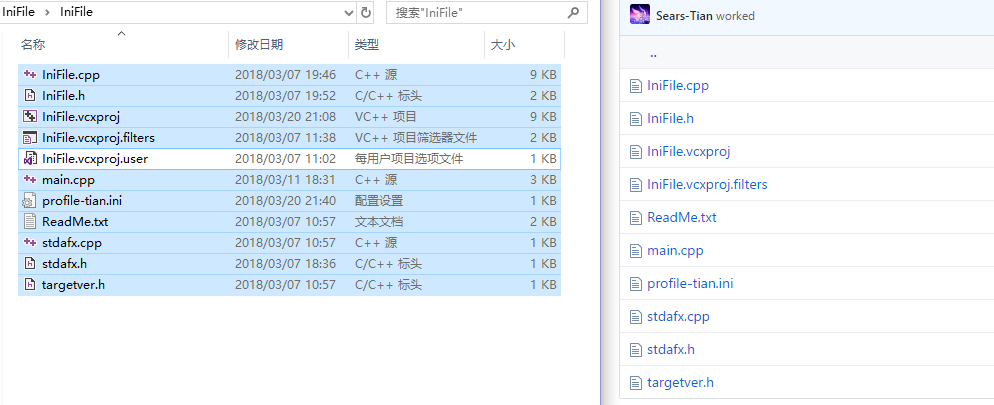


主页按钮的功能：



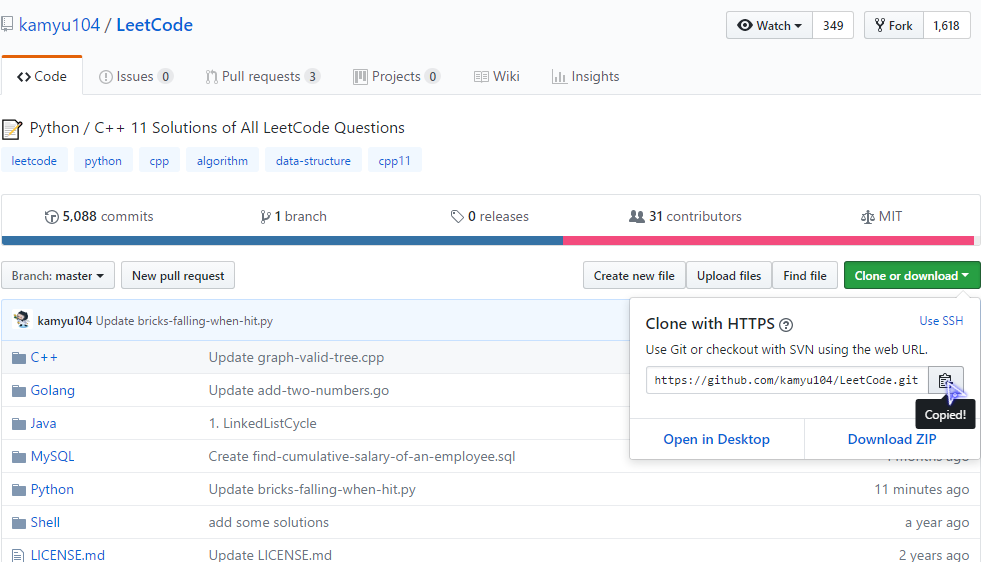
本地文件与云端文件的对比：



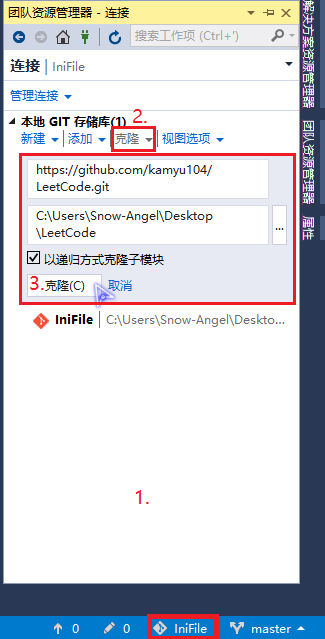


* 克隆网上项目

①复制仓库链接，

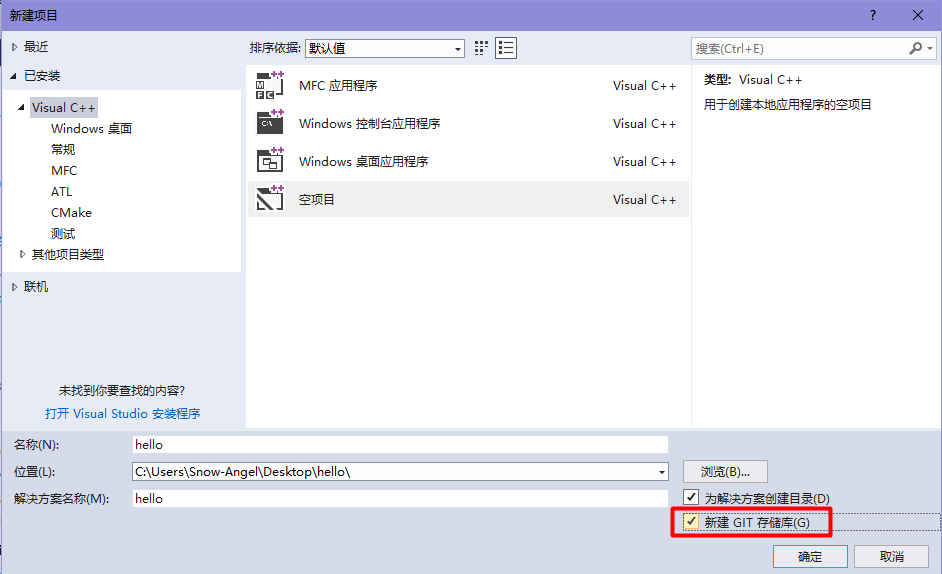


②在“连接|本地Git存储库”中分别填入 云端URL 和 本地路径 并“克隆”。

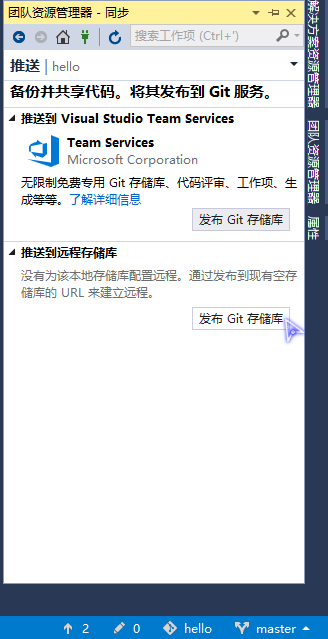


* 创建新的项目

①在新建项目的同时可以勾选“新建Git存储库”，



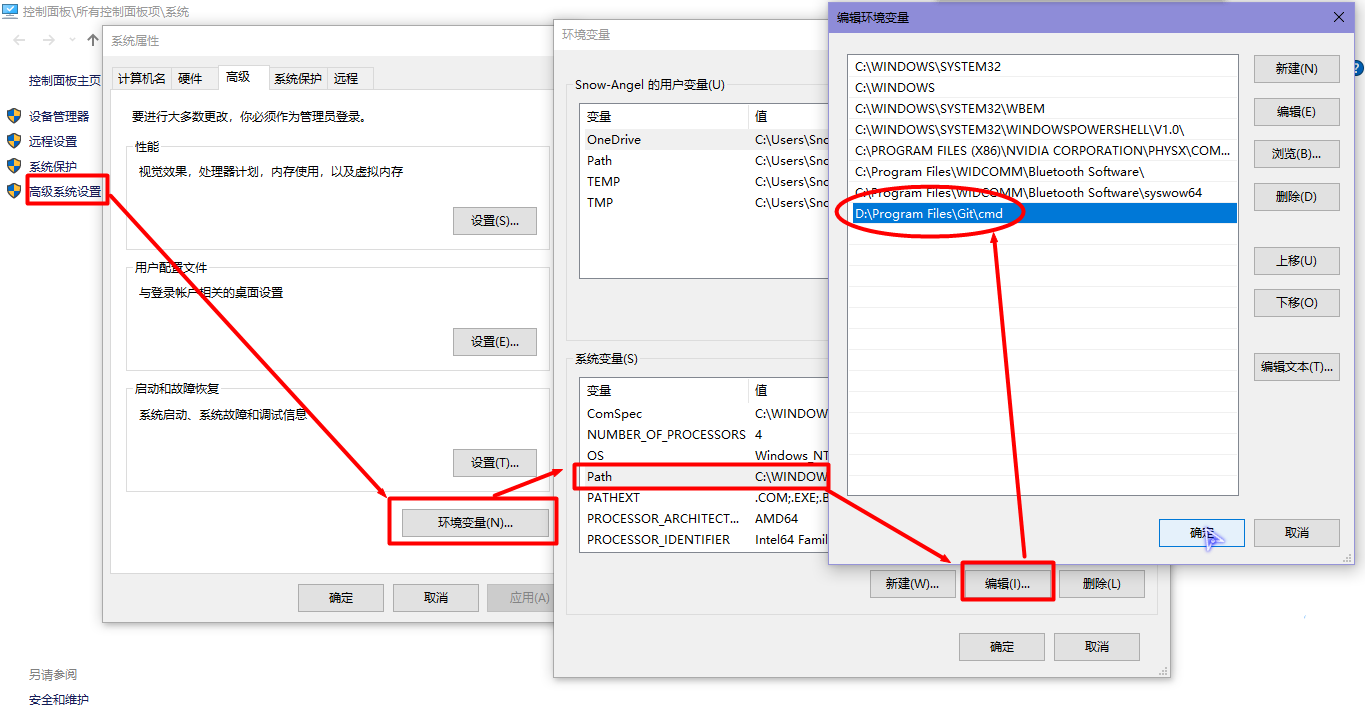
②之后设置 账户 及 发布URL 即可。



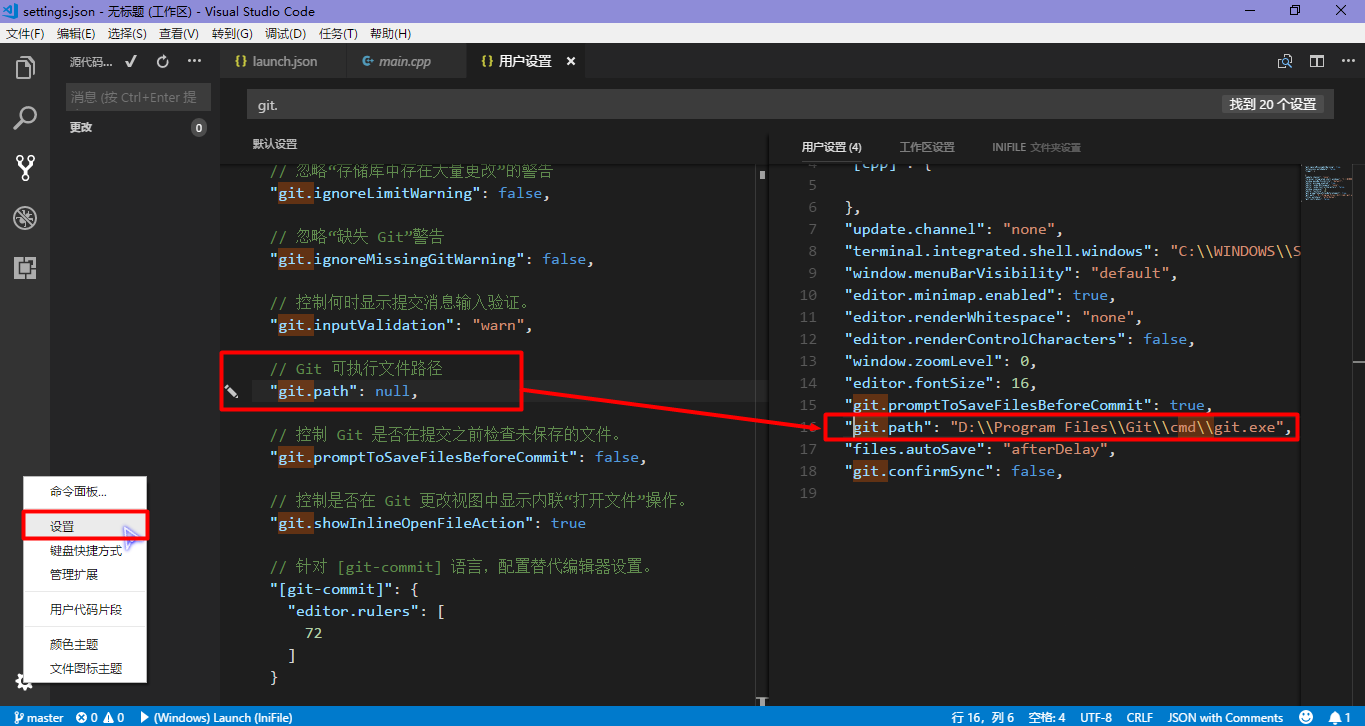
**(2) Visual Studio Code**

①安装Git for Windows应用程序，配置环境变量（默认已配置好）：

系统变量Path新建路径：D:\Program Files\Git\cmd



* 1. 更改VS Code设置："git.path":"D:/Program Files/Git/cmd/git.exe"



③配置Git：设置全局配置（*push an exting reposity*）

git config --global user.name "*Sears-Tian*"

git config --global user.email "*startjh@163.com*"

//首先进入到项目目录下

**cd C:\Users\Snow-Angel\Desktop\IniFile**

git init

//创建README.md文件 或 空文件

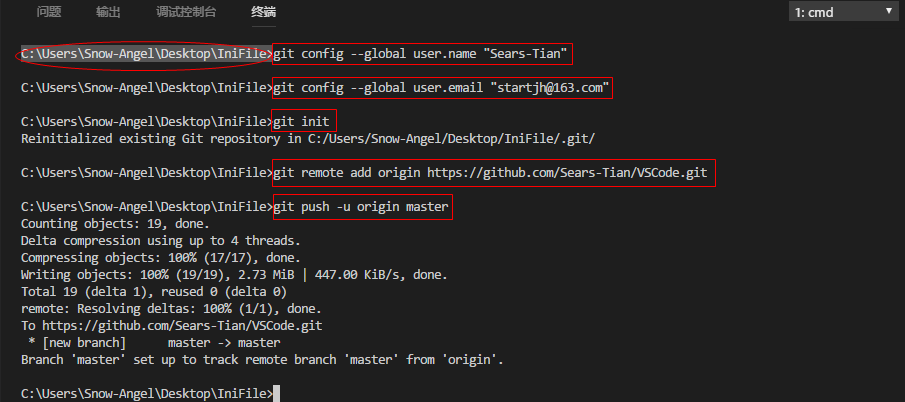
**echo "# code" >> README.md 或 touch README.md**

**git add README.md**

**git commit -m "first commit"**

git remote add origin *https://github.com/Sears-Tian/VSCode.git*

git push -u origin master





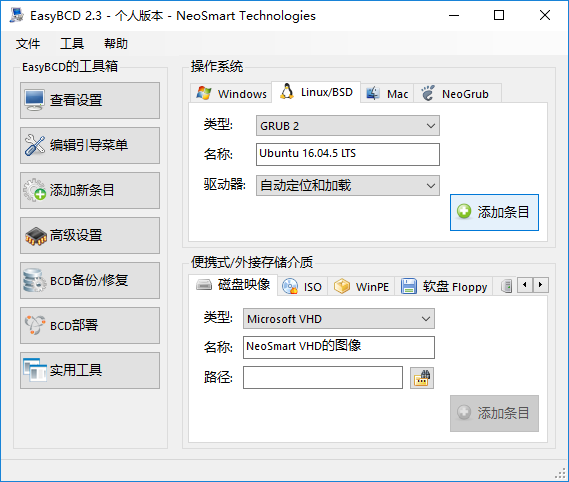
可视化Git仿真沙盒学习网站：<https://learngitbranching.js.org/>

显示全部命令：show commands。（显示答案：show solution）

1. **安装双系统：Windows 10 + Ubuntu 16.04.5 LTS**
2. 预留50GB磁盘空间用于存放Linux系统数据。
3. “**U启动**”制作U盘启动盘，安装Windows系统。
4. “**UltraISO软碟通**”将Linux系统“写入硬盘映像”制作U盘启动盘，以安装Ubuntu；
5. 在新建磁盘基础上创建Linux系统分区：先分逻辑分区，再分主分区，其余给“/home”

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 分区次序 | **大小(GB)** | **新分区的类型** | **新分区的位置** | **用于** | **挂载点** | 说明 |
| 1 | 2 | 逻辑分区 | 空间起始位置 | 交换空间 |  | 虚拟内存 |
| 2 | 1 | 逻辑分区 | Ext4日志文件系统 | /boot | 引导分区 |
| 3 | 10 | 主分区 | / | 系统主区 |
| 4 | 剩余37 | 逻辑分区 | /home | 用户数据 |

1. 开机只能进入Windows，解决方法：用“**easyBCD**”添加Linux启动项。



P.S. 你的名字：Sky，计算机名：SKY，用户名：sky，登录密码：5201314

1. **VS2017自定义导出项目模板**

（1）新建项目（如：空项目 TemplateProject）后，修改配置管理器：

Debug/Release | x86/x64:

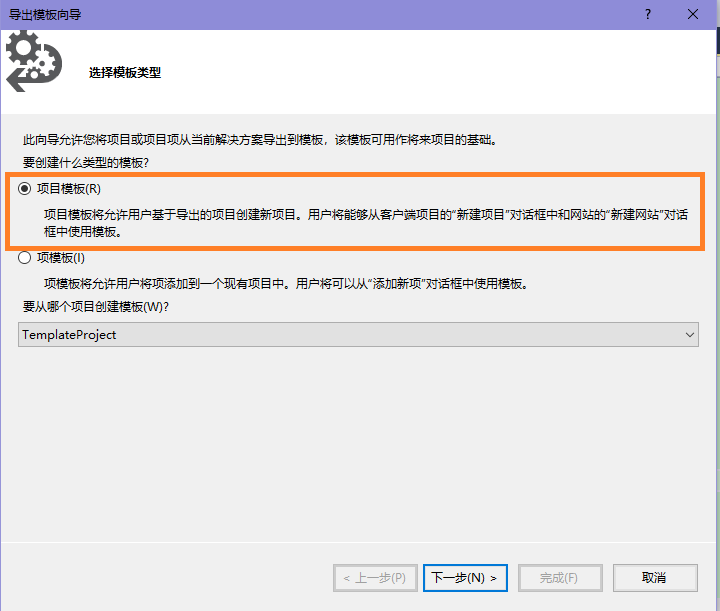
项目属性：使用Unicode字符集+多线程/MT运行库；高级保存选项：Unicode-代码页1200。

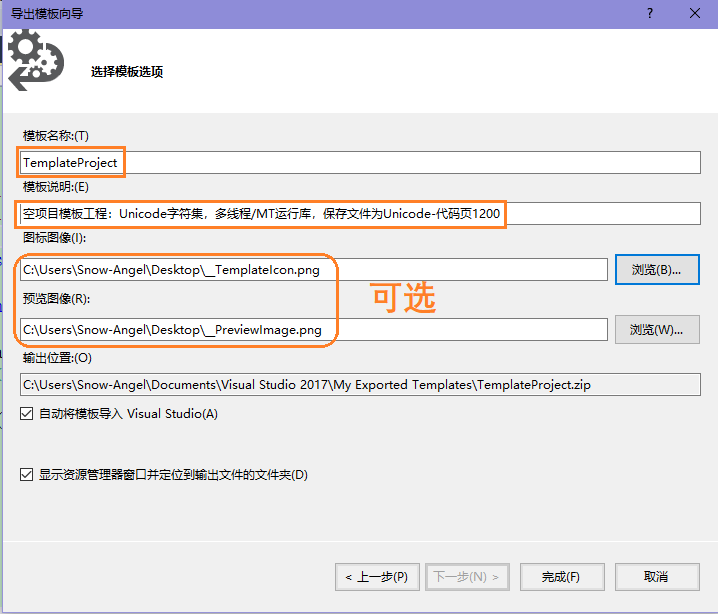
（2）“源文件”=>“添加”=>“新建项”：main.cpp

时间：$time$，账户名：$username$

|  |
| --- |
| /\*  $time$  \*/  #include <iostream>  #include <locale>  using namespace std;  int main(int argc, char\* argv[])  {  setlocale(LC\_ALL, "chs");  wcout << "This is a Template Project." << endl;  system("pause");  return 0;  } |

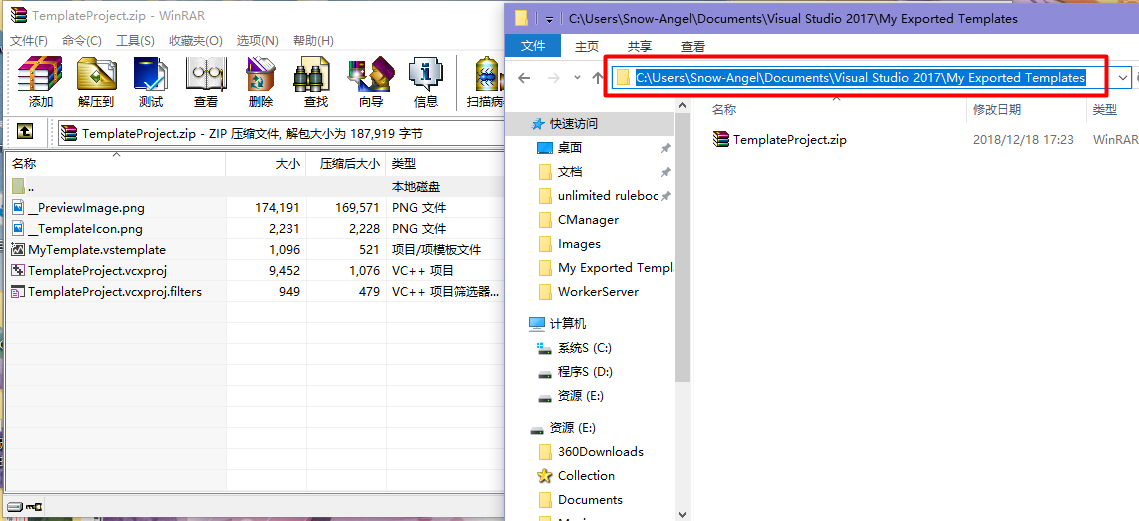
（3）菜单栏：“项目”=>“导出模板…”，打开“导出模板向导”对话框。

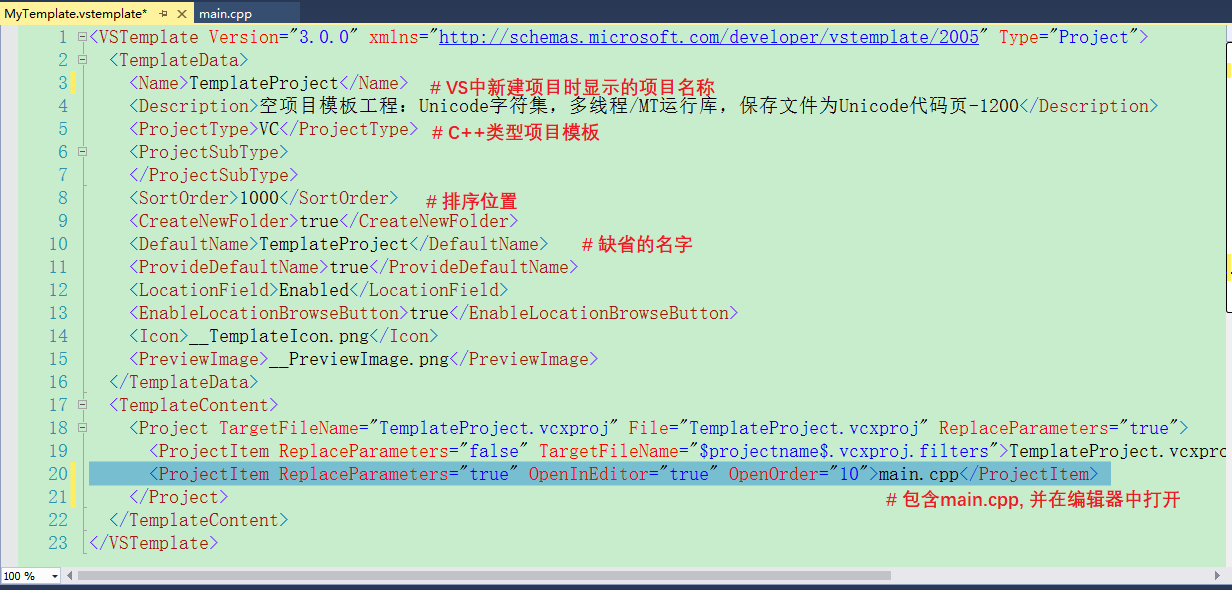




（4）找到导出的压缩文件，打开后添加main.cpp文件，并修改MyTemplate.vstemplate属性：

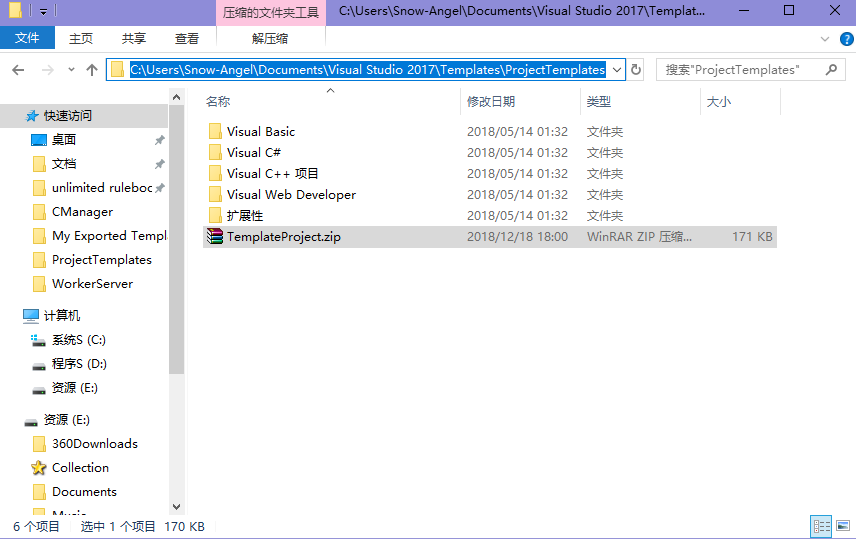
<ProjectItem ReplaceParameters="true" OpenInEditor="true" OpenOrder="10">main.cpp</ProjectItem>



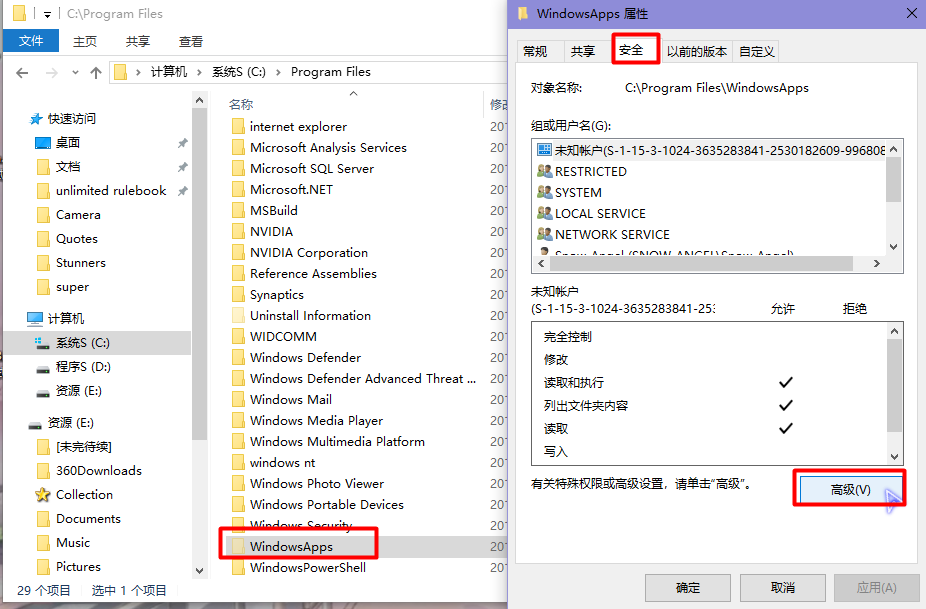


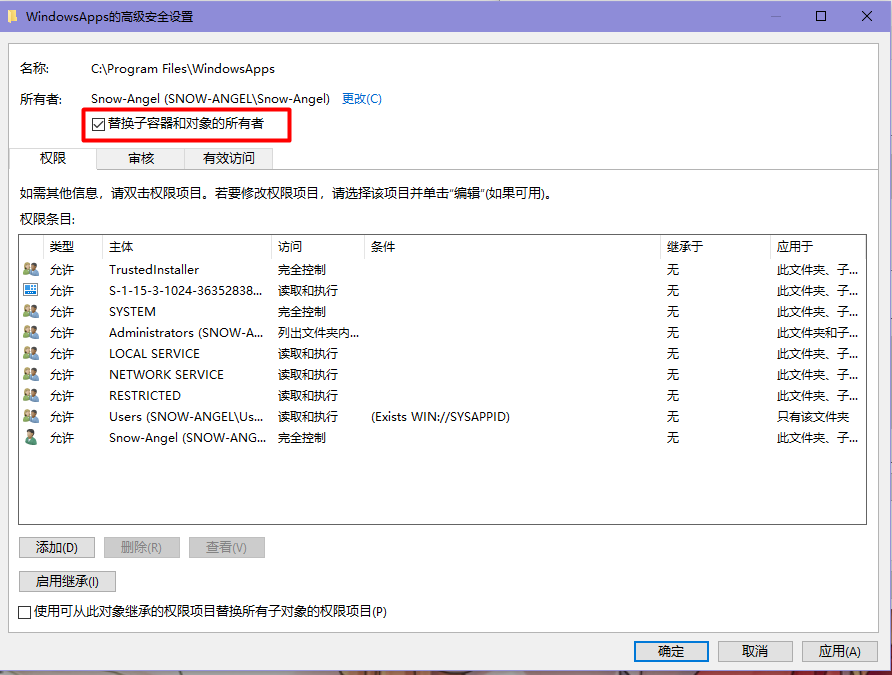
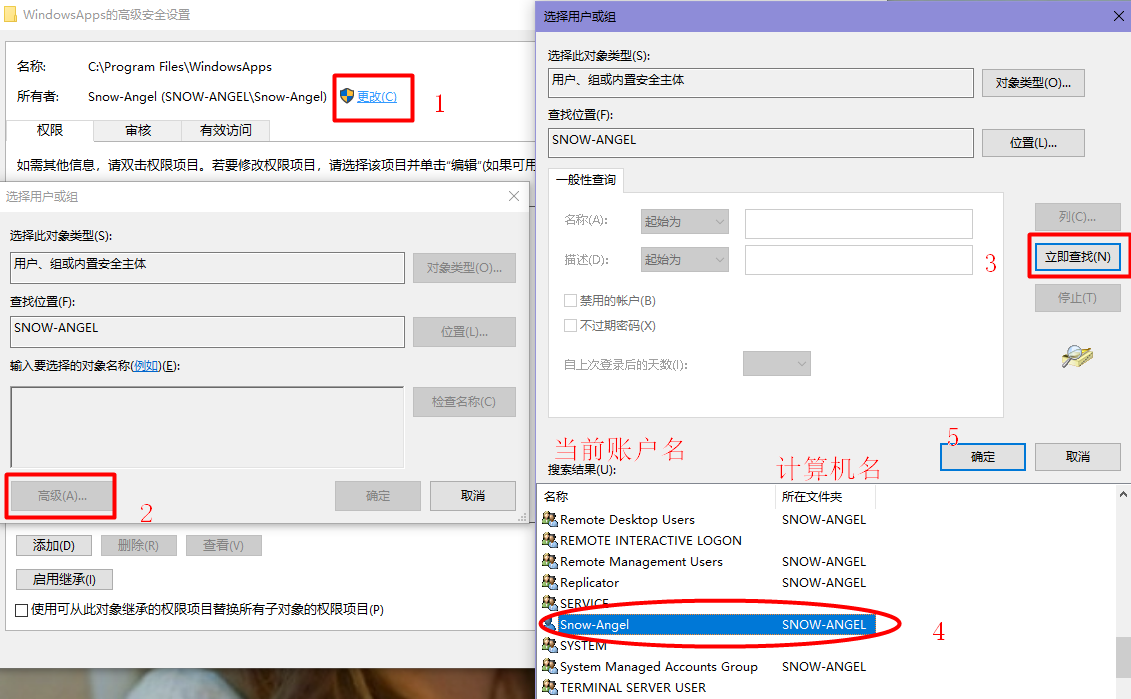
（5）重新压缩后，放在如下目录中，若删除则可从新建项目列表中去除：

C:\Users\Snow-Angel\Documents\Visual Studio 2017\Templates\ProjectTemplates



1. **解决Windows10拒绝你访问文件夹**
2. 在文件夹上 右键：属性 | 安全 | 高级，打开“高级安全设置”对话框；
3. 点击“更改”，在“选择用户或组”对话框中点击“高级…” | “立即查找”，选择当前用户名和计算机名（如：Snow-Angel, SNOW-ANGEL）“确定”后回到“高级安全设置”；
4. 勾选上“替换子容器和对象的所有者”后“确定”即可。

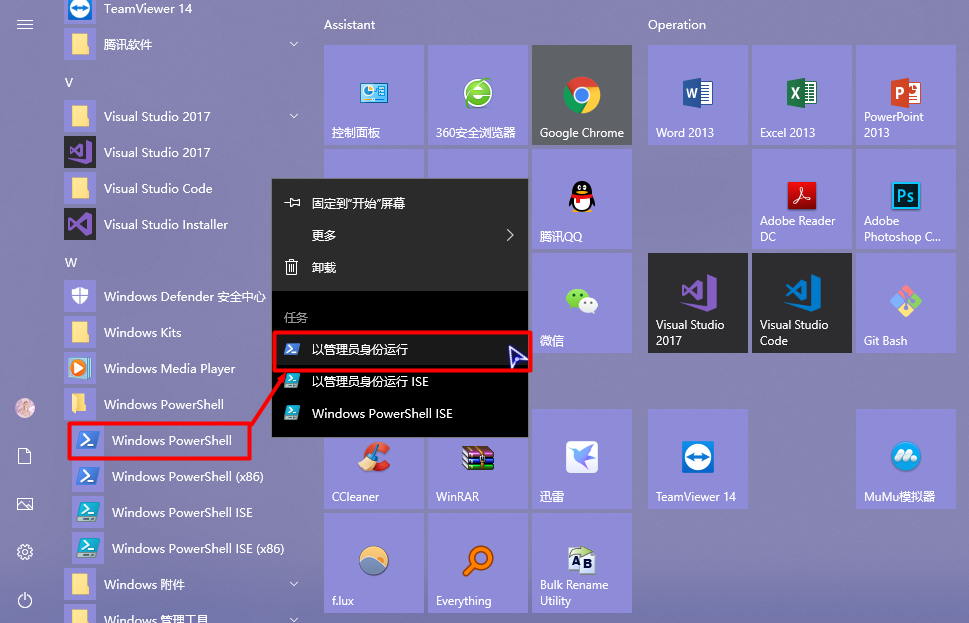


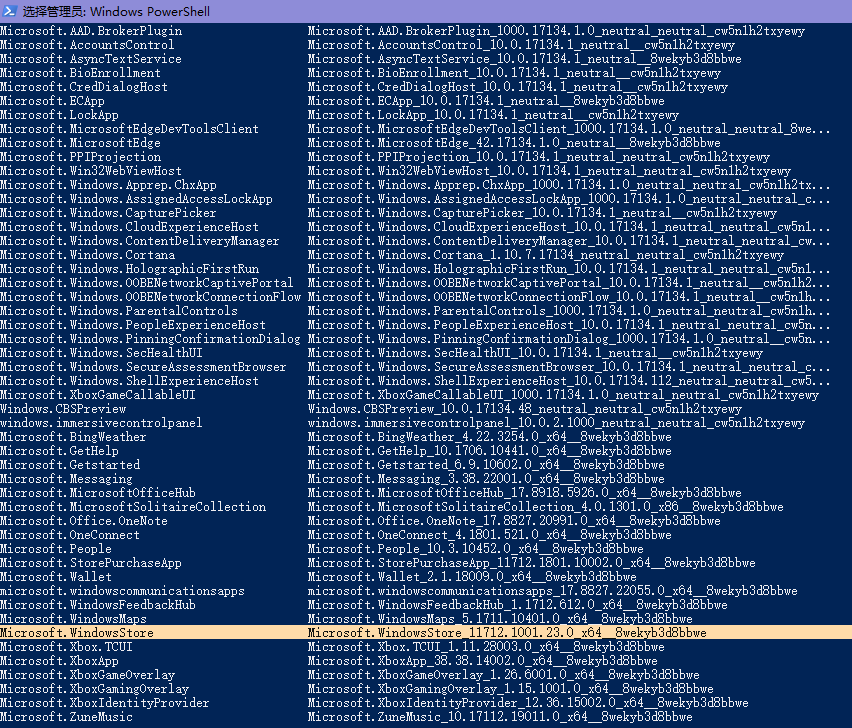


1. **Windows10恢复应用商店Microsoft Store**
2. 以管理员身份运行Windows PowerShell；
3. 输入：**Get-AppxPackage -allusers | Select Name, PackageFullName**
4. 找到“Microsoft.WindowsStore”（应用商店）应用，复制右侧对应的包名称

（如：Microsoft.WindowsStore\_11712.1001.23.0\_x64\_\_8wekyb3d8bbwe）；

1. 输入：**Add-Appxpackage -register "C:\Program Files\WindowsApps\Microsoft.WindowsStore\_11712.1001.23.0\_x64\_\_8wekyb3d8bbwe\appxmanifest.xml" -disabledevelopmentmode**
2. 回车后开始部署。





1. **MySQL数据库**

(1) 设置：**user = root, password = 5200**。运行”cmd”，输入：mysql -u root -p 进入。

(2) 解决：Navicat Premium 12连接MySQL 8.0.15.0 出现错误：Authentication plugin 'caching\_sha2\_password' cannot be loaded：

#登陆MySQL

mysql -u root -p

#使密码从不失效

ALTER USER 'root'@'localhost' IDENTIFIED BY '5200' PASSWORD EXPIRE NEVER;

#修改加密规则

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY '5200';

#刷新权限

FLUSH PRIVILEGES;

#重置下密码

alter user 'root'@'localhost' identified by '5200';

(3) 远程连接访问：（在同一个局域网/网段内）

e.g. MySQL服务主机：IP地址 192.168.1.140，访问主机：IP地址 192.168.1.139

① 在服务主机上：

#创建用户

CREATE USER 'new'@'%' IDENTIFIED BY '1234';

#授予权限

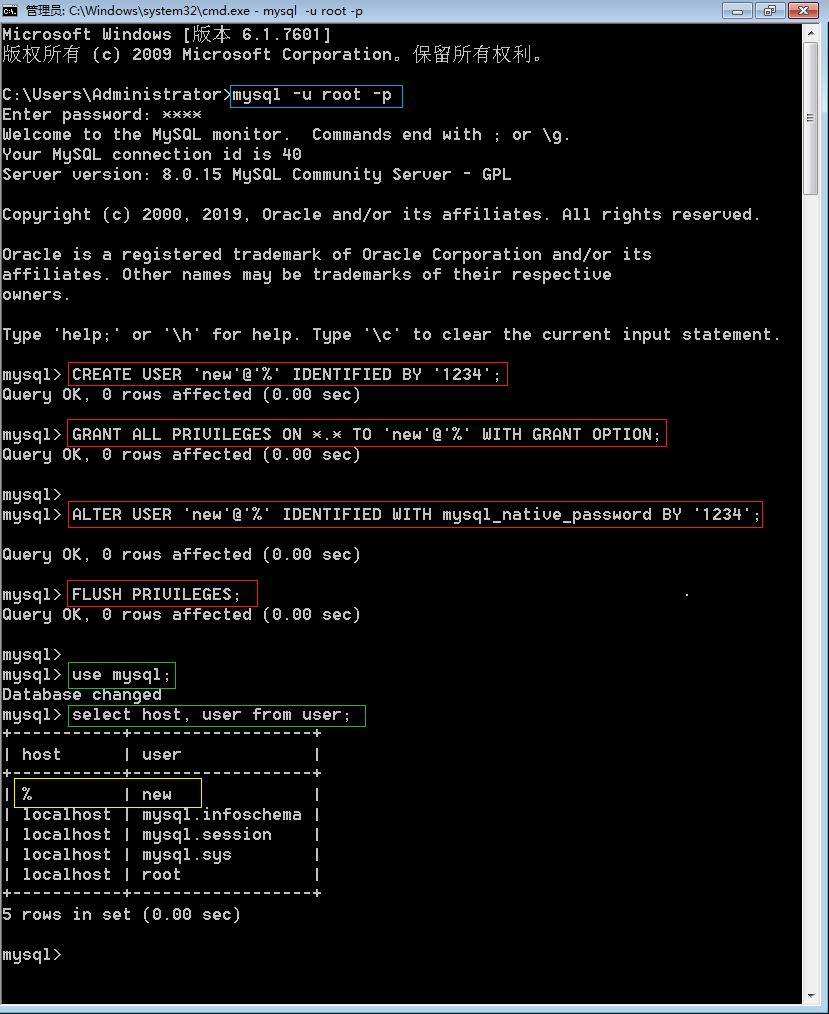
GRANT ALL PRIVILEGES ON \*.\* TO 'new'@'%' WITH GRANT OPTION;

#修改加密规则

ALTER USER 'new'@'%' IDENTIFIED WITH mysql\_native\_password BY '1234';

# 刷新权限

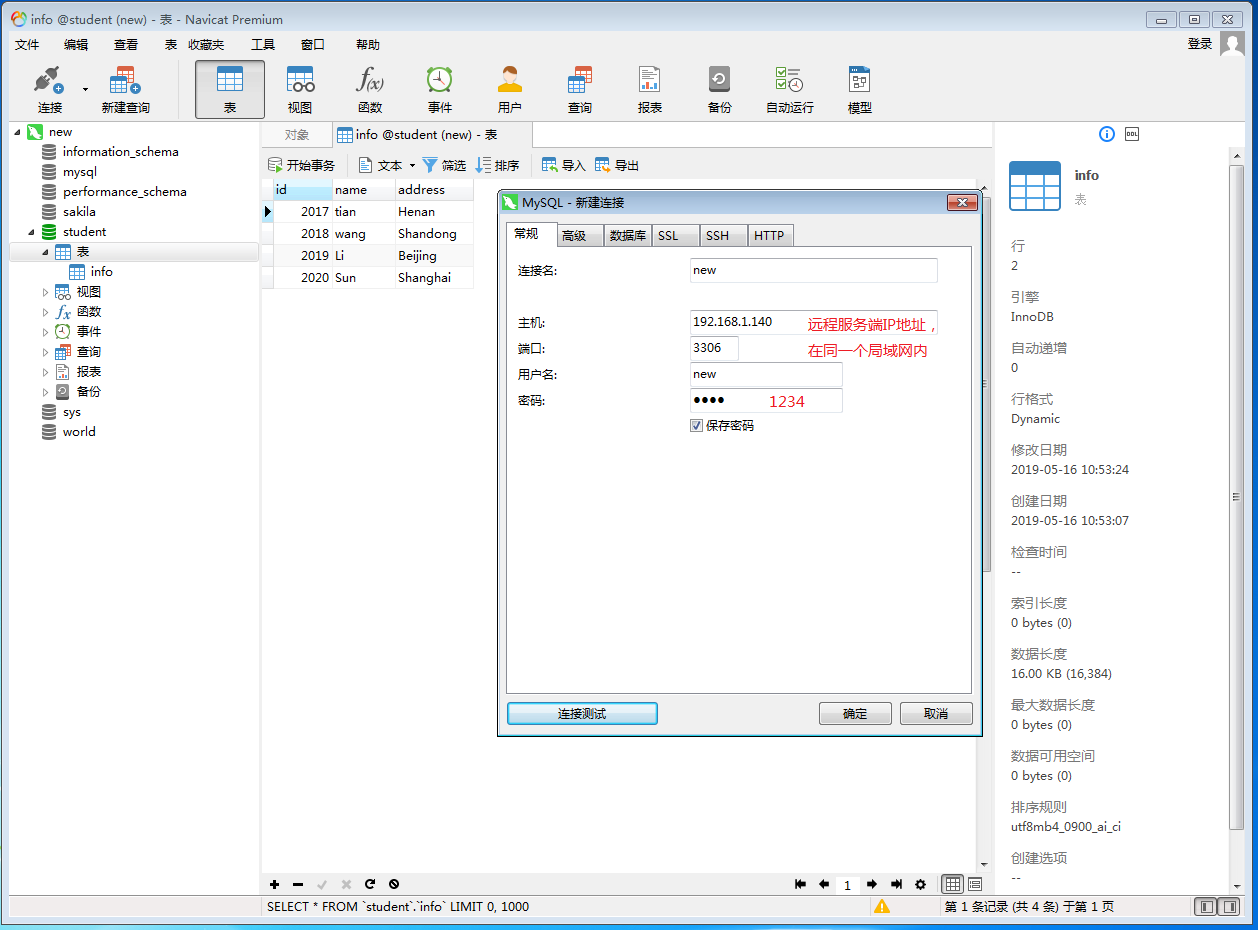
FLUSH PRIVILEGES;



② 在访问主机上：（通过Navicat Premium 12连接）

新建MySQL连接：

连接名：new（自定义），主机：192.168.1.140，端口：3306，用户名：new，密码：1234



1. **VS2017编译SQLite3生成.lib**

sqlite3.dll动态链接库，允许程序共享执行特殊任务所必需的代码和其他资源。

动态的sqlite3.lib相当于一个.h文件，是对实现部分（.dll文件）的导出部分的声明。编译后只是将导出声明部分编译到宿主程序中，其生成的sqlite3.exe在运行时需要相应的.dll文件支持。

静态的sqlite3.lib将导出声明和实现都放在.lib中，编译后所有代码都嵌入到宿主程序，其生成的sqlite3.exe可直接运行。

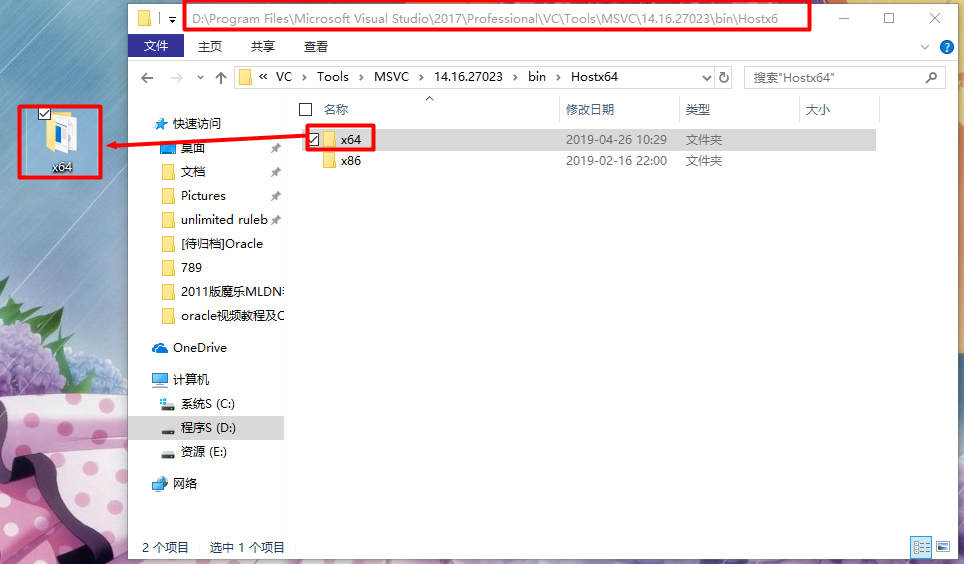
以编译生成x64位的为例。下载文件：<https://www.sqlite.org/download.html>

sqlite-amalgamation-3280000.zip (包含文件：shell.c, sqlite3.c, sqlite3.h, sqlite3ext.h),

sqlite-dll-win64-x64-3280000.zip (包含文件：sqlite3.def, sqlite3.dll)

需要使用VS2017的lib.exe，在D:\Program Files\Microsoft Visual Studio\2017\Professional\VC\Tools\MSVC\14.16.27023\bin\Hostx64\x64\目录下，

复制整个文件夹到新的目录（如：桌面），

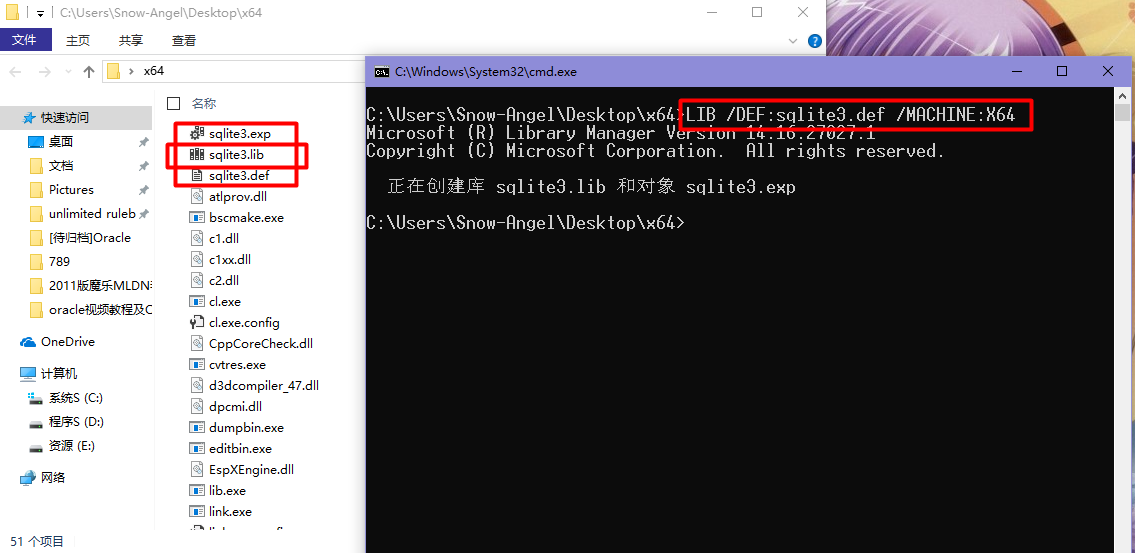


**方案一：编译生成动态sqlite3.lib**

sqlite3.def = lib.exe => dynamic sqlite3.lib = solution => SQLite3-dynamic\_lib.exe + sqlite3.dll

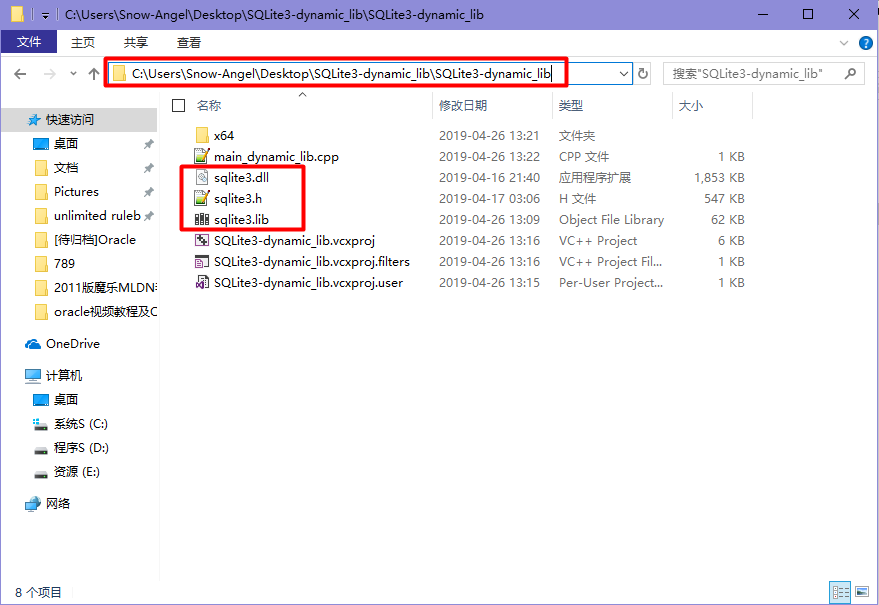
1. 将sqlite3.def复制到新的目录下（如：C:\Users\Snow-Angel\Desktop\x64），

进入cmd输入：LIB /DEF:sqlite3.def /MACHINE:X64，生成需要的动态sqlite3.lib文件。



2. 新建空项目测试：SQLite3-dynamic\_lib，

添加sqlite3.h, sqlite3.dll和生成的sqlite3.lib到项目目录下

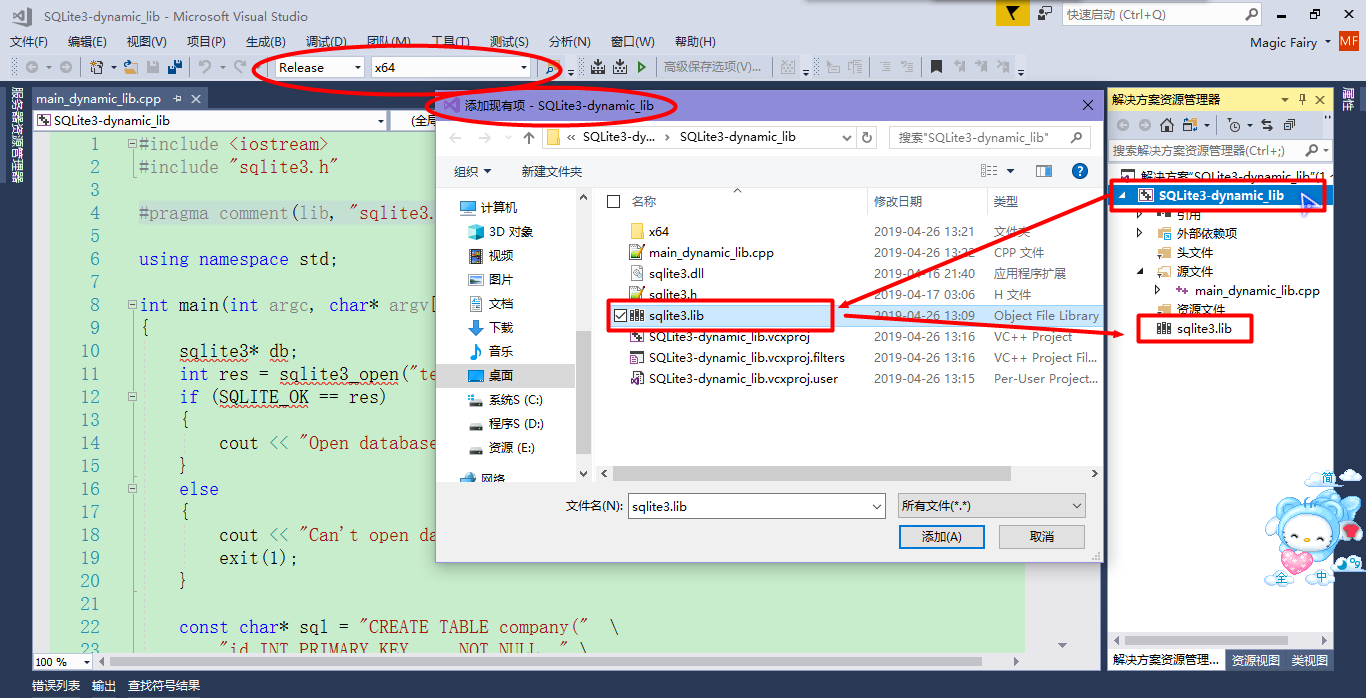


3. 包含头文件：#include “sqlite3.h”

预编译lib文件：

方式1. #pragma comment(lib, "sqlite3.lib")

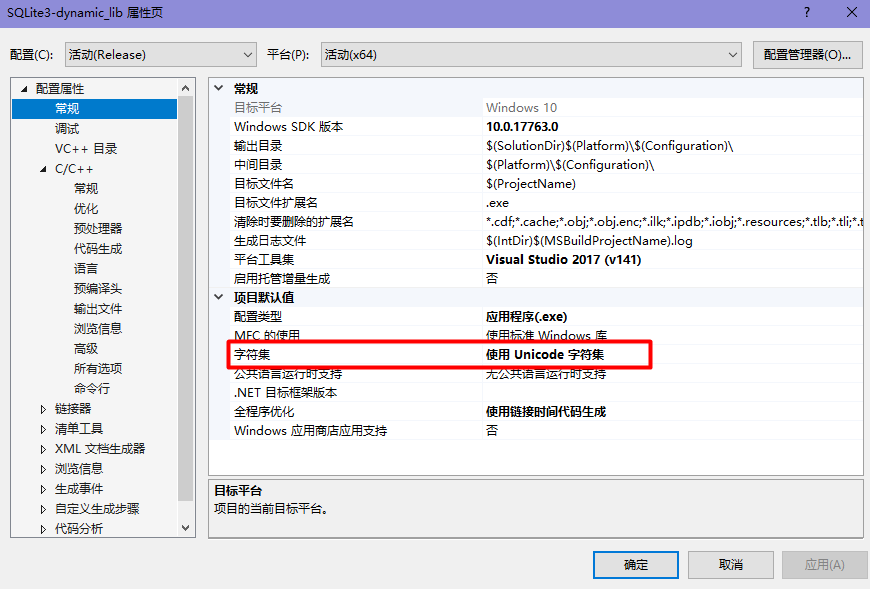
方式2. 在项目上 右键 | 添加 | 现有项…，选择sqlite3.lib，自动添加到资源文件中

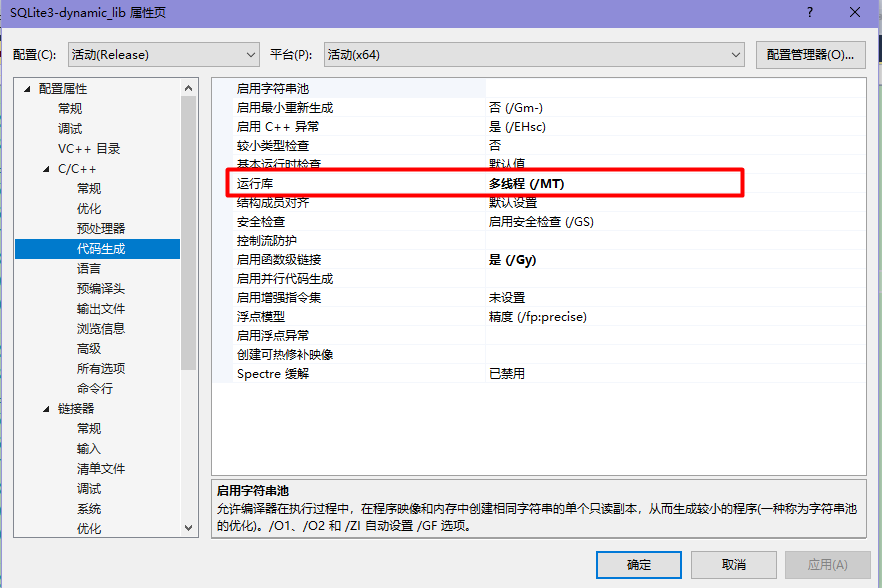


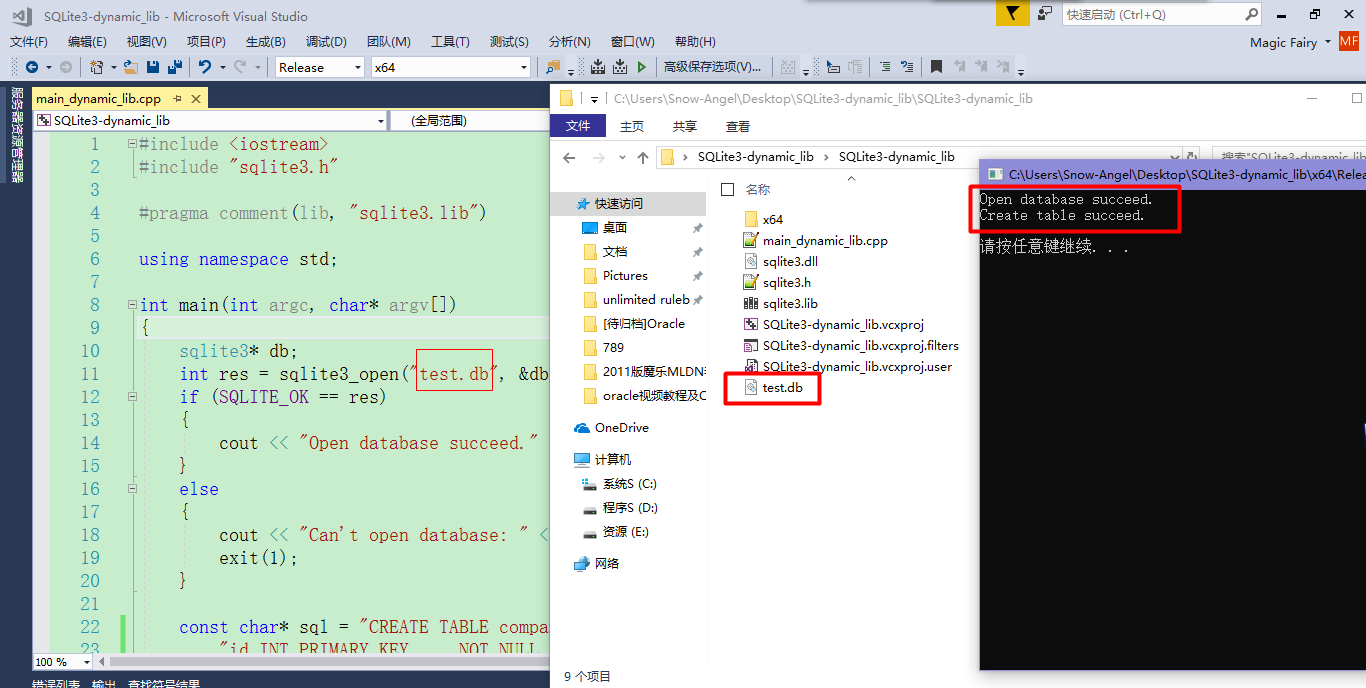
4. 项目属性 | 配置属性 |

常规 | 字符集：使用Unicode字符集，C/C++ | 代码生成 | 运行库：多线程(/MT)

设置为**Release x64**，编译运行。







5. 测试代码：

|  |
| --- |
| #include <iostream>  #include "sqlite3.h"  #pragma comment(lib, "sqlite3.lib")  using namespace std;  int main(int argc, char\* argv[])  {  sqlite3\* db;  int res = sqlite3\_open("test.db", &db);  if (SQLITE\_OK == res)  {  cout << "Open database succeed." << endl;  }  else  {  cout << "Can't open database: " << sqlite3\_errmsg(db) << endl;  exit(1);  }  const char\* sql = "CREATE TABLE company(" \  "id INT PRIMARY KEY NOT NULL, " \  "name TEXT NOT NULL, " \  "age INT NOT NULL, " \  "address CHAR(50), " \  "salary REAL );";  // Execute SQL statement  char\* errmsg = nullptr;  res = sqlite3\_exec(db, sql, 0, 0, &errmsg);  if (SQLITE\_OK == res)  {  cout << "Create table succeed.\n" << endl;  }  else  {  cout << "SQL error: " << errmsg << endl;  sqlite3\_free(errmsg);  }  sqlite3\_close(db);  system("pause");  return 0;  } |

**方案二：编译生成静态sqlite3.lib**

sqlite3.c, sqlite3.h, sqlte3ext.h = solution\_obj / dll=> sqlite3.obj = LIB sqlite3.obj =>

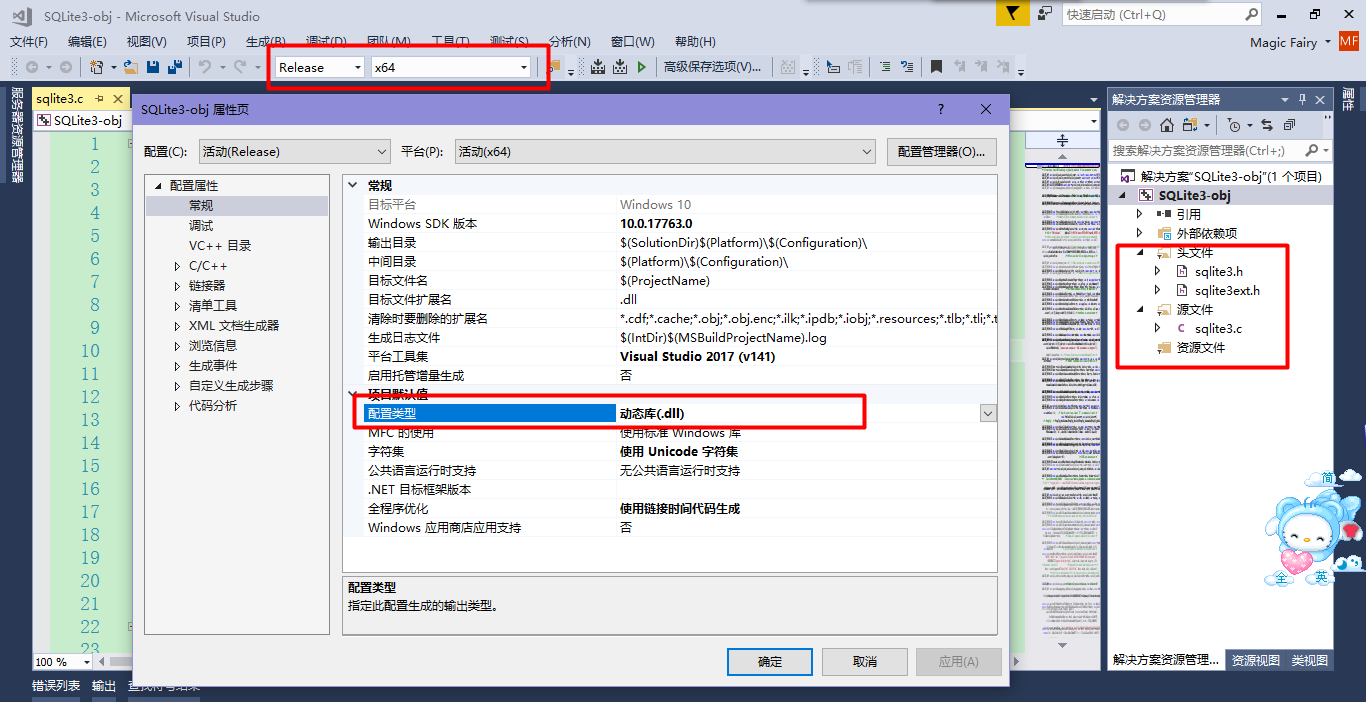
static sqlite3.lib => = solution => SQLite3-static\_lib.exe

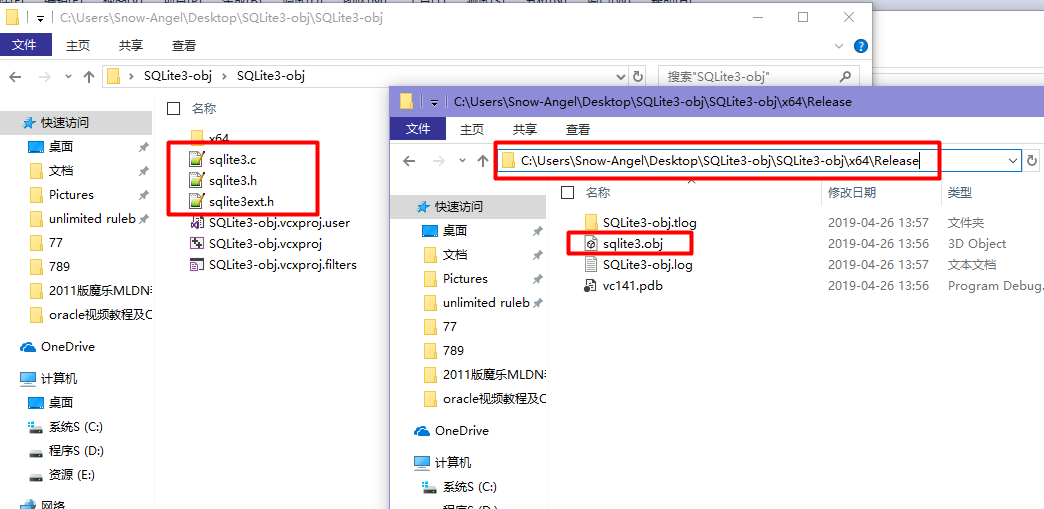
1. 新建空项目SQLite3-obj，添加文件sqlite3.c, sqlite3.h, sqlte3ext.h到项目目录下，

配置类型为动态库(.dll)，生成 **Release x64**，

在路径

C:\Users\Snow-Angel\Desktop\SQLite3-obj\SQLite3-obj\x64\Release下会生成sqlite3.obj文件

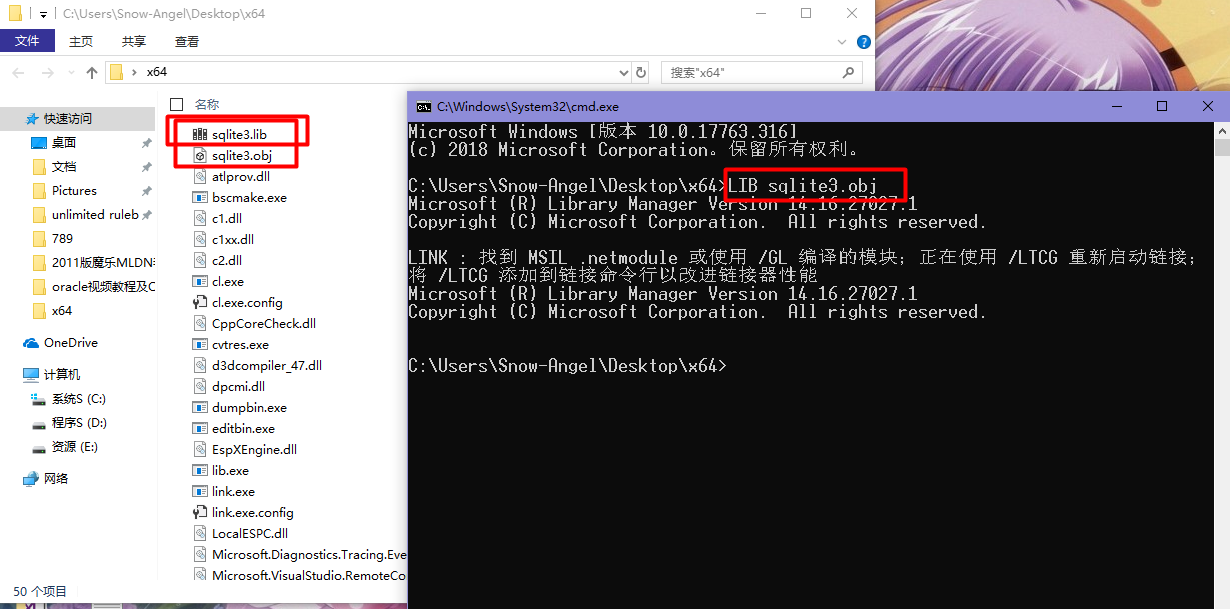




2. 复制sqlite3.obj到复制的目录C:\Users\Snow-Angel\Desktop\x64下，

进入cmd输入：LIB sqlite3.obj，生成静态的sqlite3.lib。

（其实是将.c .h等文件编译后的目标文件.obj用lib.exe重新生成链接库文件.lib）



3. 新建空项目测试：SQLite3-static\_lib

添加sqlite3.h, sqlite3.dll和生成的sqlite3.lib到项目目录下，同上。

