VC++ UDF Studio 2021R1 Tutorial Open Chinese Version

Supported Versions:

| Windows, Fluent, Visual Studio Versions	Support or not
Win XP~Win10 (x86/x64)	✔
Fluent 6.3~2021R1(x86/x64)	✔
Visual Studio 2008 SP1~2013 Professional or Ultimate	✔
Visual Studio 2015 or higher	✖

* Win10 + Visual Studio 2010 Ultimate+ Fluent 17.0 (or higher) are recommended

Trial Version vs. Registered Version:

| Capabilities	Academic Version		Enterprise Version	
	Trial	Registered	Trial	Registered
Compile and debug (serial, single precision)	✔ max. 2 macros	✔ unlimited macros	✔ max. 2 macros	✔ unlimited macros
Compile and debug (serial, double precision)	✖	✔	✖	✔
Compile and debug (parallel, single/double precision)	✖	✔	✖	✔
Call C++/Win32 API/MFC functions	✔	✔	✔	✔
Get boundary id from name	✔	✔	✔	✔
Call 3rd-party library	✔	✔	✔	✔
Set 3rd-party library folders	✖	✖	✔ max. 1 folder	✔ unlimited folders
Add user menu in Fluent	✖	✖	✔ max. 2 submenus	✔ unlimited submenus
Drive Fluent to iterate by UDF	✖	✖	✖	✔
Call Scheme/TUI in UDF	✖	✖	✖	✔
Couple iteration with Matlab	✖	✖	In develop.	In develop.

Important Notes on Visual Studio Installation:

1. Visual C# is recommended to be installed along with Visual C++, though it is not necessary for some Visual Studio version (e.g. VS2008 SP1 ultimate edition). But for VS2010, installation of Visual C# is mandatory, otherwise error will occur when starting Visual Studio.
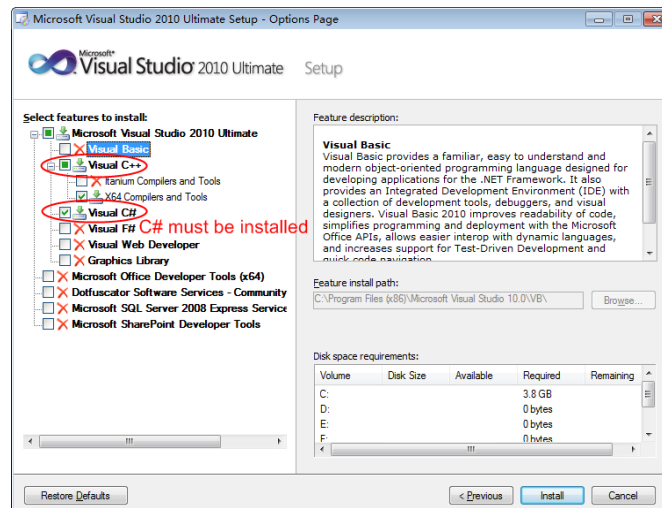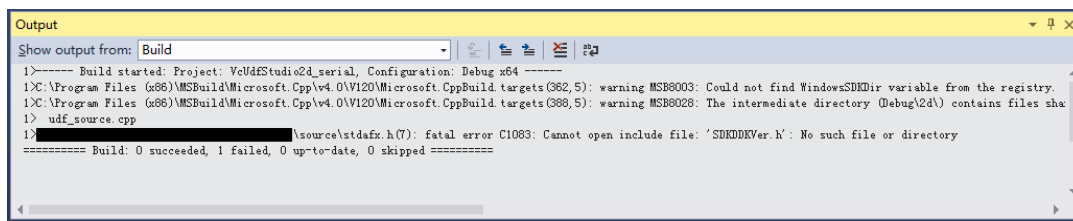
For VS2008, please assure "Visual C++ Tools" to be installed. In addition, for VS2008 standard edition, Visual C# is also necessary to avoid the "can't find cl.exe" error. In addition, VS 2008 Service Pack 1 must be installed based on the original version.

For 64 bit Windows, you have to install 64bit Fluent and please assure that "X64 compilers and Tools" will be installed.
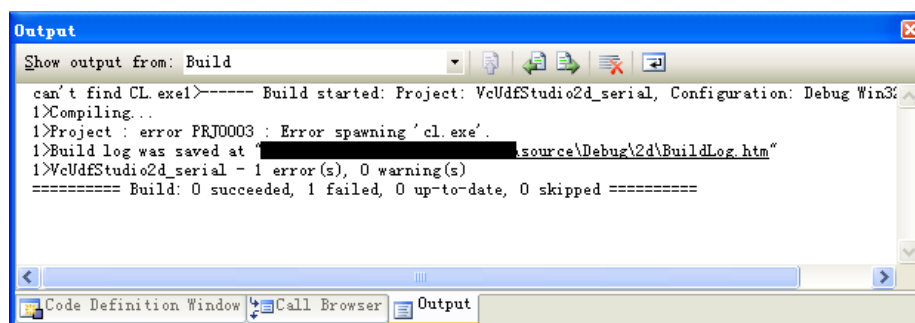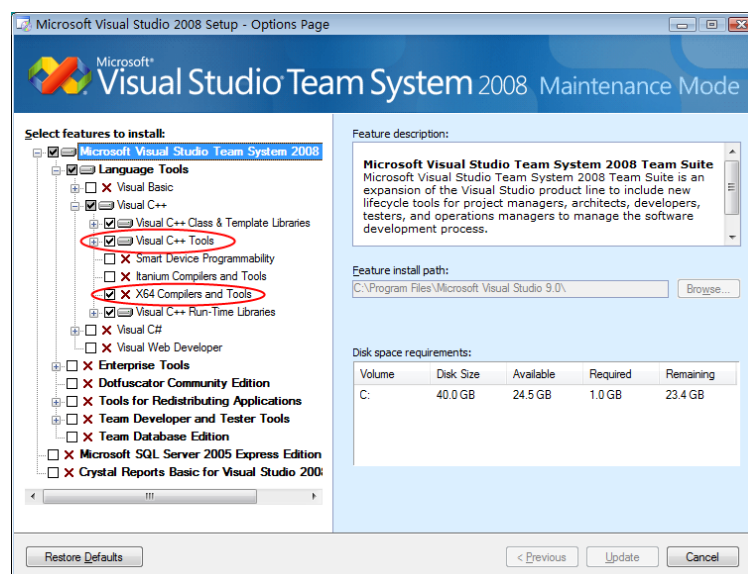
2. If you prefer Visual Studio 2013, please install the latest **Visual Studio 2013 update5.** For other earlier 2013 version, errors may occur, such as "cannot open include file 'afxv_cpu.h'", or even chaos in VC++UdfStudio menu.

Secondly, please first assure the internet connection during installation. Otherwise, "WindowSDKDir" variable may be lost as following figure shows.
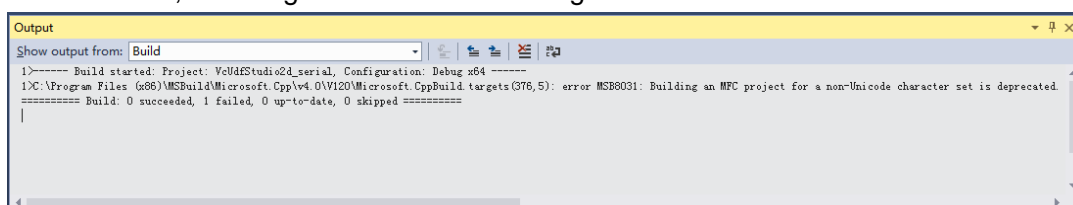

Last of all, please download and install "Visual C++ MFC Multi-Byte-Character-Set Library" first for Visual Studio 2013.

Website: https://www.microsoft.com/en-US/download/details.aspx?id=40770

Otherwise, following error will occur if using Visual Studio 2013.

Step-by-step Example:

1. Assure "Visual C#", "Visual C++" and "Visual C++ Tools" selected when selecting Visual Studio components to be installed (See previous section "Notes on Visual Studio Installation" for detailed requirements).

2. Install Service Pack 1 if you are using Visual Studio 2008. Other Visual Studio version can skip this step.

3. Start the "VC++ UDF Studio" launcher, and select the FLUENT version and VC version you'd like to use and click "OK". Click the "Browse" button to select a FLUENT

installation directory whose corresponding version is not in the list but you want to run.
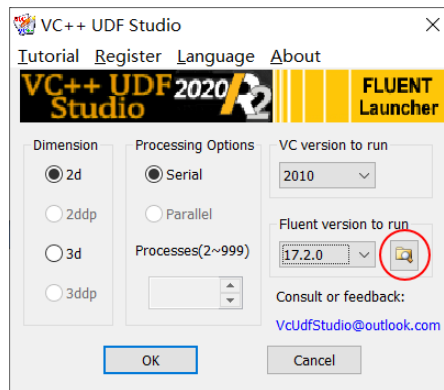
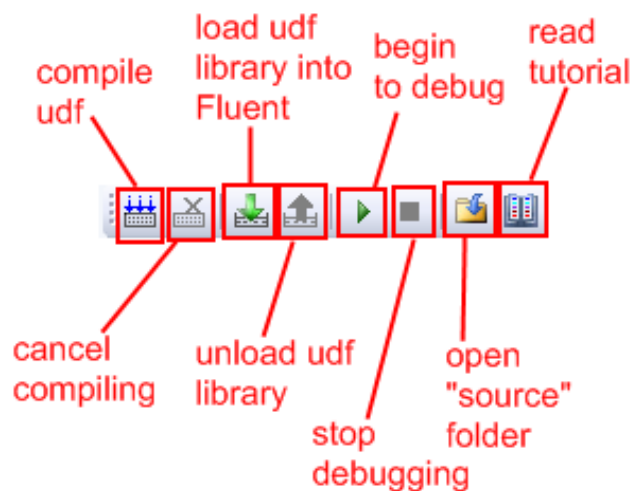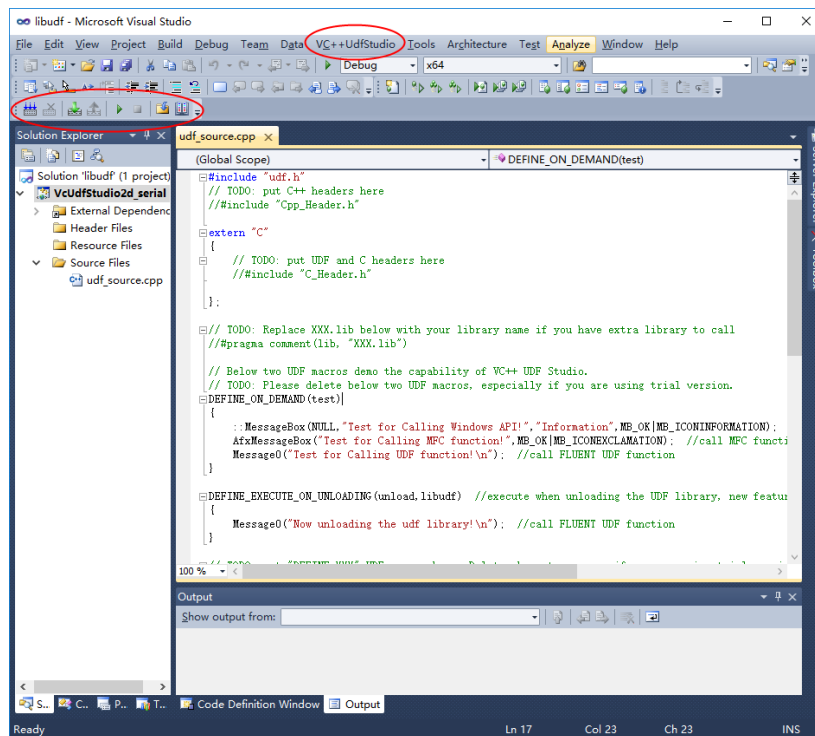4. Read a Fluent case then click "Start Visual Studio" menu and begin to code UDF.
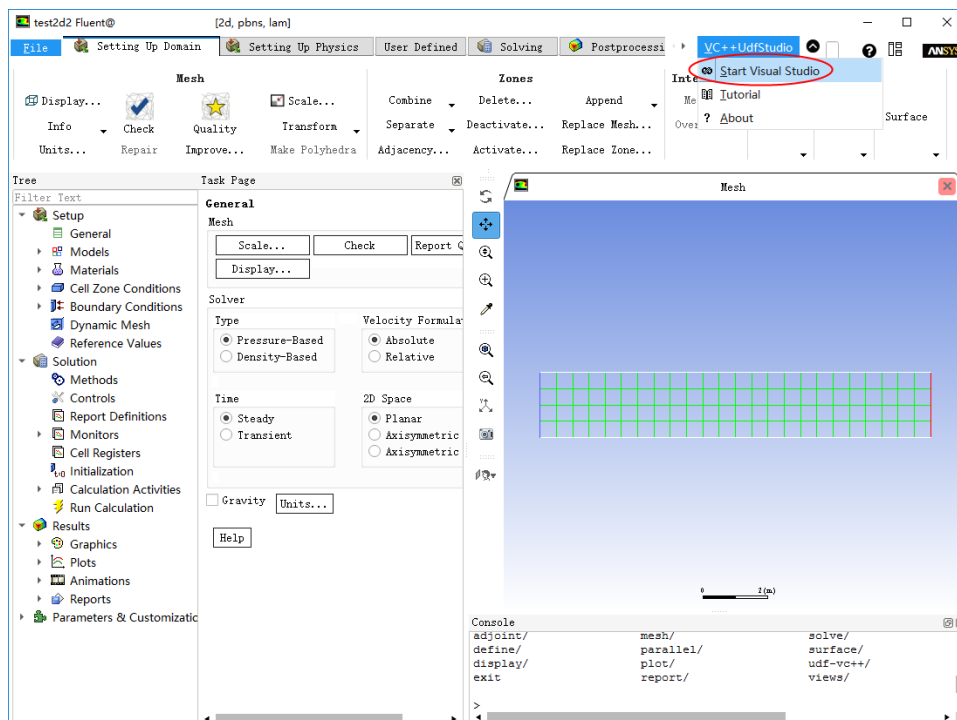
"VC++ UDF Studio" toolbar and menu will be shown in the Visual Studio. In the meantime, a folder called "source" will be created in the case directory containing all the source codes.

If "udf_source.cpp" file exists in the "source" folder, a warning message box will appear as below to confirm whether to overwrite it or not. Project files "*.vcproj" (VS2008) or "*.vcxproj"(VS2010 and later version) will be auto deleted when Visual Studio closes. Therefore, please never change the project settings. If you want to link additional library "XXX.lib", you can add #pragma comment(lib, "XXX.lib") in "udf_source.cpp".
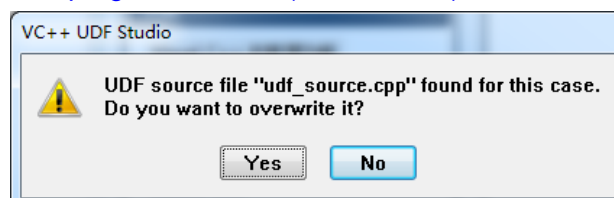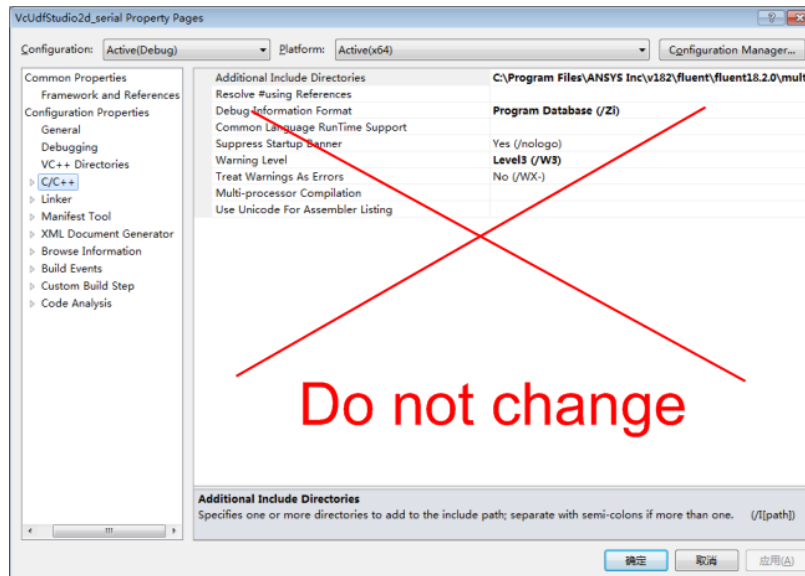
Note: All project files and intermediate folders (e.g., *.sln, *.suo, *.vcproj, *.vcxproj, *.user, *.filters, *.ncb, *.sdf, Debug folder, Release folder) EXCEPT "udf_source.cpp" and "udf_source.cpp.bak" will be auto deleted after Visual Studio closed.

5. Edit UDF source code. Add following test code to the end of "udf_source.cpp" file (If you are using trial version, please delete the DEFINE_ON_DEMAND and DEFINE_EXECUTE_ON_UNLOADING macros. Otherwise, macro over limitation error will be reported).

```
DEFINE_ON_DEMAND(debug)
{
    int aaa=123;
    int bbb=345;
    int ccc=aaa+bbb;
}
```
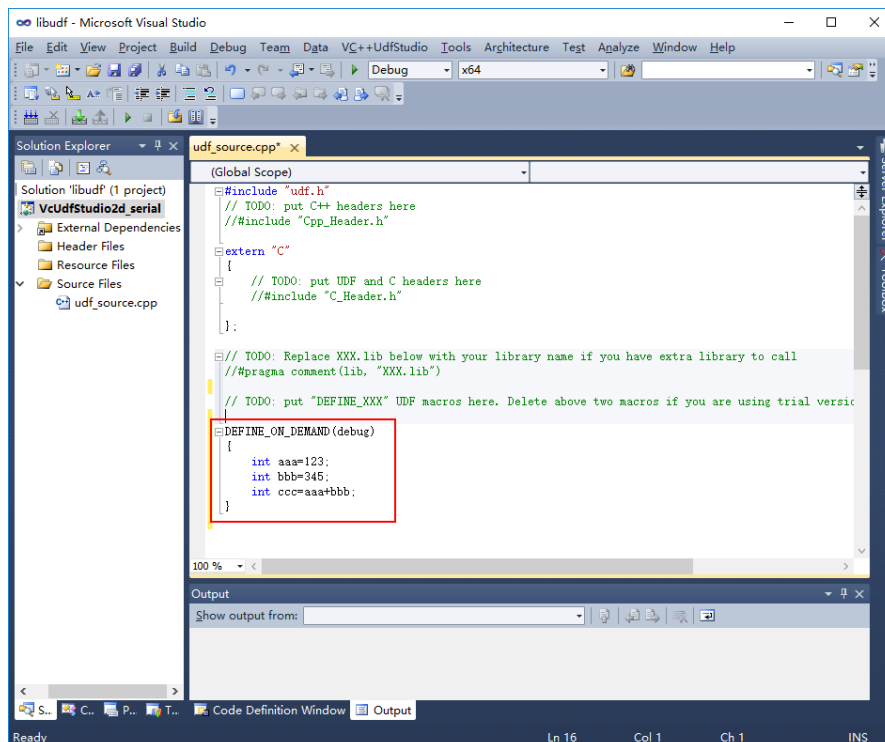
6. Click the "Build UDF library" button (or hotkey "F7") to compile the UDF source.

If no syntax errors found, then compiling will succeed (shown in below figure). You can't load or debug until the source code has been successfully built.
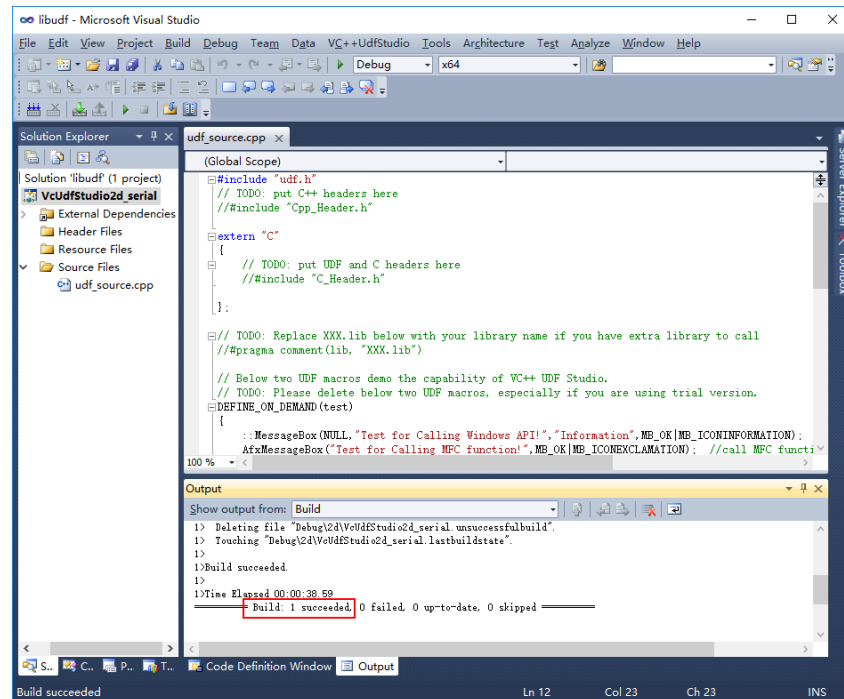
7. Click "Load UDF library to Fluent" button to load the library to FLUENT. Then, the FLUENT console should show message that "libudf" library is opened.
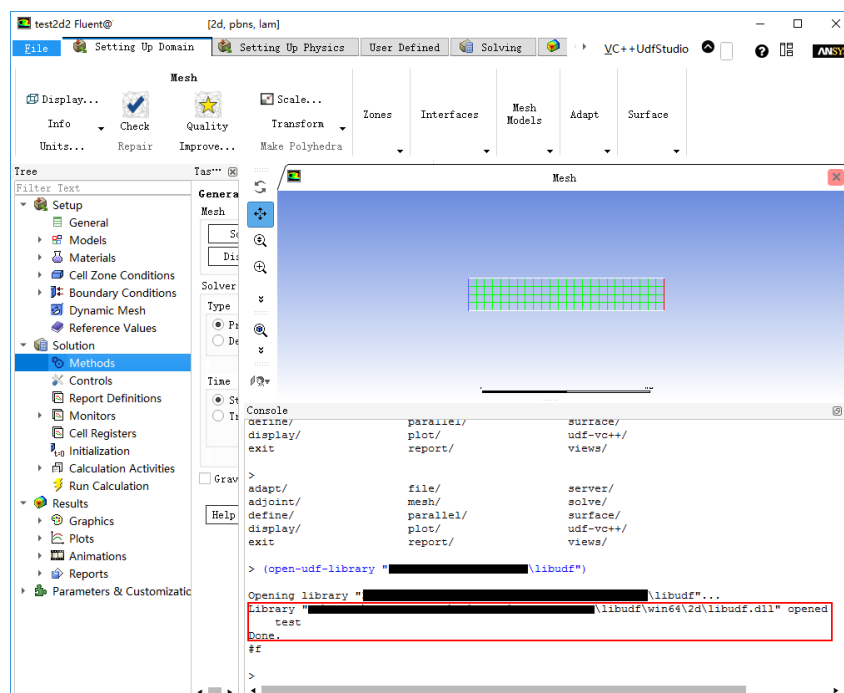
8. Set a breakpoint before "int aaa=123;" (mouse hover on this line and press hotkey

"F9") and press "Start debugging UDF library" button. Of course, directing clicking this button is also OK without loading the library first. This tool will auto load the library first for you. But anyway, UDF library must be generated successfully.

9. Execute "debug::libudf" in Fluent. Visual Studio will auto stop at the breakpoint can you can see all the variable values.

Note: Please first understand when Fluent calls the macro you want to debug. If Fluent doesn't call the macro, breakpoints in the macro won't work. For example, DEFINE_SOURCE is called during iteration and DEFINE_INIT is called when initialization. If you have set a breakpoint in DEFINE_SOURCE but haven't started iteration, program won't stop at the breakpoint.
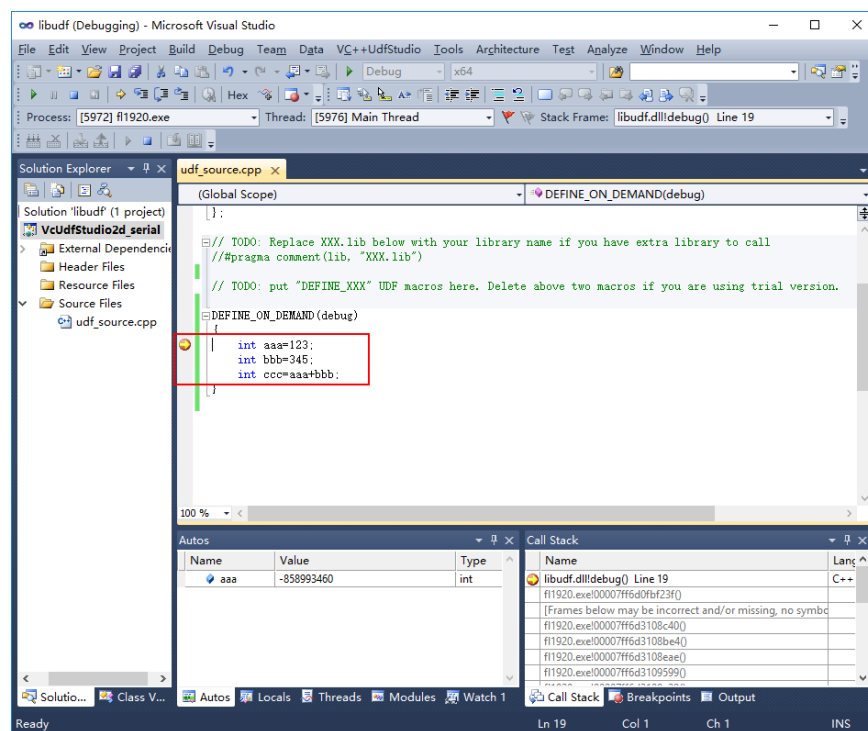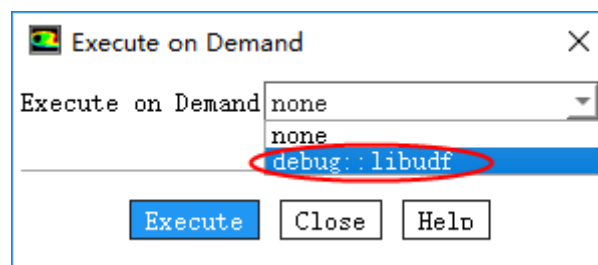
10. Step over the code line by line (or "F10" hotkey). You can observe all variable values just in the usual debugging way.

8
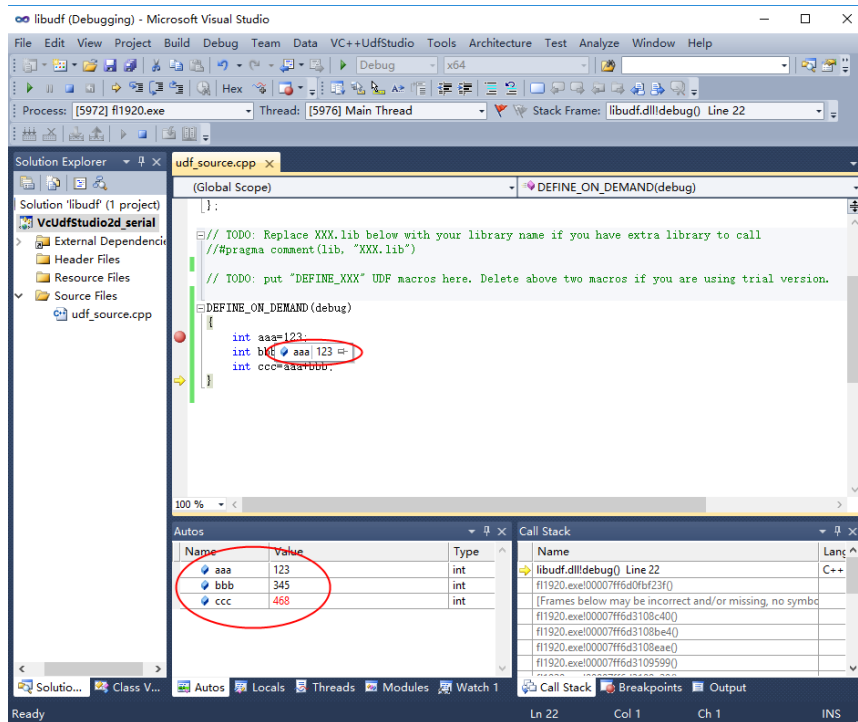
11. After all bugs removed, you can change from Debug to Release and re-compile it in release mode.
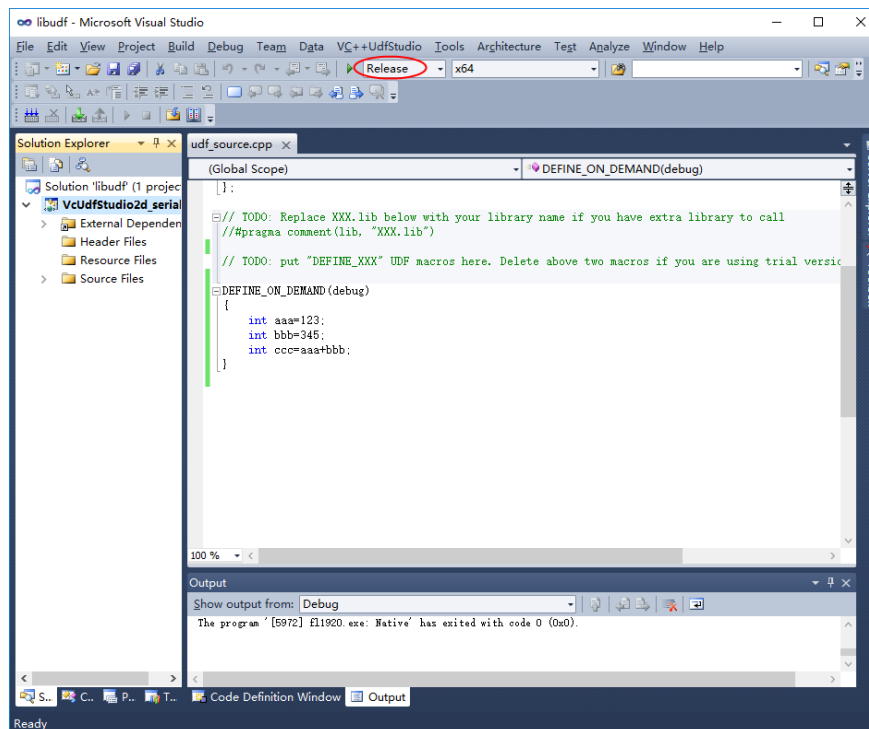
12. Now, your case folder may look like below, where "libudf" folder contains the release version of your UDF library and "source" folder stores the UDF source file "udf_source.cpp".

executable
udf library   source folder
where stores
udf_source.cpp   case file   mesh file
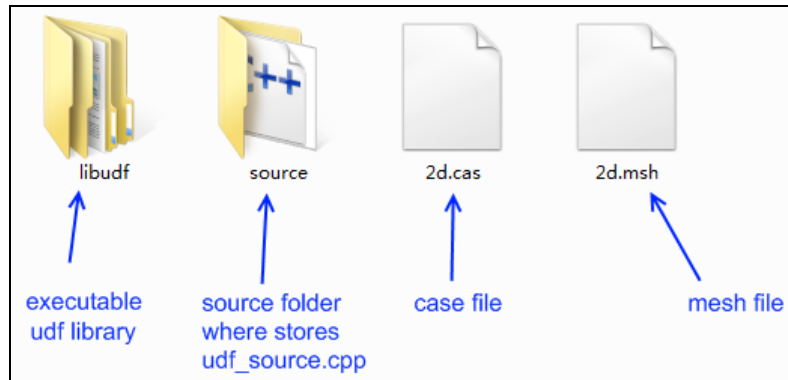
13. After successfully compiling your release version of UDF library, you needn't start Fluent from VC++ UDF Studio launcher if you only want to run the case (not modify/compile UDF source). Just start Fluent in usual way and load the UDF library by "Define->User-Defined->Functions->Manage" menu and type "libudf" in the "Library Name" edit box and press "Load" button.
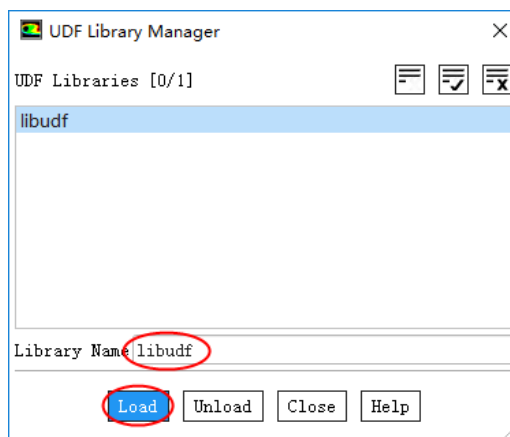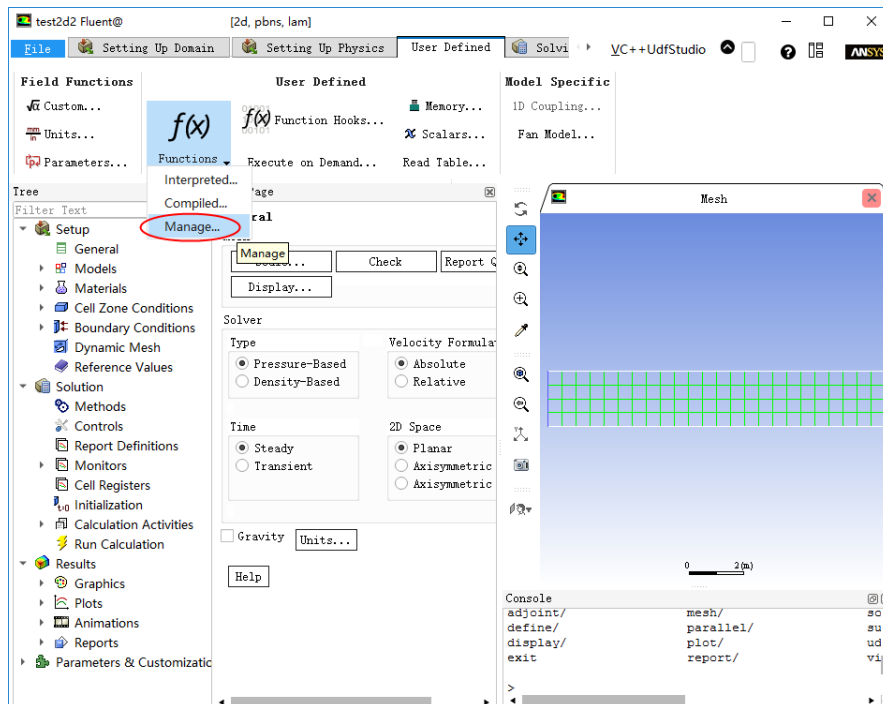
Set Up 3rd-party Directories (Enterprise Version Only):

Enterprise users can use below two buttons to set up 3rd-party header directories and library directories.

Set up 3rd-party header directories   Set up 3rd-party library directories
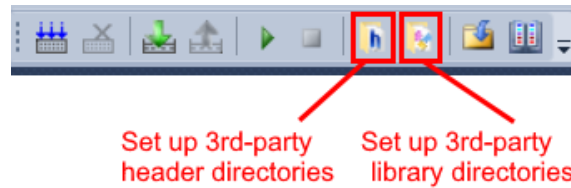
Below two figures show the dialogs after pushing these buttons. Additional 3rd-party header directories and library directories can be added by manual input or browse. $(Build-in_UDF_Include_Directories) and $(Build-in_UDF_Library_Directories) represent the necessary header directories and library directories required by VC++ UDF Studio. They are forbidden to be modified, only allowed to be moved their relative positions with additional 3rd-party directories.
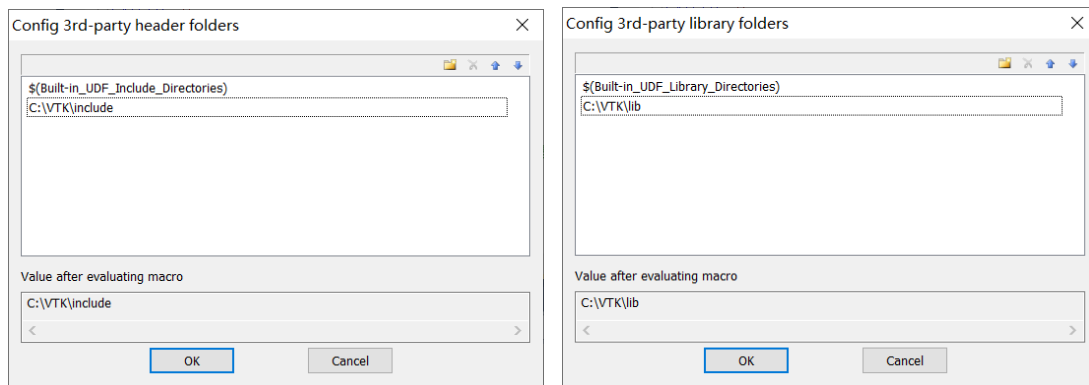
Enable SuperUdf Extension Library:

This tool packages some useful functions in the form of 3rd-party library for users' direct calling. As below figure shows, the user just needs to remove the comments of following two lines.

#include "SuperUdfExtension.h"
#pragma comment(lib, "SuperUdfExtension.lib")
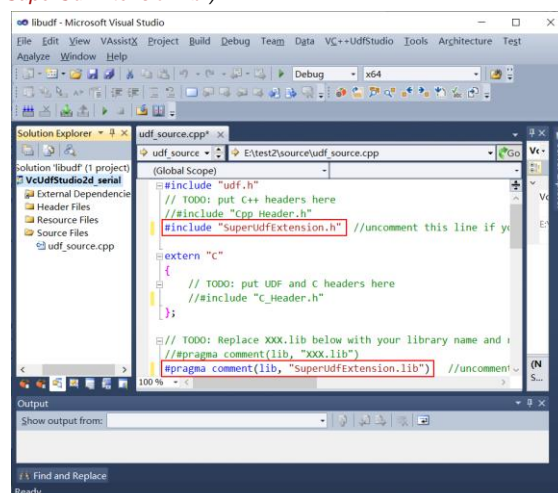
Extended Function List:

1. `void SuperUdf_Initialize(HMODULE hLibudfDllModule)`

   This function is used for the SuperUdf library initialization, where "`hLibudfDllModule`" is the `module handle of udf library. You can use AfxGetInstanceHandle() to get it (see the example in next section).`

   Note：This function has to be called before other SuperUdf extension functions. The best place to call it is in the "DEFINE_EXECUTE_ON_LOADING" macro.

2. `int SuperUdf_GetZoneIdByName(char* strZoneName)`

   Get zone ID according to zone or boundary name. For example the case in below figure, if we call SuperUdf_GetZoneIdByName("outlet"), the function will return 2.

   This function is mainly used to improve the robustness of UDF source code. As we know, LookUp_Thread(domain, zone_ID) is the common way to get Thread, where zone_ID is a key parameter. However, it will change with the input mesh. Many users have to inquire the zone ID manually and revise/recompile the UDF source code each time the input mesh changes, which is very inconvenient. After using this function, we can set a fixed name for the zone when we draw the mesh and thus the UDF source code needn't be changed anymore.
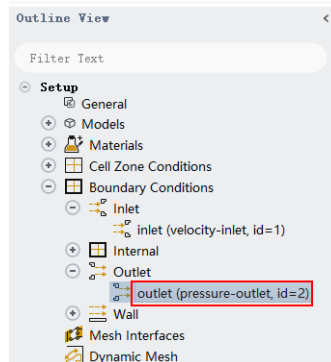
   Note：This function can only be called on serial or host. It will return -1 on node. A workaround is that we can call it on serial or host and then call host_to_node_int to send the value to node.

3. `HWND SuperUdf_GetFluentMainWnd();`

   Get the handle of Fluent main window (Enterprise version only, see programming guide in enterprise version).

4. `void SuperUdf_Steady_Iterate(int nTimes)`

   Drive Fluent to iterate n steps in steady case (Enterprise version only, see programming guide in enterprise version).

5. `void SuperUdf_ExecuteConsoleCommand(char* strAnsiConsoleCommand)`

   Drive Fluent to perform TUI or scheme command (Enterprise version only, see programming guide in enterprise version).

6. `void SuperUdf_AddUserMenu(UINT uMenuResourceID)`

   Insert user menu in Fluent (Enterprise version only, see programming guide in enterprise version).

7.   void SuperUdf_EnableMenuItem(UINT uTargetMenuID, BOOL bEnabled)

Enable or disable user menu item (Enterprise version only, see programming guide in enterprise version).

8.   void SuperUdf_SetMenuBmpAndFun(MenuItemBmpAndFun menuBmpAndFuns[], ULONG nCount)

Set the bitmaps and click action functions (Enterprise version only, see programming guide in enterprise version).

9.   void SuperUdf_SetMenuSelectCallBack(MENUSELECTPROC UserCallBackFunction)

Set the call back function of select menu. Menu items can be dynamically disabled or enabled in the call back function (Enterprise version only, see programming guide in enterprise version).

Extended Function Example (Academic Version):

```
#include "udf.h"
#include "SuperUdfExtension.h"
#pragma comment(lib, "SuperUdfExtension.lib")

DEFINE_EXECUTE_ON_LOADING(load,libudf)
{
      SuperUdf_Initialize(AfxGetInstanceHandle());
}

DEFINE_ON_DEMAND(GetOutletId)
{
      int outlet_id;
      #if !RP_NODE    // only call on serial or host
            outlet_id=SuperUdf_GetZoneIdByName("outlet"); //get the id of zone name "outlet"
      #endif
      host_to_node_int_1(outlet_id);
      Message("myid=%d, outlet id=%d\n",myid, outlet_id);
}
```

How To Register:

1. Open launcher and click "Register" menu.
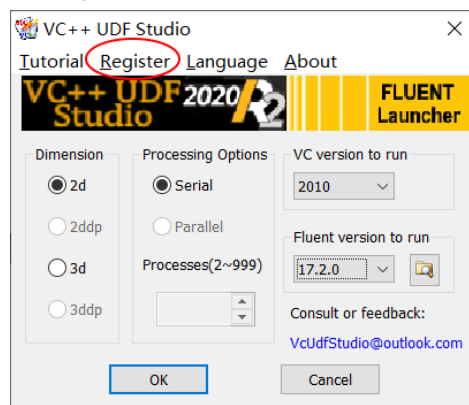
2. Input your username and leave the "Code" text box empty, then click OK. All your user name and hardware information will be put into the text file "user.ini".
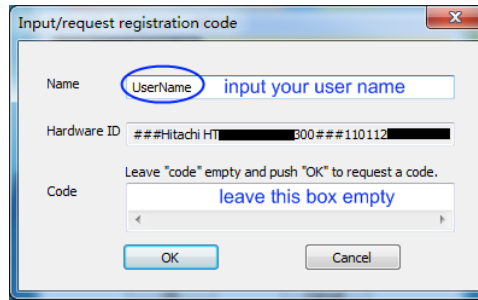
3. Attach the "user.ini" in the email to vcUdfStudio@outlook.com (International) or vcUdfStudio@sohu.com (China). The content in the "user.ini" should look like below:

Name=UserName

HardWare ID=###Hitachi HTSABC434CD300###110782PCDN403M7H6KM

4. Contact vcUdfStudio@outlook.com (International) or vcUdfStudio@sohu.com (China) and pay for the software. After receiving returned email with register code, run the Launcher as administrator and click "Register" menu again. Input the user name and register code and all functions are available now.

Tips:

1. "Visual assistant" (www. wholetomato.com) is highly recommended to install, which has a lot of extended functions (such as code completion, braces matching, user-defined keyword colors).
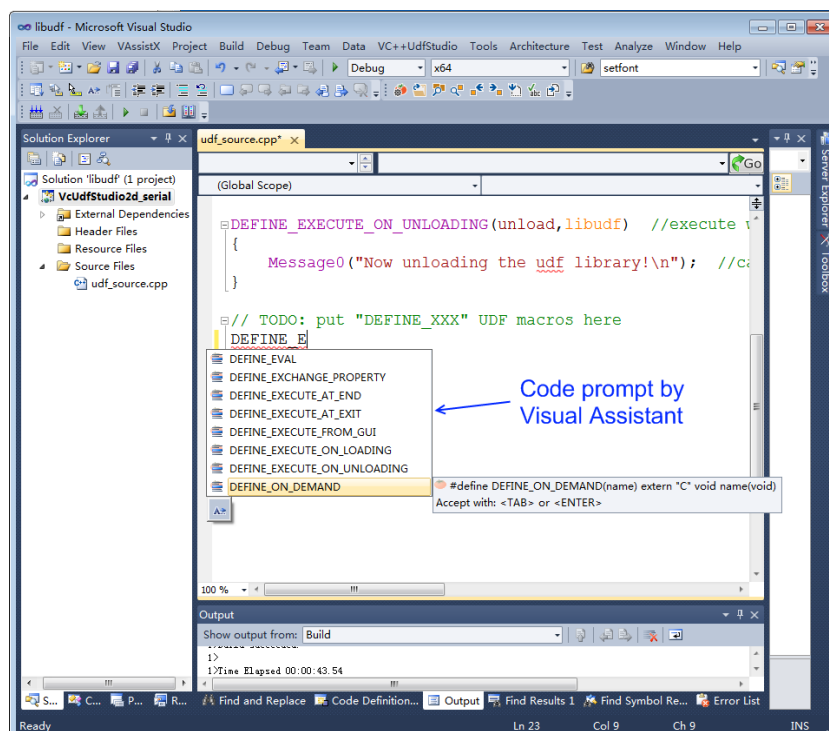
2. VC++UdfStudio menu in Fluent can be loaded or unloaded by TUI command
udf-vc++/load or *udf-vc++/unload*

```
FLUENT [2d, pbns, lam]
File  Grid  Define  Solve  Adapt  Surface  Display  Plot  Report  Parallel  VC++UdfStudio  Help
>
adapt/              grid/              surface/
define/             parallel/          udf-vc++/
display/            plot/              view/
exit                report/
file/               solve/

>
adapt/              grid/              surface/
define/             parallel/          udf-vc++/
display/            plot/              view/
exit                report/
file/               solve/

>
adapt/              grid/              surface/
define/             parallel/          udf-vc++/
display/            plot/              view/
exit                report/
file/               solve/

> u-v

/udf-vc++>
load                unload

/udf-vc++> |
```

For more information, see the website at https://vcUdfStudio.bitbucket.io

To order registered version or report bugs, please contact vcUdfStudio@outlook.com (International) or vcUdfStudio@sohu.com (China).