ZW3D API Introduction

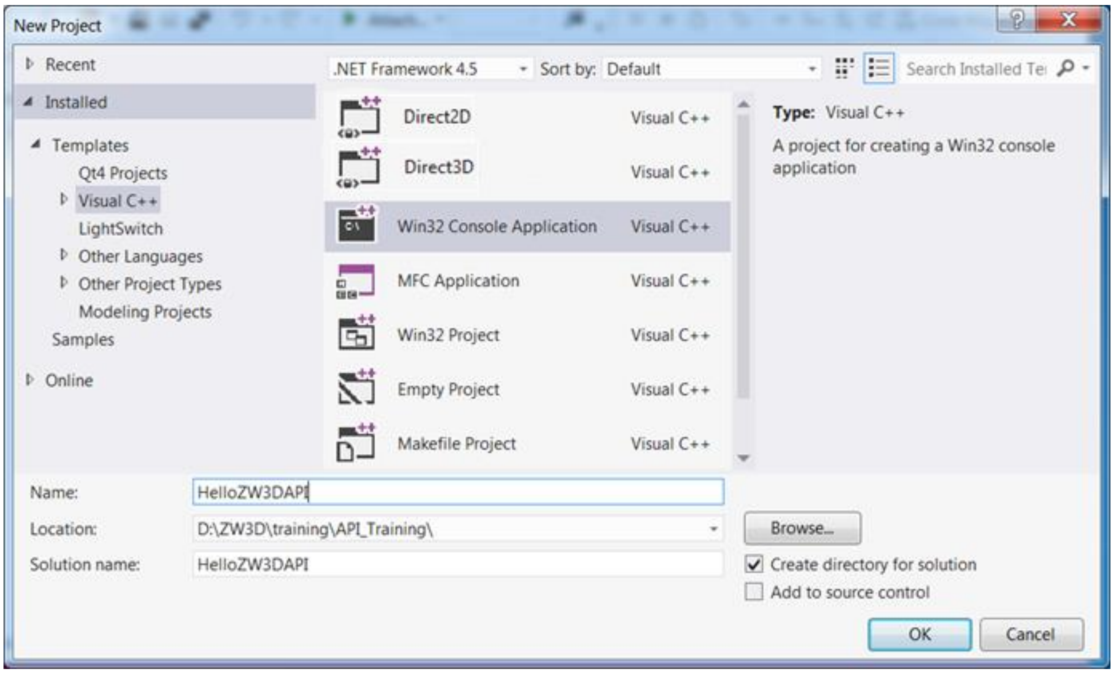
This document only gives basic introduction to ZW3D API. We recommend C/C++ for developing add-ons based on ZW3D. We will keep improving the API to make it more supportive to help our partners develop powerful applications based on ZW3D.
Should you have any suggestions or requirements, please feel free to contact us.
You could email zdn@zwsoft.comfor help.
Thanks.

Contents

ZW3D API Introduction

Chapter 1: Start with ZW3D API

1. System requirement
   a) Windows 7 or above
   b) Visual studio 2012 (or any another IDE for C/C++)
   c) ZW3D 2012 or above
2. Create a project
   a) Open Visual Studio 2012, and choose **File → New → Project…**, choose **Win32 Console Application**, then give it a name **HelloZW3DAPI**.



Type: Visual C++

A project for creating a Win32 console application

Name: HelloZW3DAPI

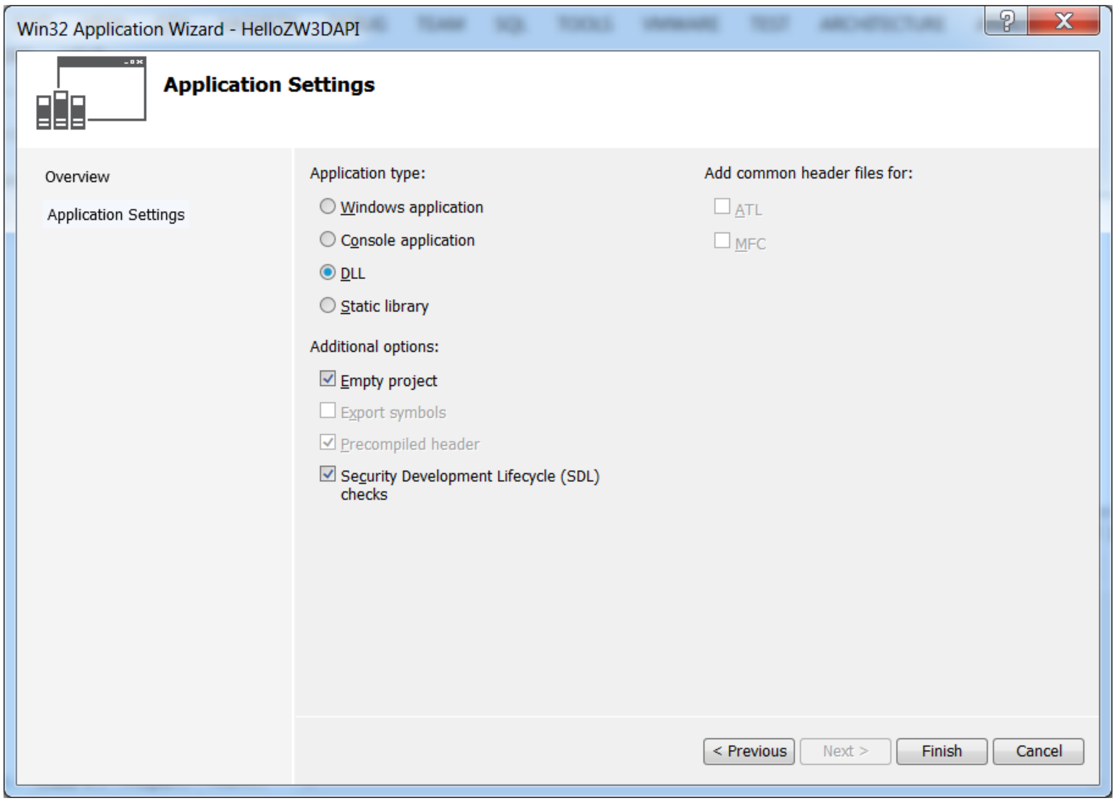
Location: D:\ZW3D\training\API_Training\

Solution name: HelloZW3DAPI

☑ Create directory for solution

☐ Add to source control

   b) Click **Next**. Then change the Type to **DLL** and set the Additional options to **Empty project**. Click **Finish**.


Win32 Application Wizard - HelloZW3DAPI

Application Settings

Application type:

( ) Windows application
( ) Console application
(●) DLL
( ) Static library

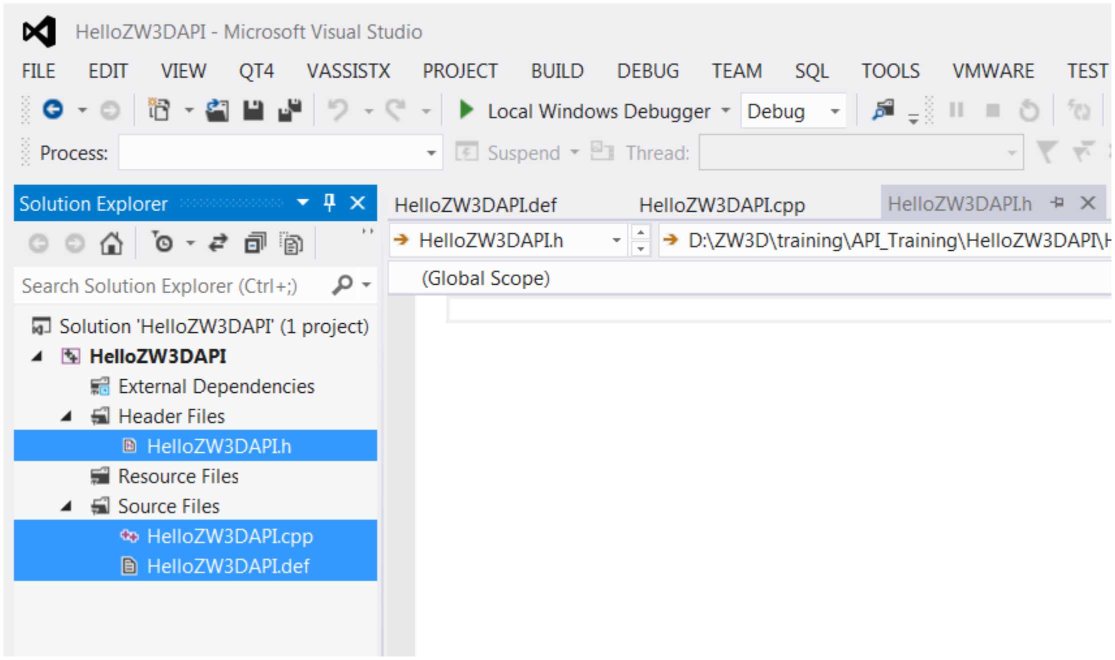
Additional options:

☑ Empty project
☐ Export symbols
☑ Precompiled header
☑ Security Development Lifecycle (SDL) checks

Add common header files for:

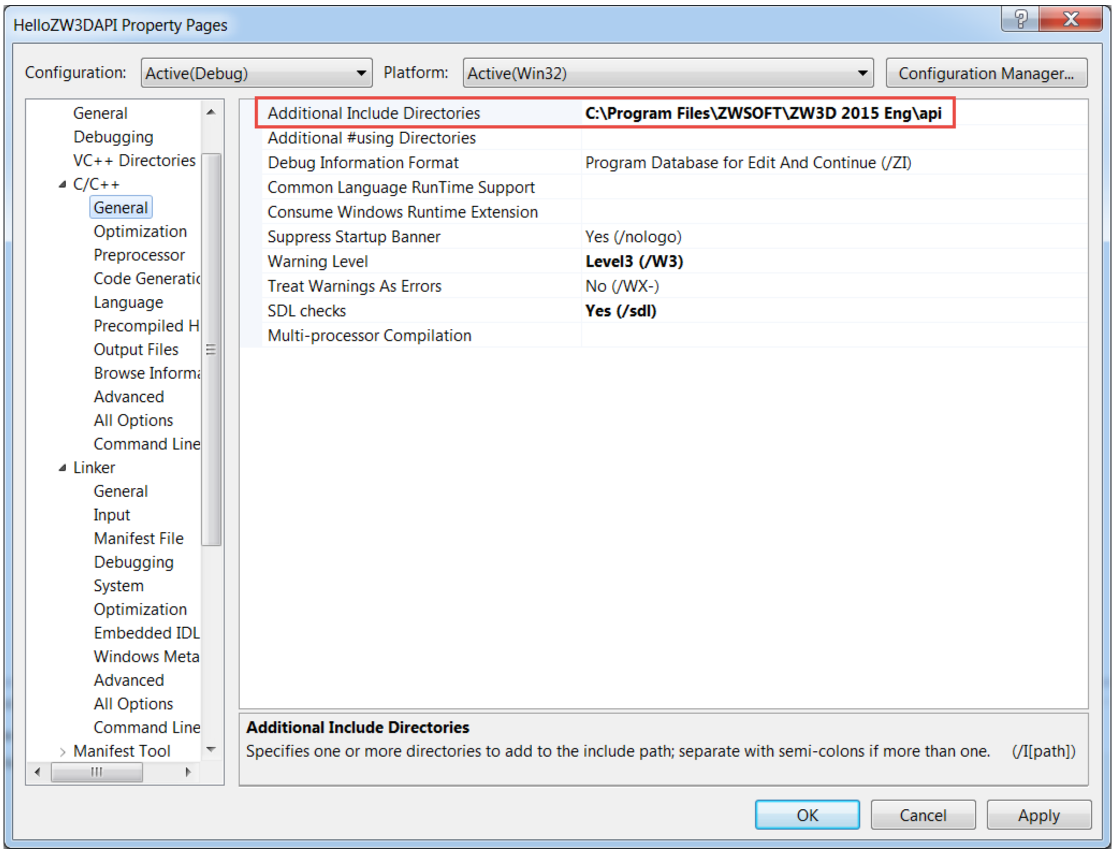
☐ ATL
☐ MFC

3. Add HelloZW3DAPI.h, HelloZW3DAPI.cpp HelloZW3DAPI.def to this project.


HelloZW3DAPI - Microsoft Visual Studio
FILE EDIT VIEW QT4 VASSISTX PROJECT BUILD DEBUG TEAM SQL TOOLS VMWARE TEST
Local Windows Debugger
Debug
Process:
Suspend
Thread:
Solution Explorer
HelloZW3DAPI.def
HelloZW3DAPI.cpp
HelloZW3DAPI.h
HelloZW3DAPI.h
D:\ZW3D\training\API_Training\HelloZW3DAPI\H
(Global Scope)
Search Solution Explorer (Ctrl+;)

Solution 'HelloZW3DAPI' (1 project)
- HelloZW3DAPI
  - External Dependencies
  - Header Files
    - HelloZW3DAPI.h
  - Resource Files
  - Source Files
    - HelloZW3DAPI.cpp
    - HelloZW3DAPI.def

4. Set the project.
   a) Right click on the project and choose Properties. Then, add the directory of ZW3D API head file to **C/C++ → General → Additional Include Directories**.



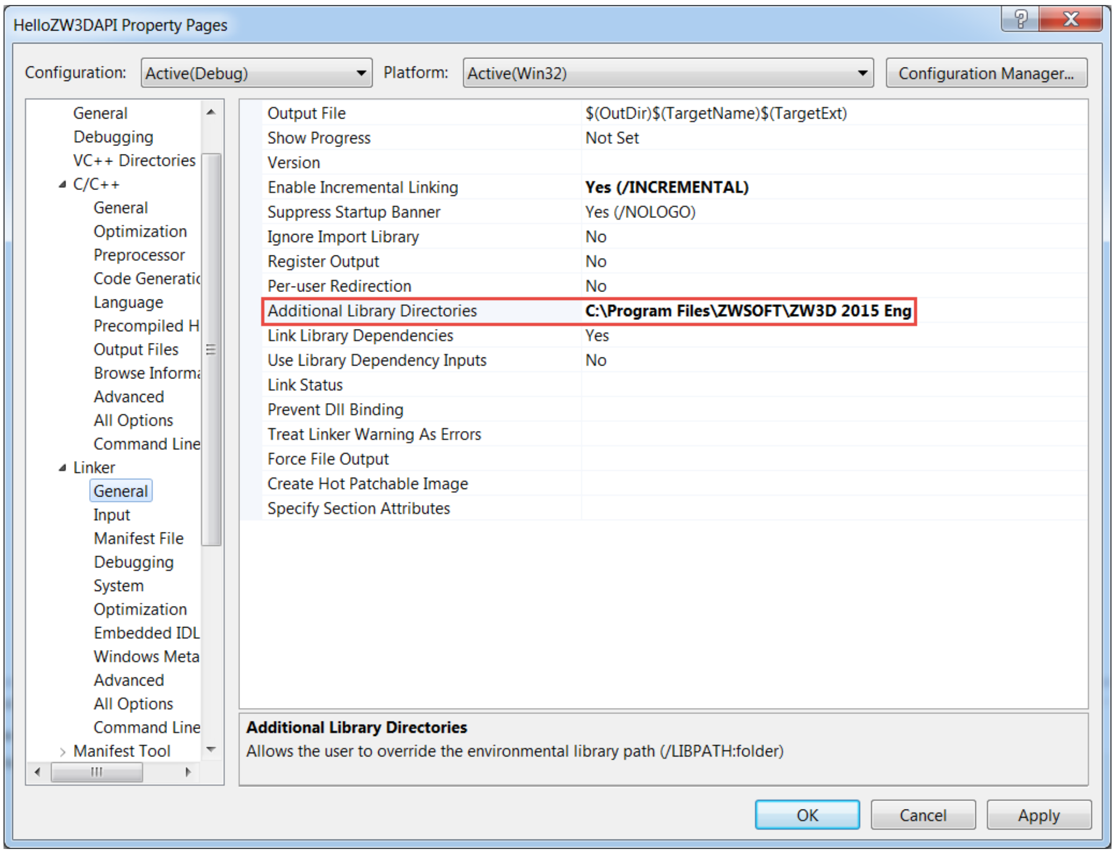
HelloZW3DAPI Property Pages

Configuration: Active(Debug)   Platform: Active(Win32)   Configuration Manager...

Additional Include Directories	**C:\Program Files\ZWSOFT\ZW3D 2015 Eng\api**
Additional #using Directories	
Debug Information Format	Program Database for Edit And Continue (/ZI)
Common Language RunTime Support	
Consume Windows Runtime Extension	
Suppress Startup Banner	Yes (/nologo)
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
SDL checks	**Yes (/sdl)**
Multi-processor Compilation	

Additional Include Directories
Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])

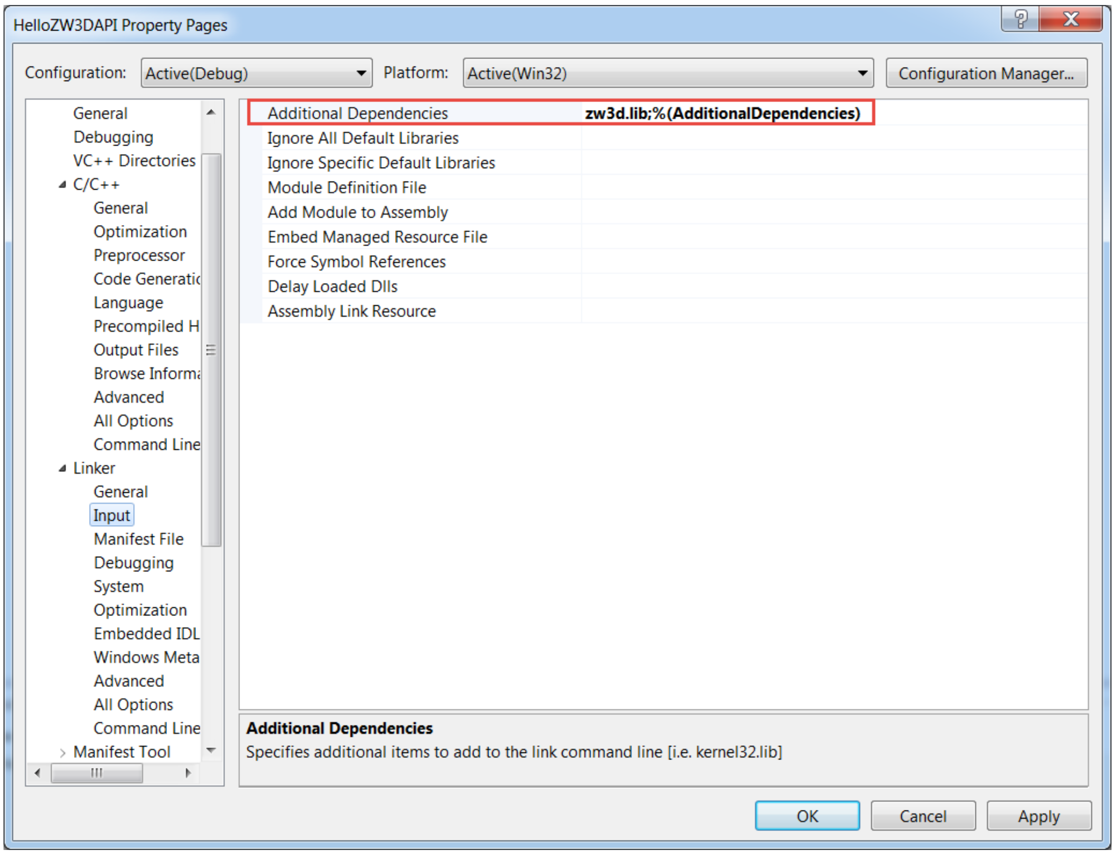
b) Add the directory of ZW3D API library file to **Linker→ General → Additional Library Directories**.



Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	**C:\Program Files\ZWSOFT\ZW3D 2015 Eng**
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Additional Library Directories
Allows the user to override the environmental library path (/LIBPATH:folder)

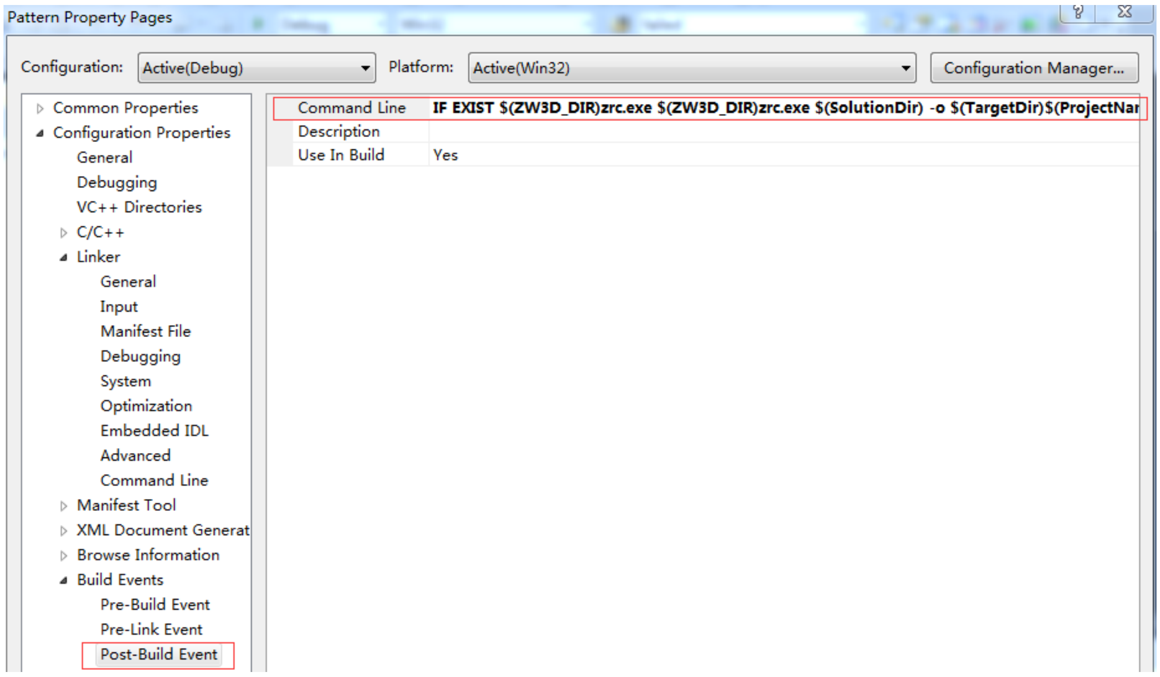
c) Add ZW3D API library to **Linker→ Input → Additional Dependencies**.



Additional Dependencies	**zw3d.lib;%(AdditionalDependencies)**
Ignore All Default Libraries	
Ignore Specific Default Libraries	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded Dlls	
Assembly Link Resource	

Additional Dependencies
Specifies additional items to add to the link command line [i.e. kernel32.lib]

d) To compile resource,Add command line to **Build Events→Post-Build Event→Command Line**.



Configuration: Active(Debug) Platform: Active(Win32)

Command Line	IF EXIST $(ZW3D_DIR)zrc.exe $(ZW3D_DIR)zrc.exe $(SolutionDir) -o $(TargetDir)$(ProjectNar
Description	
Use In Build	Yes

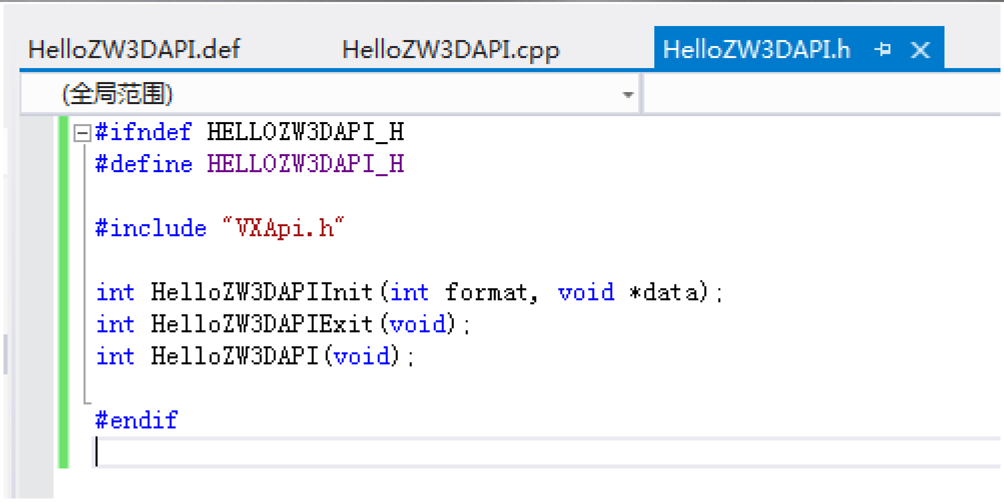
(Note: If there is no resource file in your project. Such as image/UI. This step is optional)
The contents as shown below:

```
IF EXIST "$(ZW3D_DIR)zrc.exe" "$(ZW3D_DIR)zrc.exe" "$(SolutionDir)\." -o
"$(TargetDir)$(ProjectName).zrc"
```

(Note:ZW3D_DIR is an environment variable whose value is ZW3D installation path)

5. Define the functions in HelloZW3DAPI.h. You can copy the following code directly.

```
#ifndefHELLOZW3DAPI_H
#defineHELLOZW3DAPI_H

#include"VXApi.h"

IntHelloZW3DAPIInit(int format, void *data);
IntHelloZW3DAPIExit(void);
IntHelloZW3DAPI(void);

#endif
```


HelloZW3DAPI.def HelloZW3DAPI.cpp HelloZW3DAPI.h

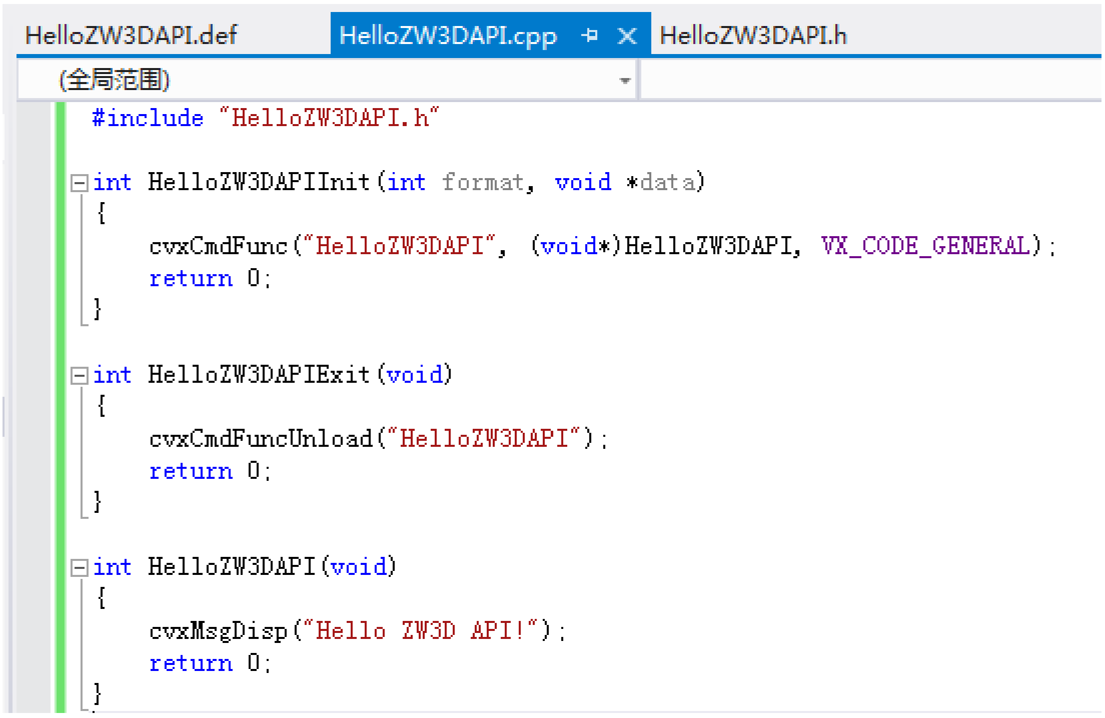
(全局范围)

```
#ifndef HELLOZW3DAPI_H
#define HELLOZW3DAPI_H

#include "VXApi.h"

int HelloZW3DAPIInit(int format, void *data);
int HelloZW3DAPIExit(void);
int HelloZW3DAPI(void);

#endif
```

Note: The prefix of the functions must be the same as the project, which means @customizedApp@Init() and @customizedApp@Exit() are the entrance functions of the application. When @customizedApp@.dll is loaded by ZW3D, @customizedApp@Init() and @customizedApp@Exit() will be checked to know the customized functions. *(@customizedApp@ meansthe name of any project you have created.)*

6. Implement the functions in HelloZW3DAPI.cpp. You can copy the following code directly.

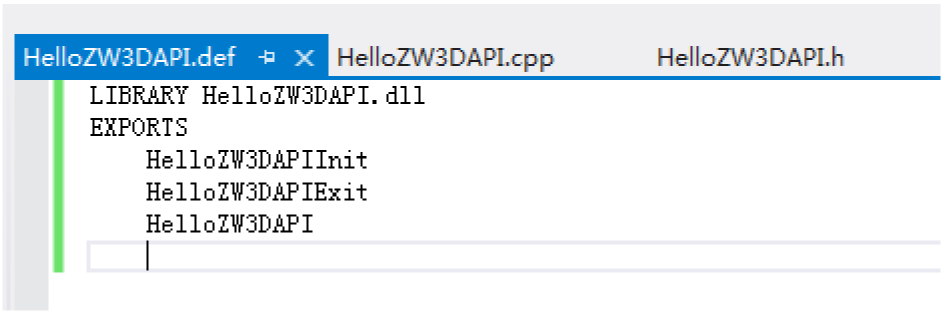
```
#include"HelloZW3DAPI.h"

IntHelloZW3DAPIInit(int format, void *data)
{
 cvxCmdFunc("HelloZW3DAPI", (void*)HelloZW3DAPI, VX_CODE_GENERAL);

 return0;
}

IntHelloZW3DAPIExit(void)
{
 cvxCmdFuncUnload("HelloZW3DAPI");

 return0;
}

IntHelloZW3DAPI(void)
{
 cvxMsgDisp("Hello ZW3D API!");

 return0;
}
```



```
#include "HelloZW3DAPI.h"

int HelloZW3DAPIInit(int format, void *data)
{
    cvxCmdFunc("HelloZW3DAPI", (void*)HelloZW3DAPI, VX_CODE_GENERAL);
    return 0;
}

int HelloZW3DAPIExit(void)
{
    cvxCmdFuncUnload("HelloZW3DAPI");
    return 0;
}

int HelloZW3DAPI(void)
{
    cvxMsgDisp("Hello ZW3D API!");
    return 0;
}
```

7. Define the Module Definition file, HelloZW3DAPI.def.
   You can copy the following code directly.

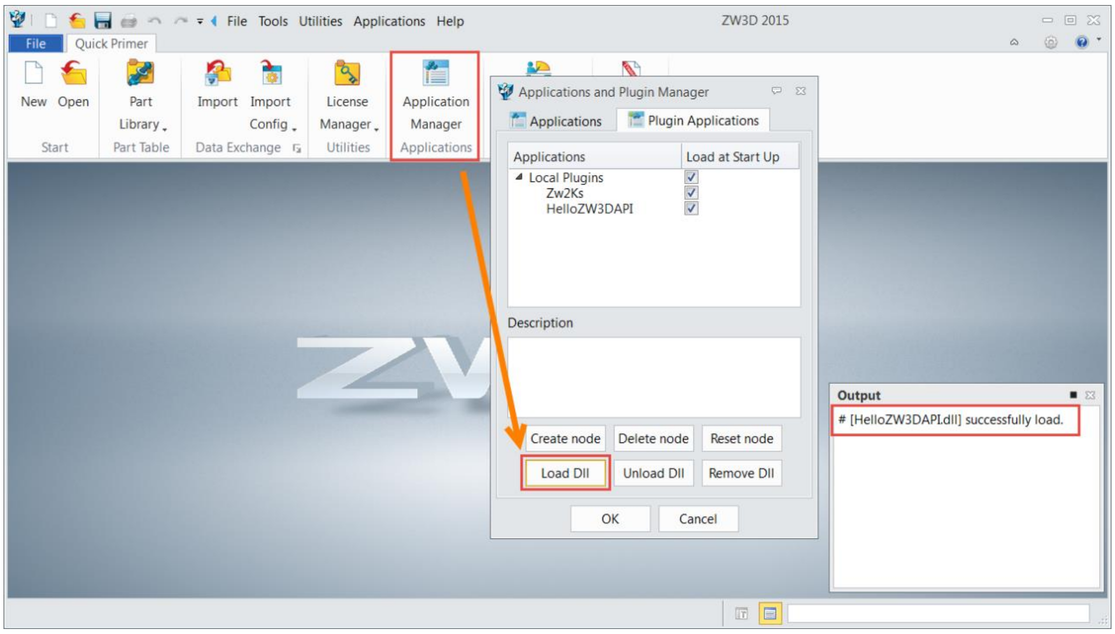
```
LIBRARY HelloZW3DAPI.dll
EXPORTS
    HelloZW3DAPIInit
    HelloZW3DAPIExit
    HelloZW3DAPI
```



```
LIBRARY HelloZW3DAPI.dll
EXPORTS
    HelloZW3DAPIInit
    HelloZW3DAPIExit
    HelloZW3DAPI
```

8. Build the project.
   Right click on this project to build the project. Then, you can find HellowZW3DAPI.dll in the directory ".\HelloZW3DAPI\Debug\HelloZW3DAPI.dll".

9. Load HelloZW3DAPI.dll.
   a) Use ZW3D Applications Manager to load the DLL and a message in the Output dialog will show whether the command fails or succeeds.
      Note: This path will be remembered in registration when ZW3D is closed, and next time it will automatically follow the path to load the file.



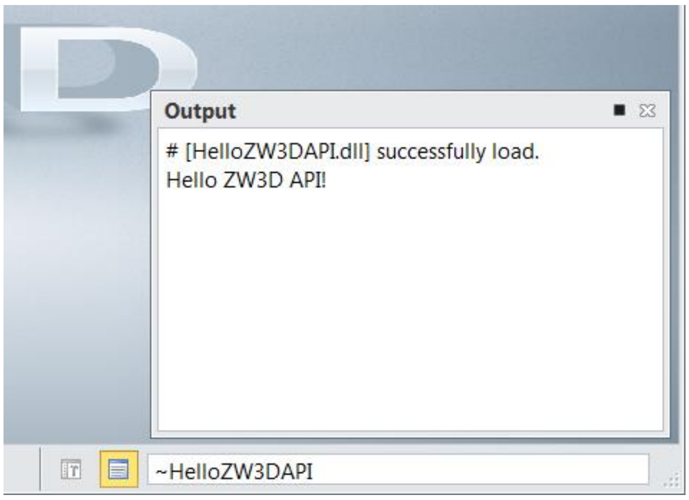
```
# [HelloZW3DAPI.dll] successfully load.
```

b) Copy HelloZW3DAPI.dll to the installation folder. “C:\Program Files\ZWSOFT\ZW3D 2015 Eng\apilibs”, then start ZW3D.

Note: ZW3D will search for applications in “.\apilibs”. If any, they will be loaded.

10. Run this application.

Input “~HelloZW3DAPI” in the command line, then press Enter. You can find “Hello ZW3D API!” was shown in the Output dialog.



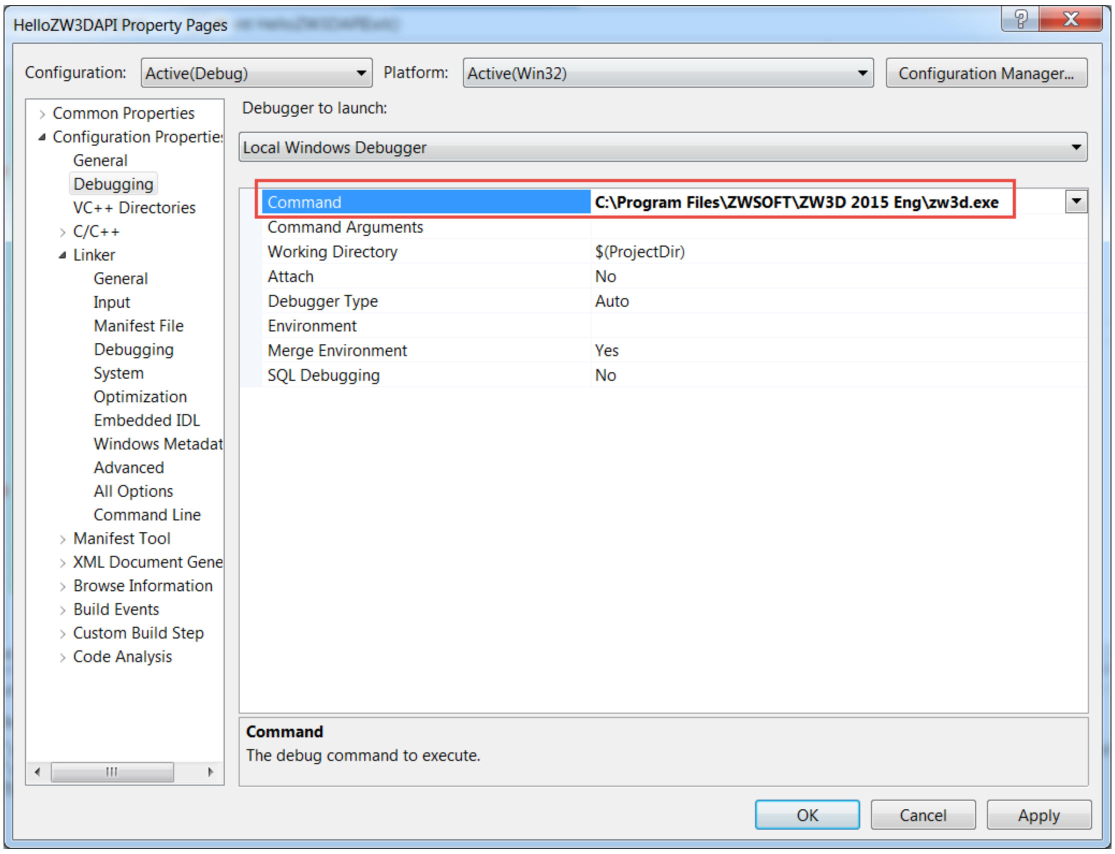
```
# [HelloZW3DAPI.dll] successfully load.
Hello ZW3D API!
```

```
~HelloZW3DAPI
```

Note: Function name can be defined to any name, andit is not necessary to be the same as the project. But it is a good habit to define it the same as the name of project.

11. Debug the application.

a) Right click on the project and set the Debugging Command as below. Then run “Local Windows Debugger” or press F5 to start the debugger.


HelloZW3DAPI Property Pages

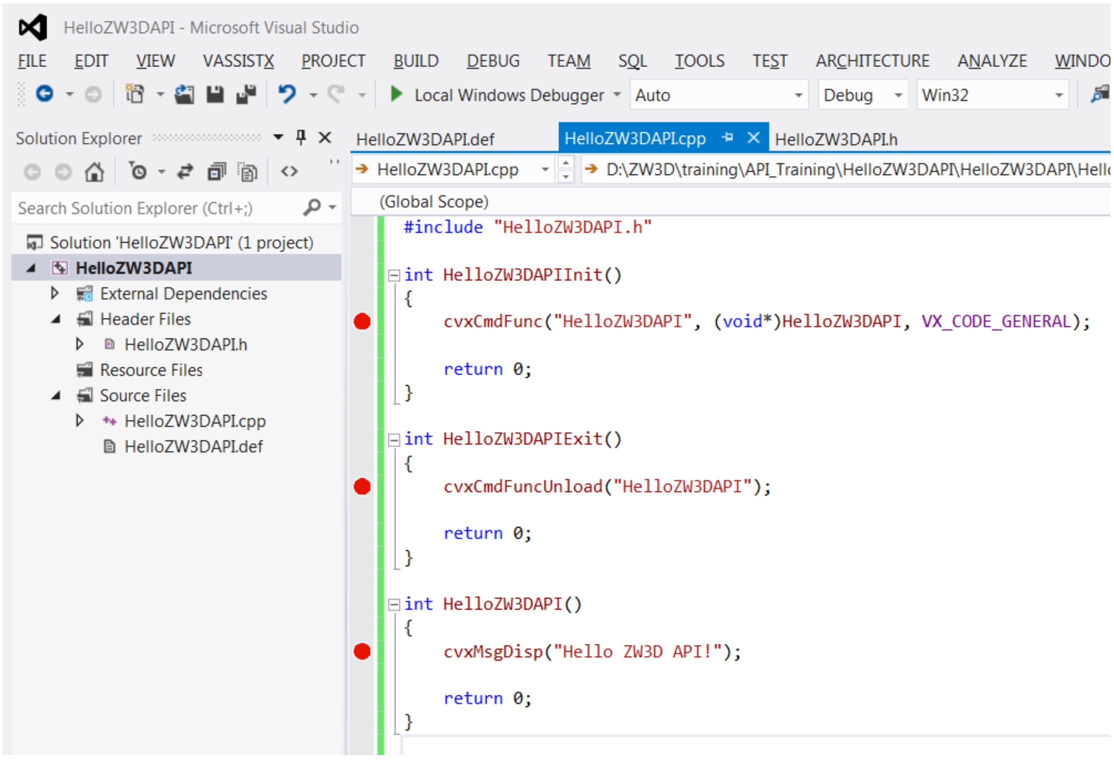
Configuration: Active(Debug) Platform: Active(Win32) Configuration Manager...

Debugger to launch: Local Windows Debugger

Command	C:\Program Files\ZWSOFT\ZW3D 2015 Eng\zw3d.exe
Command Arguments	
Working Directory	$(ProjectDir)
Attach	No
Debugger Type	Auto
Environment	
Merge Environment	Yes
SQL Debugging	No

Command
The debug command to execute.

b) Set the break point for each function.


HelloZW3DAPI - Microsoft Visual Studio
FILE EDIT VIEW VASSISTX PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE ANALYZE WINDO
Local Windows Debugger
Auto
Debug
Win32
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'HelloZW3DAPI' (1 project)
HelloZW3DAPI
External Dependencies
Header Files
HelloZW3DAPI.h
Resource Files
Source Files
HelloZW3DAPI.cpp
HelloZW3DAPI.def
HelloZW3DAPI.def
HelloZW3DAPI.cpp
HelloZW3DAPI.h
HelloZW3DAPI.cpp
D:\ZW3D\training\API_Training\HelloZW3DAPI\HelloZW3DAPI\Hell
(Global Scope)

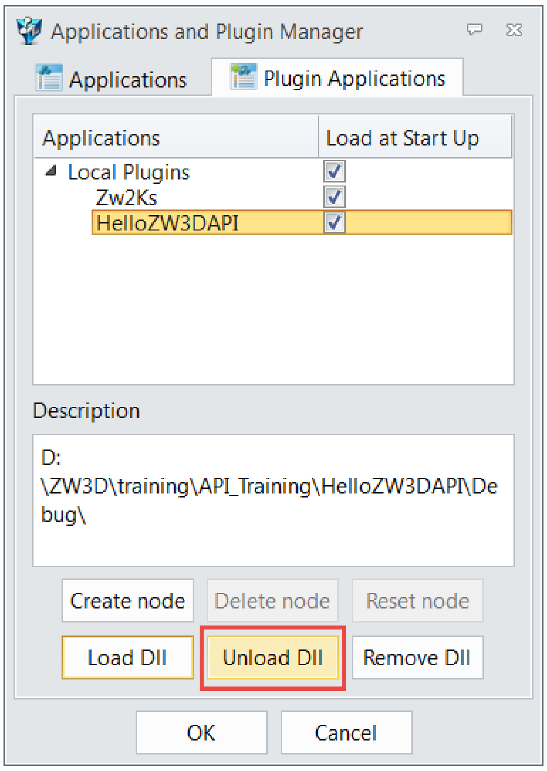
```
#include "HelloZW3DAPI.h"

int HelloZW3DAPIInit()
{
    cvxCmdFunc("HelloZW3DAPI", (void*)HelloZW3DAPI, VX_CODE_GENERAL);

    return 0;
}

int HelloZW3DAPIExit()
{
    cvxCmdFuncUnload("HelloZW3DAPI");

    return 0;
}

int HelloZW3DAPI()
{
    cvxMsgDisp("Hello ZW3D API!");

    return 0;
}
```

c) Reference step 9 → a) to Load the DLL, you will find the following situations.

i. The debugger will get into HelloZW3DAPIInit() when the DLL is loaded.

ii. The debugger will get into HelloZW3DAPI() when "~HelloZW3DAPI" (step 10) is run.

iii. The debugger will get into HelloZW3DAPIExit() when the application is unloaded.



Applications and Plugin Manager

Applications	Load at Start Up
Local Plugins	✓
Zw2Ks	✓
HelloZW3DAPI	✓

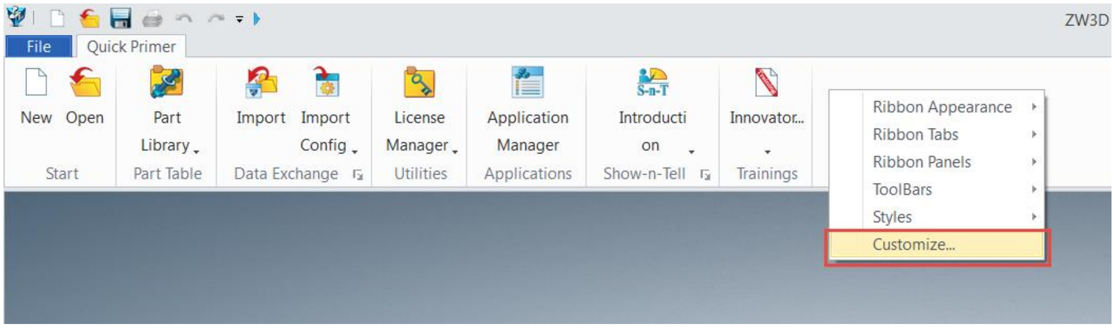
Description

D:
\ZW3D\training\API_Training\HelloZW3DAPI\Debug\

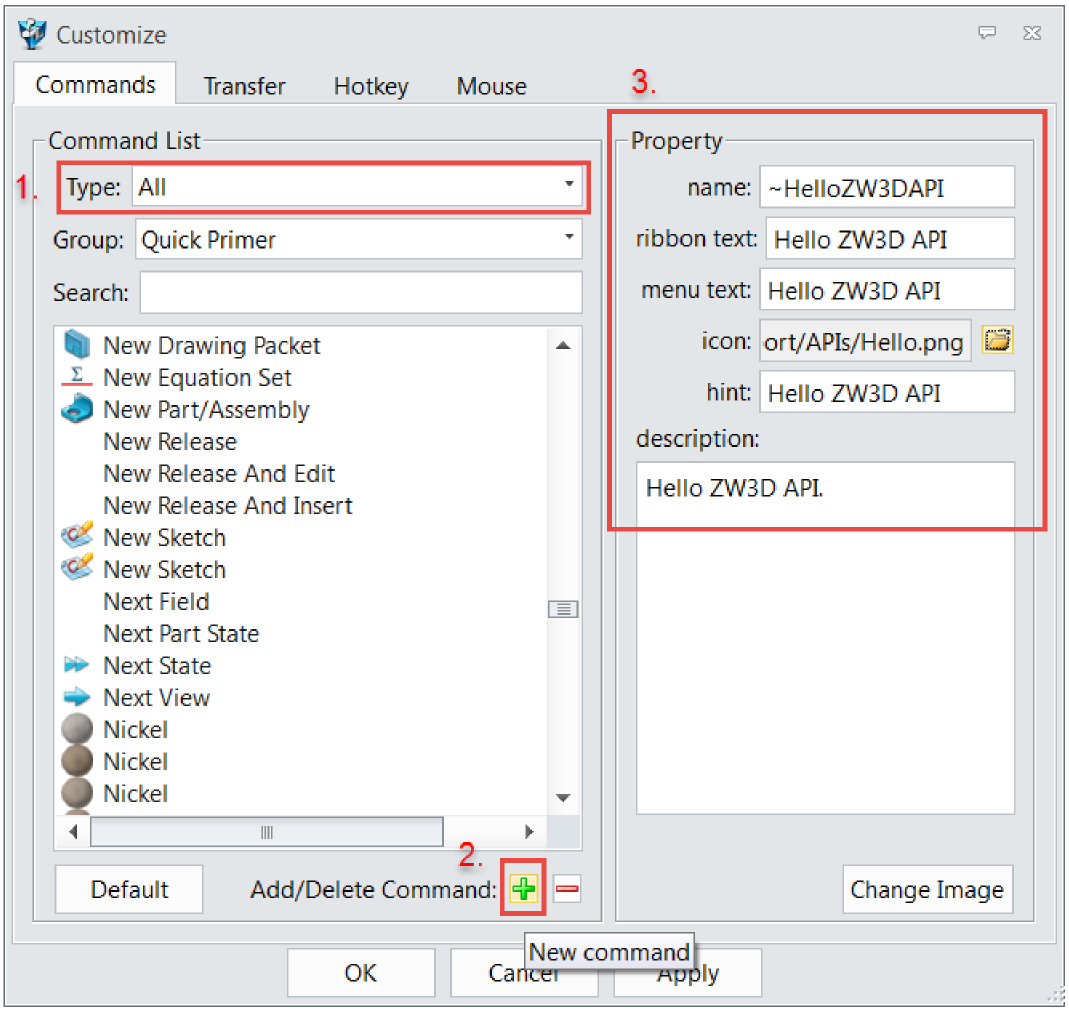
Chapter2: Customized Menu / Ribbon / Toolbar

We create the first customized command in ZW3D. But input the command is complex. I will introduce how to put this command in the Menu / Ribbon / Toolbar.

1. Open the ZW3D and right click on the space of the ribbon. Then, click **Customize…**

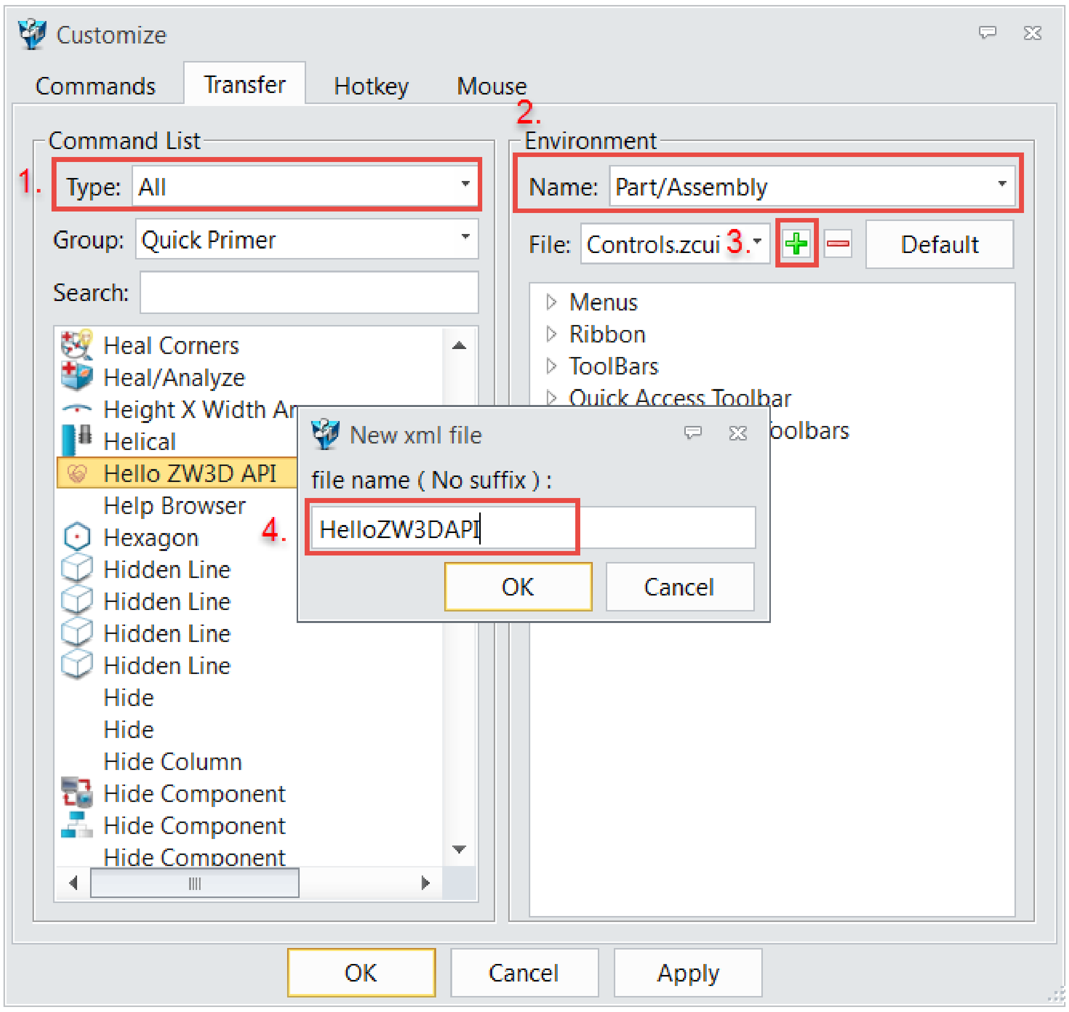


2. Define your command.
   a) Change the type to **All**.
   b) Add a new command.
   c) Change the Property of the new command.

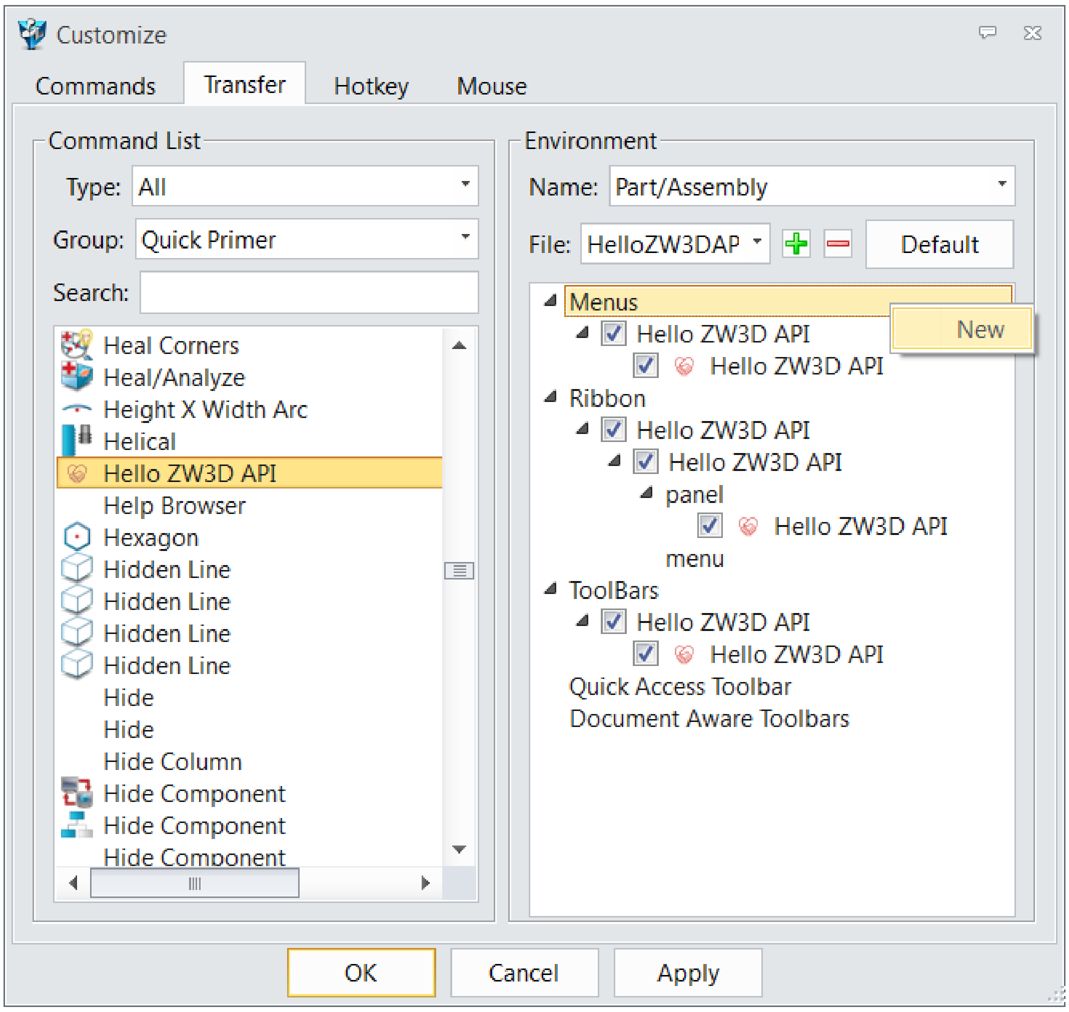


Property
name: ~HelloZW3DAPI
ribbon text: Hello ZW3D API
menu text: Hello ZW3D API
icon: ort/APIs/Hello.png
hint: Hello ZW3D API
description:
Hello ZW3D API.

3. Change the tap to **Transfer** to create your Own Menu / Ribbon / Toolbar.
   a) Change the Type to **All**.
   b) Change the Environment to **Part/Assembly**.
   c) Add a new xml file.
   d) Give the name to **HelloZW3DAPI**.

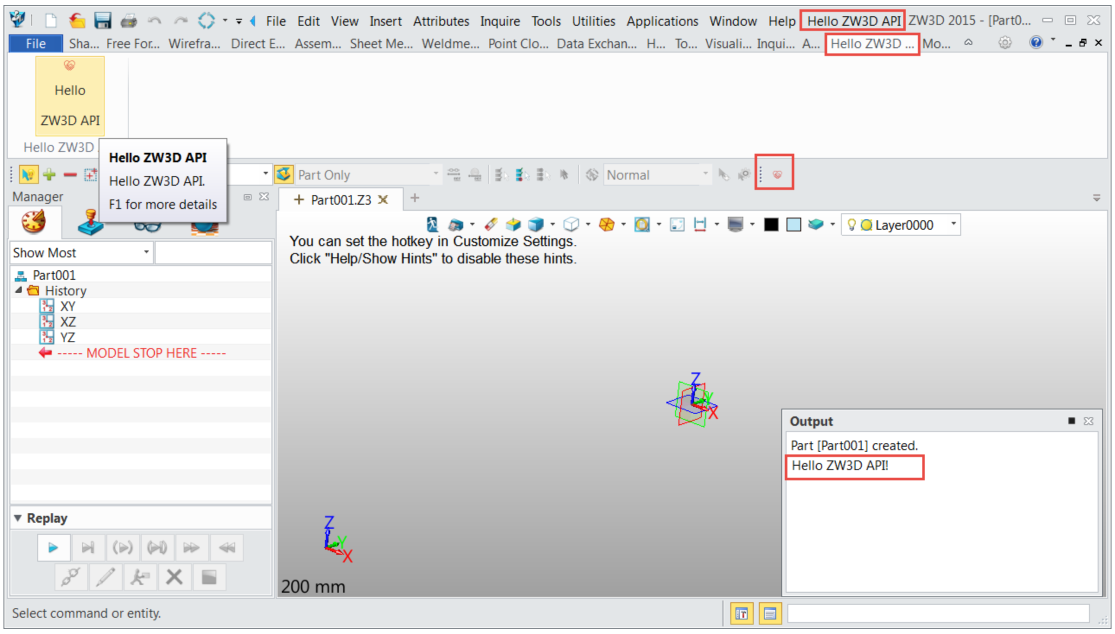


   e) Customize the new xml file.
      i. Right click on the Menus / Ribbon / ToolBars to create new items.
      ii. Drag the "Hello ZW3D API" command from left to the new items you just have created.
      iii. Please refer to the following picture to create the same Menu / Ribbon / Toobar.



f) Click **OK** to finish the customization.

g) Then, create a new part, and you can find the Menu / Ribbon / Toolbar there. You can press the button to run the command.


File Edit View Insert Attributes Inquire Tools Utilities Applications Window Help Hello ZW3D API
Hello ZW3D API
Hello ZW3D API
Hello ZW3D API.
F1 for more details
Part001
History
XY
XZ
YZ
----- MODEL STOP HERE -----
Replay
Output

You can set the hotkey in Customize Settings.
Click "Help/Show Hints" to disable these hints.

Part [Part001] created.
Hello ZW3D API!

Select command or entity.

Note: if you get this warning: "ALERT: Unable to find HelloZW3DAPI: No such symbol." It means ZW3D cannot find your command. You need to load the DLL first. Please refer to Chapter 1, point 9 to load the DLL.

4. Share the customization to other people.

   a) Get the customized UI XML file (HelloZW3DAPI.zcui) from the user's folder:

      If your ZW3D version is prior to 2020, get from:

      %appdata%\ZW3D 2015Eng\profiles\Default\Environment-2\Controls

      If your ZW3D version after 2020(included), get from:

%appdata%\ZWSOFT\ZW3D\ZW3D 2022Eng\custom\profiles\Default\Environment-2\Controls

   b) Get the customized Command XML file (User.zcui) from the user's folder:

      If your ZW3D version is prior to 2020, get from:

      %appdata%\ZW3D 2015Eng\profiles\Default\Action

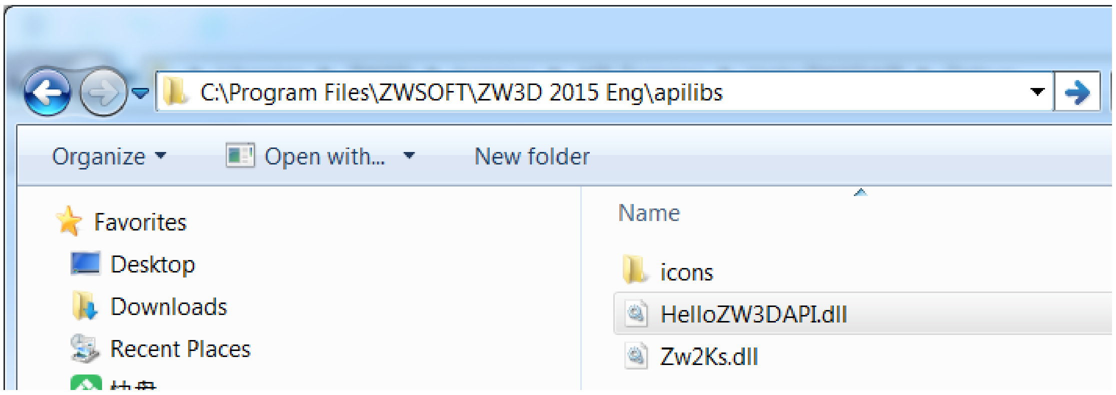
      If your ZW3D version is after 2020, get from:

%appdata% \ZWSOFT\ZW3D \ZW3D 2022Eng\custom\profiles\Default\Action

   c) Copy the files to the same directory in another PC.

   d) Copy the DLL and the image to the installation directory, and the image must be put in the folder which named as icons.

      C:\Program Files\ZWSOFT\ZW3D 2015 Eng\apilibs



Note: you can rename the XML to any other name. But you must put them in the right directory. ZW3D will load them automatically.

Chapter 3: ZW3D UI Designer introduction

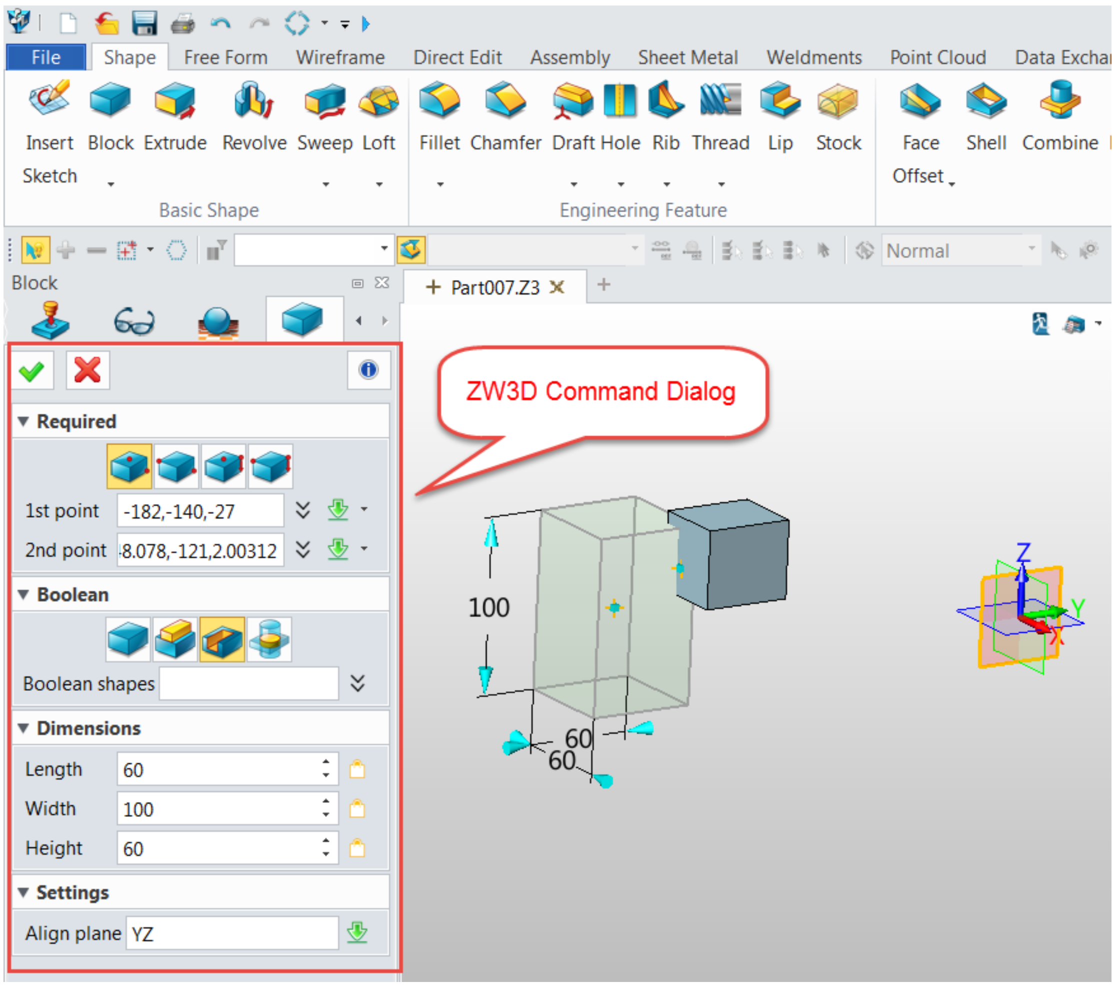
1. What is ZW3D UI Designer?

   ZW3D UI Designer is a UI designer based on QT 4.8.4. ZWSOFT bought the QT license from Digia. So, we can release our own products based on QT technology. ZW3D UI Designer is not a plug-in based on QT Designer, and it is independent product of ZWSOFT. You can use it for free.

   Note: You only can use ZW3D UI Designer for UI designer. All the coding logic, you cannot use QT technology. ZW3D UI Designer does not include the QT libraries. So, you need to buy the commercial version if you need to use the QT functions.

2. What is ZW3D Command Dialog?

   ZW3D Command Dialog is the special dialog for interaction when the user runs a command in ZW3D. ZW3D UI designer support the special controls to get the necessary values from ZW3D modeling space, or get/set some special values which only can be used in ZW3D.



Required

1st point	-182,-140,-27
2nd point	8.078,-121,2.00312

Boolean

Boolean shapes	

Dimensions

Length	60
Width	100
Height	60

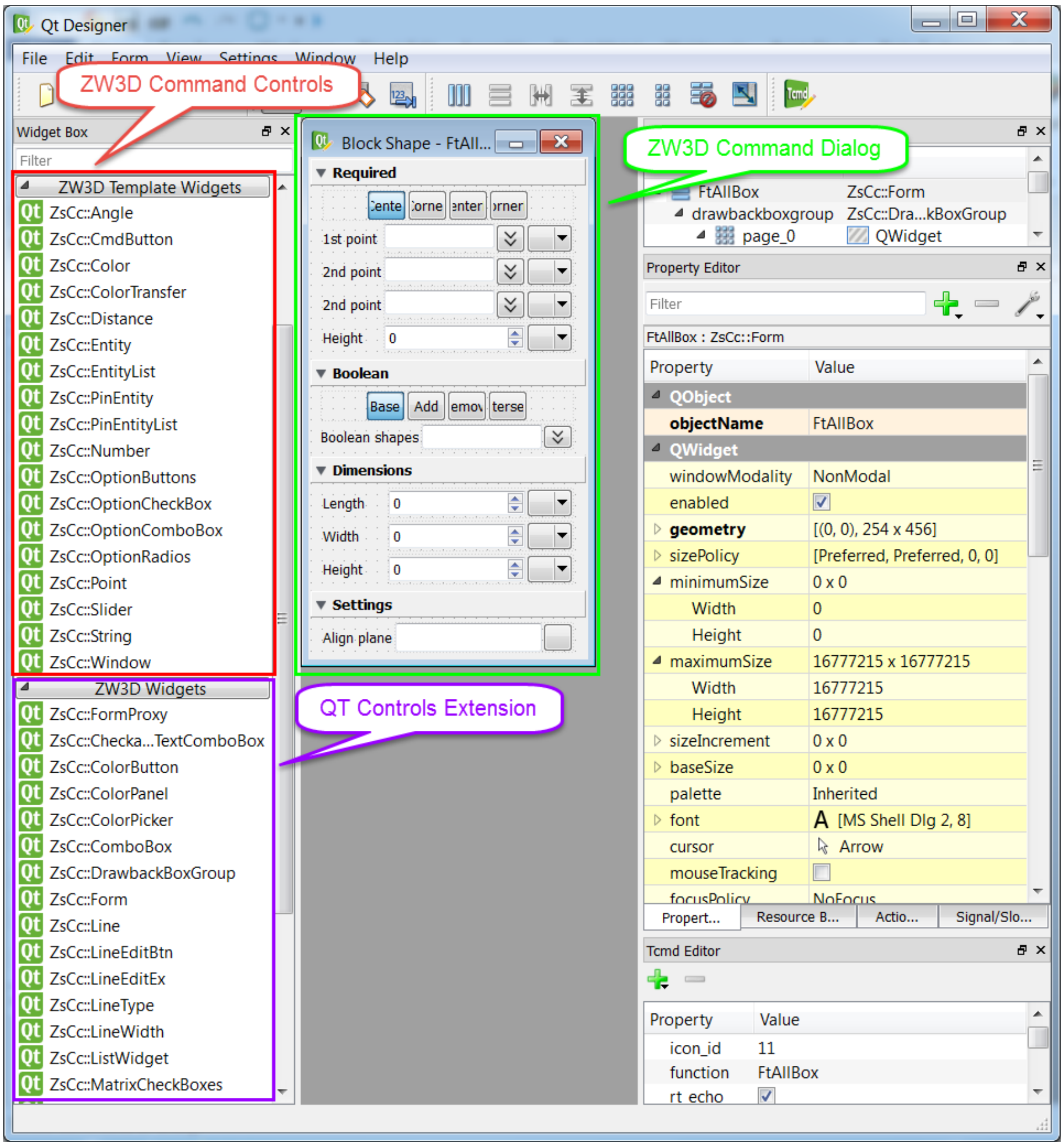
Settings

Align plane	YZ

3. ZW3D UI Designer introduction.

   Form the following picture, you can find there are two extension groups on the left side. The group marked in red is the controls for ZW3D Command Controls. The controls are used for ZW3D Command Dialog design.

The group marked in purple is the extension of Qt controls. They are used for Qt dialog design. We don't suggest using them if you don't have some special requirement.

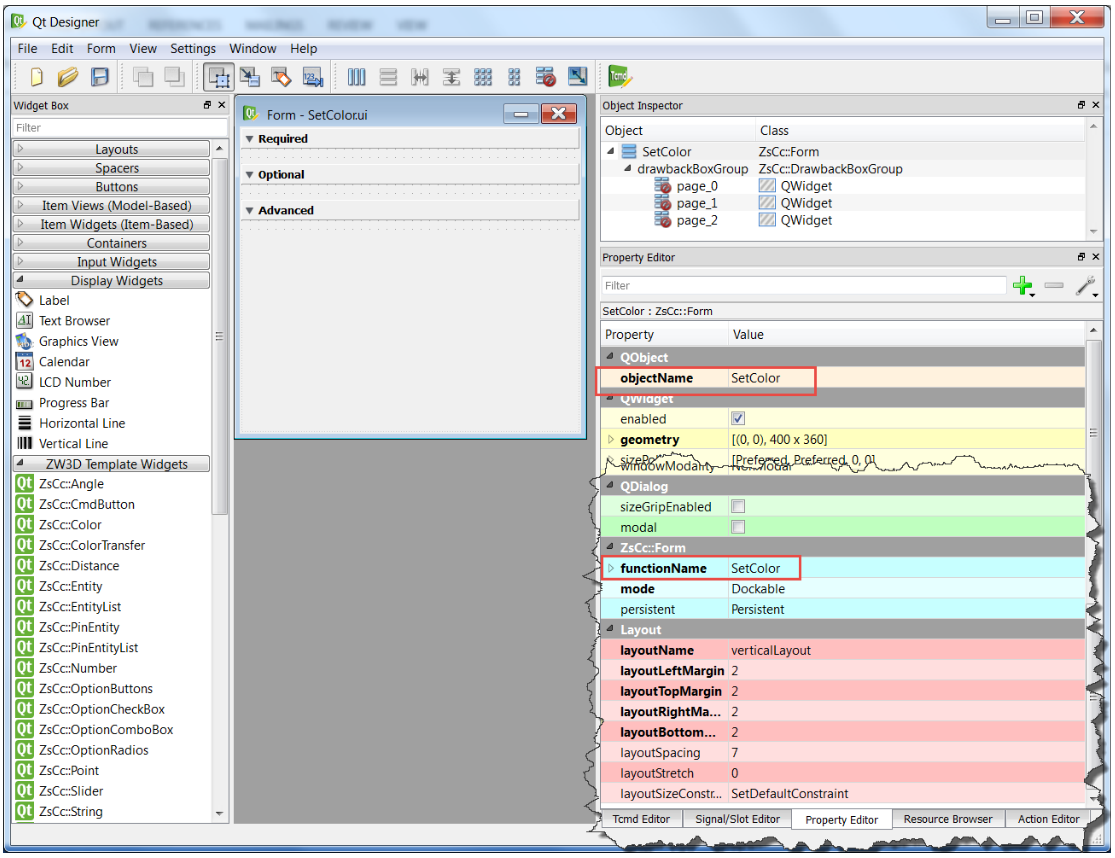


Property	Value
QObject	
objectName	FtAllBox
QWidget	
windowModality	NonModal
enabled	✓
geometry	[(0, 0), 254 x 456]
sizePolicy	[Preferred, Preferred, 0, 0]
minimumSize	0 x 0
Width	0
Height	0
maximumSize	16777215 x 16777215
Width	16777215
Height	16777215
sizeIncrement	0 x 0
baseSize	0 x 0
palette	Inherited
font	A [MS Shell Dlg 2, 8]
cursor	Arrow
mouseTracking	☐
focusPolicy	NoFocus

Property	Value
icon_id	11
function	FtAllBox
rt echo	✓

4. How to create ZW3D Command dialog?

    a) Open ZW3D UI Designer. And create a new Form based on ZsCc::Form. You can get the basic form as the picture below. Then, set the object Name and function Name to **SetColor**.

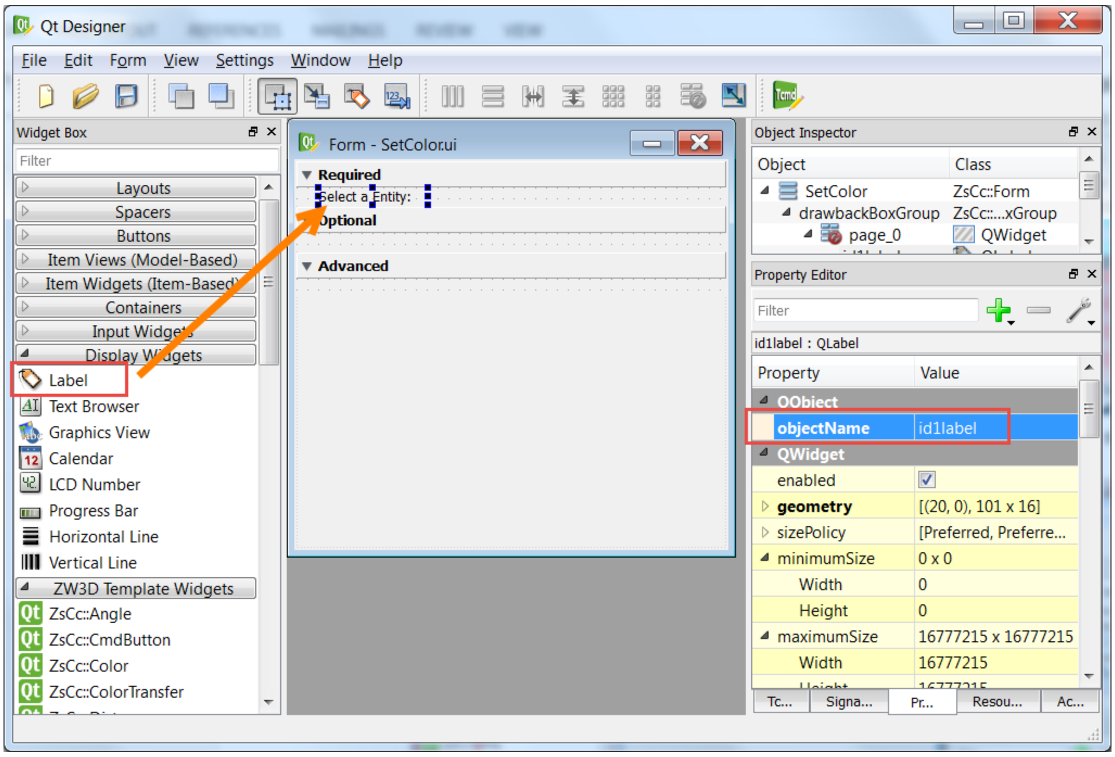


Object	Class
SetColor	ZsCc::Form
drawbackBoxGroup	ZsCc::DrawbackBoxGroup
page_0	QWidget
page_1	QWidget
page_2	QWidget

SetColor : ZsCc::Form

Property	Value
QObject	
objectName	SetColor
QWidget	
enabled	✓
geometry	[(0, 0), 400 x 360]
sizePolicy	[Preferred, Preferred, 0, 0]
windowModality	NonModal
QDialog	
sizeGripEnabled	
modal	
ZsCc::Form	
functionName	SetColor
mode	Dockable
persistent	Persistent
Layout	
layoutName	verticalLayout
layoutLeftMargin	2
layoutTopMargin	2
layoutRightMa...	2
layoutBottom...	2
layoutSpacing	7
layoutStretch	0
layoutSizeConstr...	SetDefaultConstraint

b) Drag the Label to the Required area, change the text to “Select an Entity:”. Set the **object Name** to **id1label**.

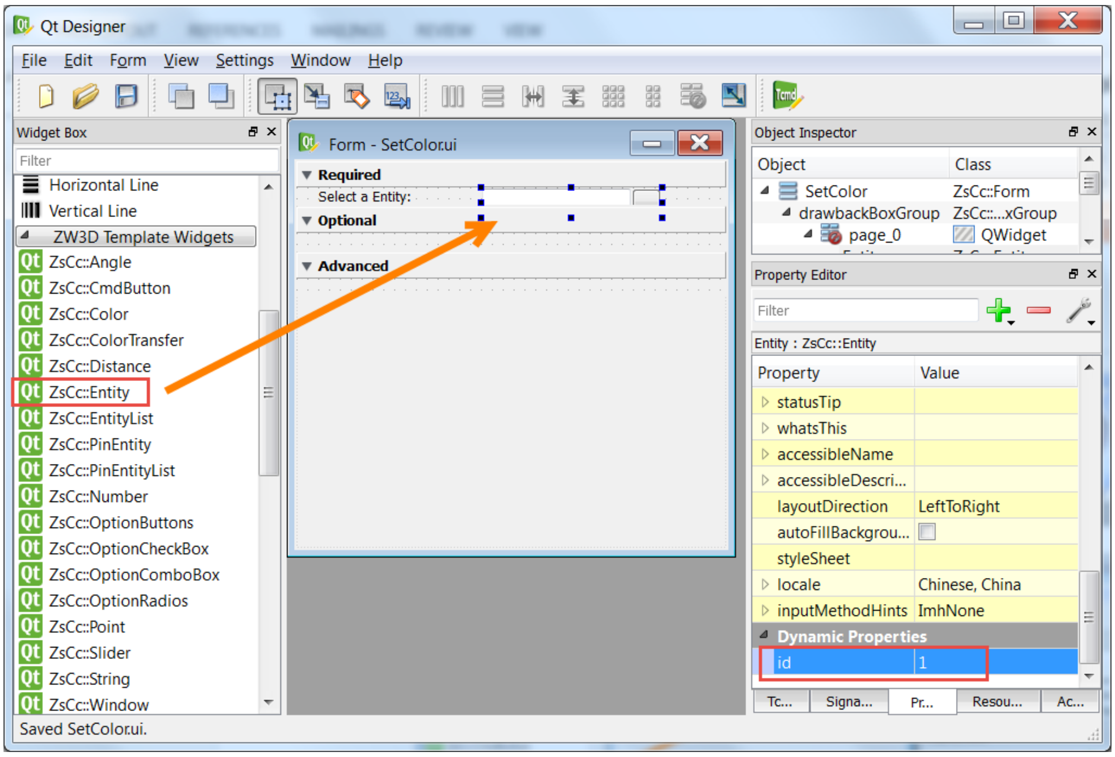


Object	Class
SetColor	ZsCc::Form
drawbackBoxGroup	ZsCc:...xGroup
page_0	QWidget

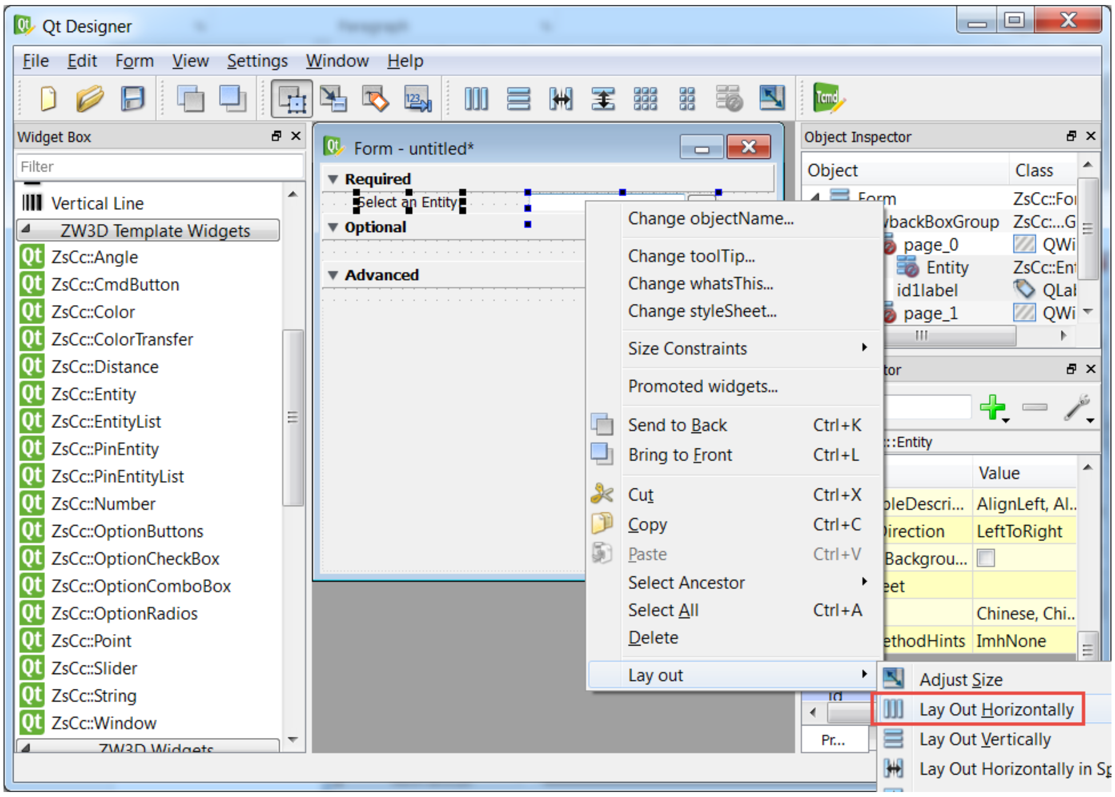
id1label : QLabel

Property	Value
QObject	
objectName	id1label
QWidget	
enabled	✓
geometry	[(20, 0), 101 x 16]
sizePolicy	[Preferred, Preferre...
minimumSize	0 x 0
Width	0
Height	0
maximumSize	16777215 x 16777215
Width	16777215

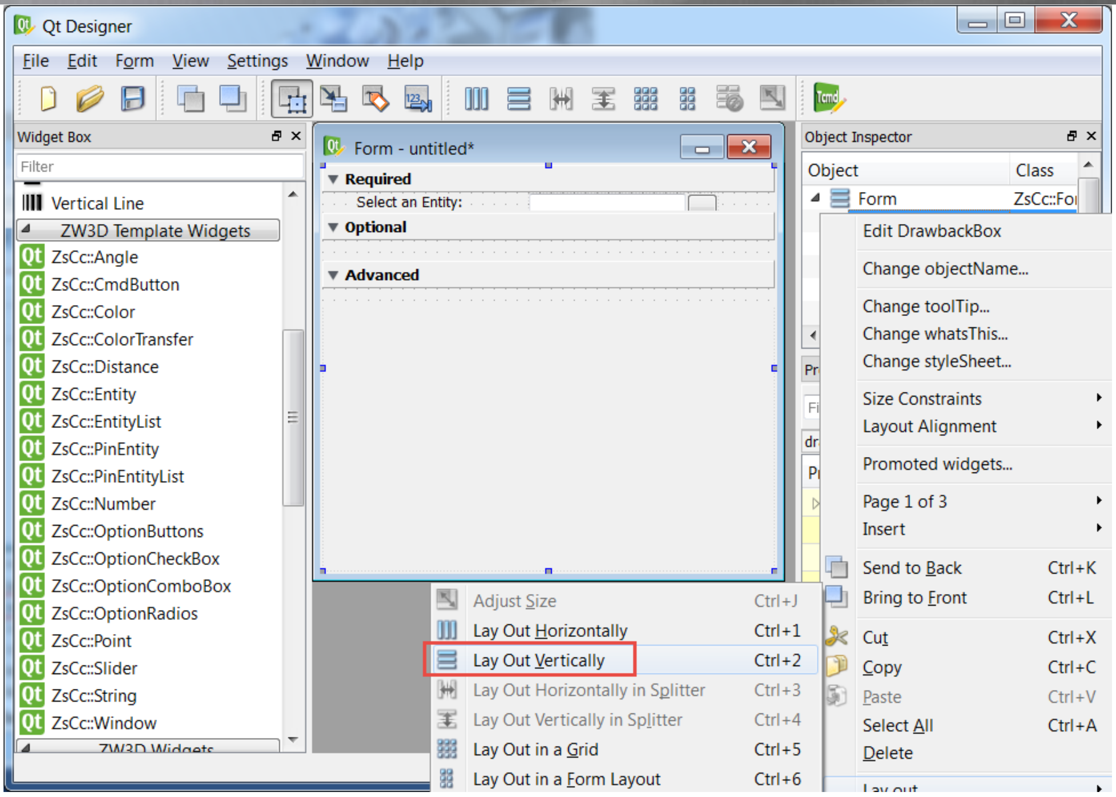
c) Drag the Entity control to the required area. Set the ID to 1. The number of the ID can be any number, but it must be the same as the number in the Label’s object Name which means these two controls are a group.



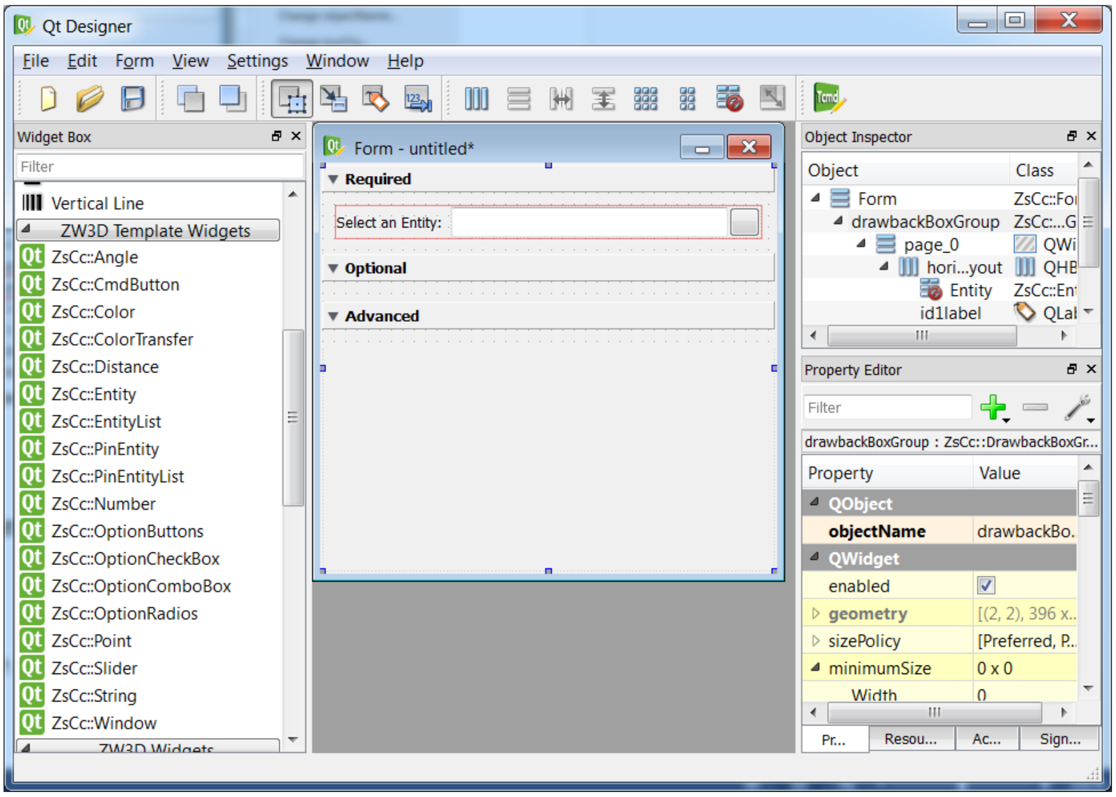
d) Use Qt functionto layout these two controls. Choose both controls and right click to choose **Lay Out Horizontally**.



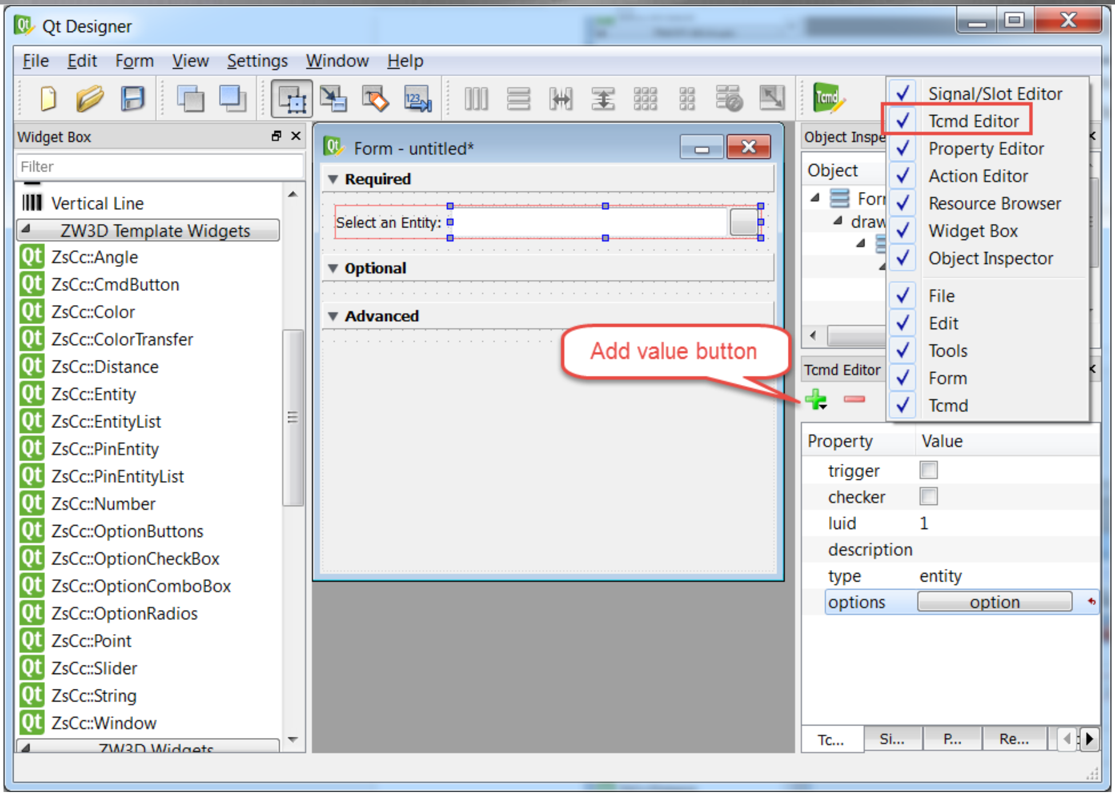
e) Right click on drawbackBoxGroup and choose **Lay Out Vertically**.



f) You will get the command dialog as follows:

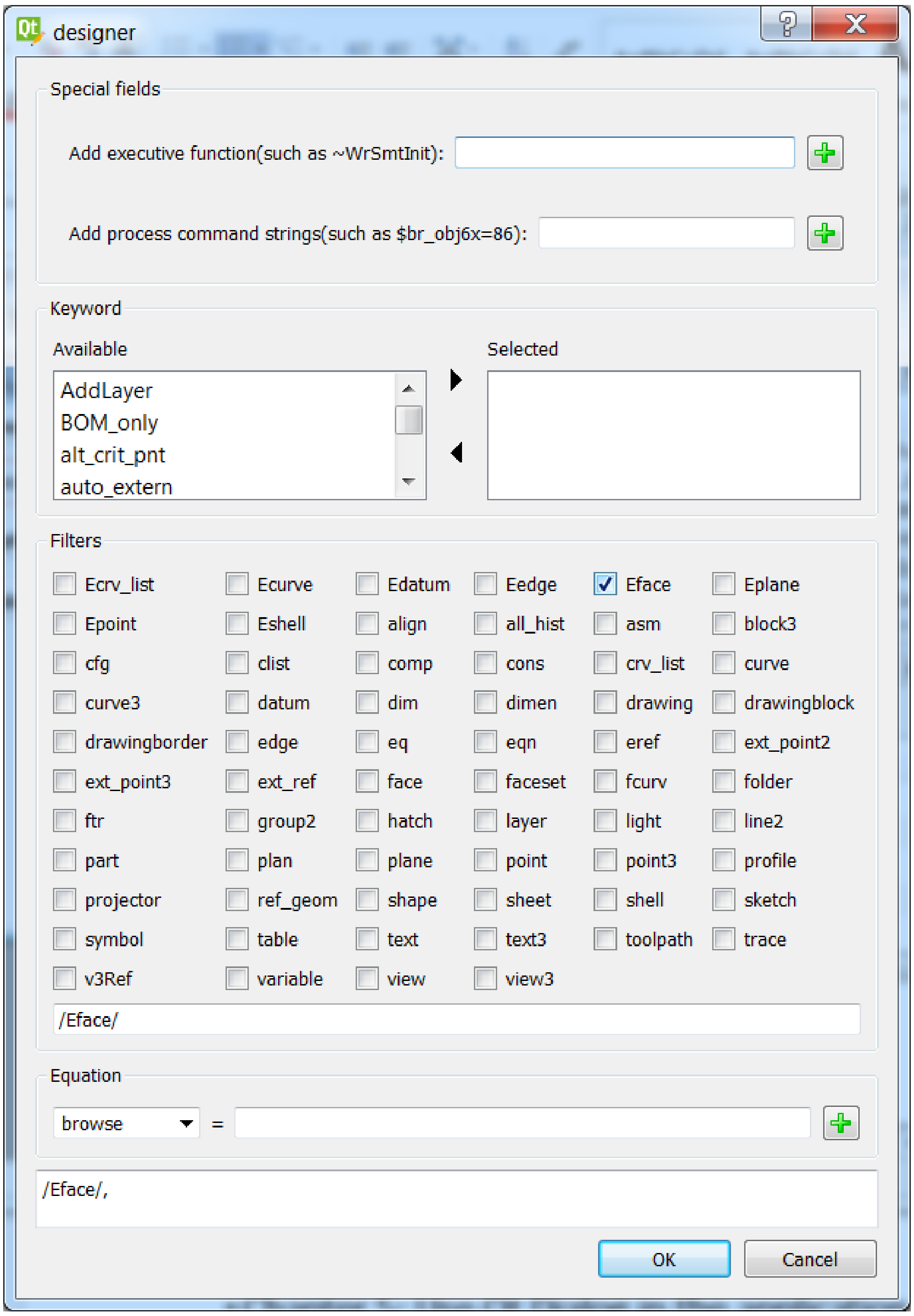


g) Right click on the space of the toolbar and show **Tcmd Editor**. Select the Entity control. You can get the Properties in the **Tcmd Editor**. Press Add value button to add **Options**.



Property	Value
trigger	☐
checker	☐
luid	1
description	
type	entity
options	option

h) Click on the **option** button, you can set the filter inside. Choose **Eface**, which means this entity control allow you to choose a face.



designer

Special fields

Add executive function(such as ~WrSmtInit):

Add process command strings(such as $br_obj6x=86):

Keyword

Available: AddLayer, BOM_only, alt_crit_pnt, auto_extern

Selected

Filters

☐ Ecrv_list	☐ Ecurve	☐ Edatum	☐ Eedge	☑ Eface	☐ Eplane
☐ Epoint	☐ Eshell	☐ align	☐ all_hist	☐ asm	☐ block3
☐ cfg	☐ clist	☐ comp	☐ cons	☐ crv_list	☐ curve
☐ curve3	☐ datum	☐ dim	☐ dimen	☐ drawing	☐ drawingblock
☐ drawingborder	☐ edge	☐ eq	☐ eqn	☐ eref	☐ ext_point2
☐ ext_point3	☐ ext_ref	☐ face	☐ faceset	☐ fcurv	☐ folder
☐ ftr	☐ group2	☐ hatch	☐ layer	☐ light	☐ line2
☐ part	☐ plan	☐ plane	☐ point	☐ point3	☐ profile
☐ projector	☐ ref_geom	☐ shape	☐ sheet	☐ shell	☐ sketch
☐ symbol	☐ table	☐ text	☐ text3	☐ toolpath	☐ trace
☐ v3Ref	☐ variable	☐ view	☐ view3		

/Eface/

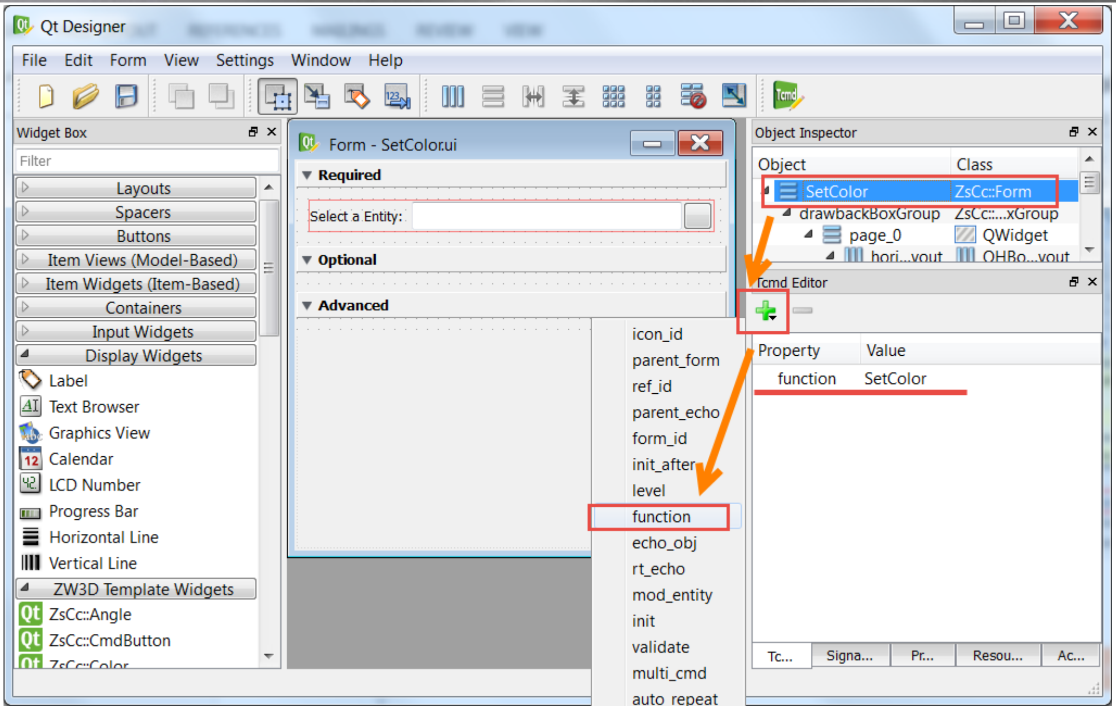
Equation

browse =

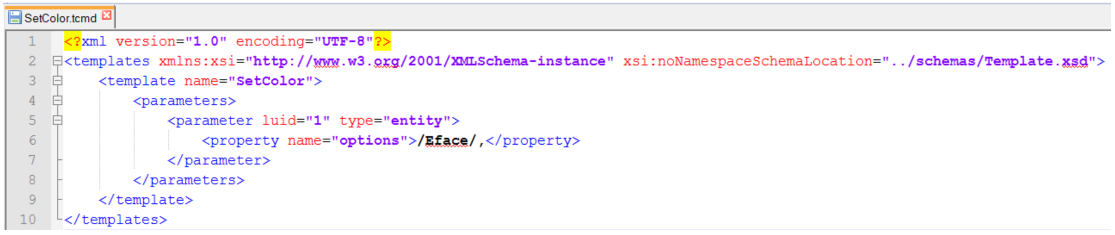
/Eface/,

OK Cancel

i) Select the form item, and add a customized property, **function**. Set the value as **SetColor**.
Note: this property sets the function when the user runs this command dialog.

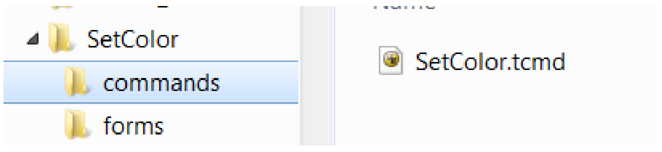


j) Save this command dialog to **SetColor** folder, and name as**SetColor.ui**. You can find ZW3D UI designer create **SetColor**.tcmd automatically. If you open the SetColor.tcmd, you can get the code as follows:


SetColor.tcmd

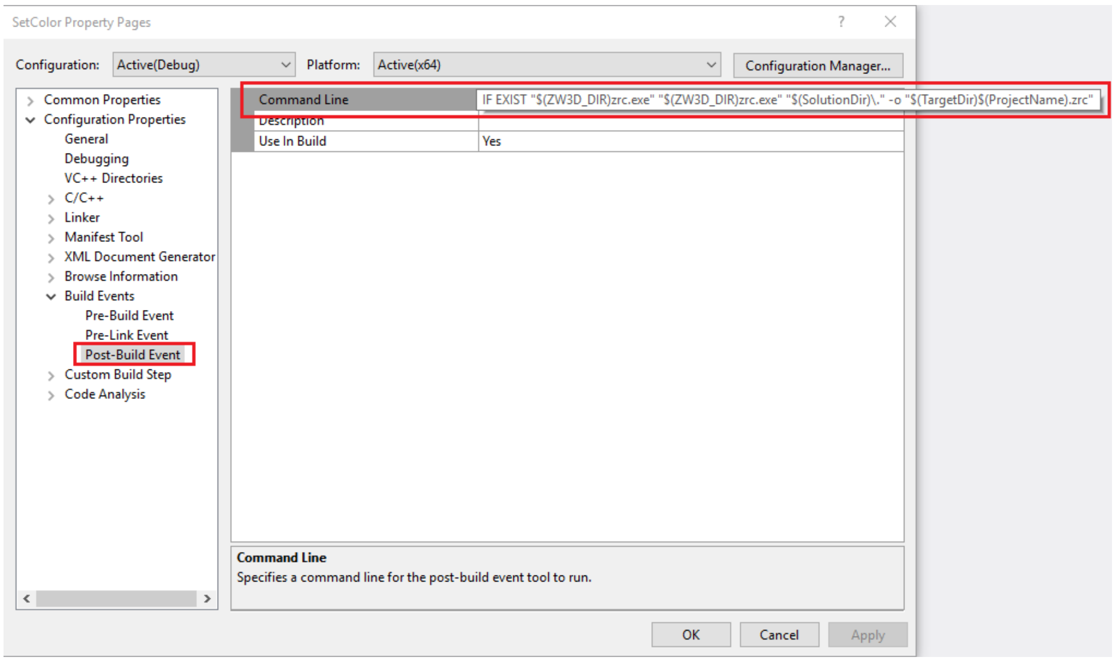
```
<?xml version="1.0" encoding="UTF-8"?>
<templates xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="../schemas/Template.xsd">
    <template name="SetColor">
        <parameters>
            <parameter luid="1" type="entity">
                <property name="options">/Eface/,</property>
            </parameter>
        </parameters>
    </template>
</templates>
```

k) Put **SetColor.ui** to the **forms**folder and put **SetColor.tcmd** to the **commands** folder. You can refer to the picture below:



l) To compile resource,your project need add command line to **Build Events→Post-Build Event→Command Line**.

The contents as shown below:
IF EXIST "$(ZW3D_DIR)zrc.exe" "$(ZW3D_DIR)zrc.exe" "$(SolutionDir)\." -o "$(TargetDir)$(ProjectName).zrc"
(Note: ZW3D_DIR is an environment variable whose value is ZW3D installation path)



Command Line	IF EXIST "$(ZW3D_DIR)zrc.exe" "$(ZW3D_DIR)zrc.exe" "$(SolutionDir)\." -o "$(TargetDir)$(ProjectName).zrc"
Description	
Use In Build	Yes

Command Line
Specifies a command line for the post-build event tool to run.

5. How to get ZW3D UI Designer?
   Download from the link:
   http://dl.zwsoft.com/zw3d/PC/ZW3D/Tech/API/ZW3D-UI-Designer.rar
   Note: You only can use ZW3D UI Designer for UI designer. All the coding logic, you cannot use QT technology. ZW3D UI Designer does not include the QT libraries. So, you need to buy the commercial version if you need to use the QT functions.

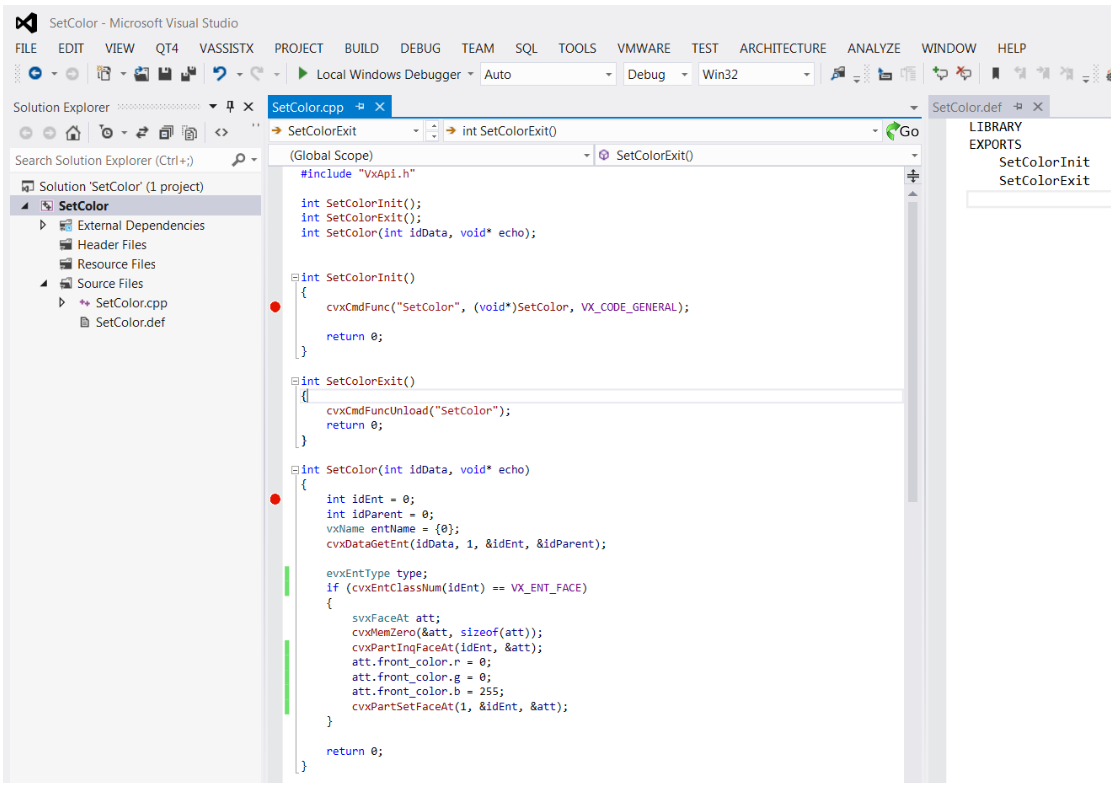
Chapter 4: Use ZW3D Command Dialog

1. Refer to Chapter 1 to create an empty project and name it as **SetColor**.
2. Add a SetColor.cpp in the project, and input the code as follows:

```
#include"VxApi.h"

intSetColor(intidData, void* echo);

intSetColorInit(int format, void *data)
{
        cvxCmdFunc("SetColor", (void*)SetColor, VX_CODE_GENERAL);
        return0;
}

intSetColorExit(void)
{
        cvxCmdFuncUnload("SetColor");
        return0;
}

intSetColor(intidData, void* echo)
{
        intidEnt = 0;
        intidParent = 0;
        vxNameentName = {0};
        cvxDataGetEnt(idData, 1, &idEnt, &idParent);
                evxEntTypetype;
                if (cvxEntClassNum(idEnt) == VX_ENT_FACE)
                {
                svxFaceAtatt;
                cvxMemZero(&att, sizeof(att));
                cvxPartInqFaceAt(idEnt, &att);
                att.front_color.r = 0;
                att.front_color.g = 0;
                att.front_color.b = 255;
                cvxPartSetFaceAt(1, &idEnt, &att);
        }
        return0;
}
```

3. Add the Module Definition file, SetColor.def. Copy the code as follows:

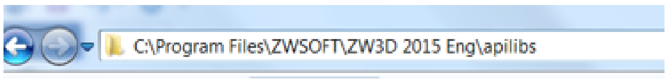
```
LIBRARY SetColor.dll
EXPORTS
        SetColorInit
        SetColorExit
        SetColor
```



```
#include "VxApi.h"

int SetColorInit();
int SetColorExit();
int SetColor(int idData, void* echo);


int SetColorInit()
{
    cvxCmdFunc("SetColor", (void*)SetColor, VX_CODE_GENERAL);

    return 0;
}

int SetColorExit()
{
    cvxCmdFuncUnload("SetColor");
    return 0;
}

int SetColor(int idData, void* echo)
{
    int idEnt = 0;
    int idParent = 0;
    vxName entName = {0};
    cvxDataGetEnt(idData, 1, &idEnt, &idParent);

    evxEntType type;
    if (cvxEntClassNum(idEnt) == VX_ENT_FACE)
    {
        svxFaceAt att;
        cvxMemZero(&att, sizeof(att));
        cvxPartInqFaceAt(idEnt, &att);
        att.front_color.r = 0;
        att.front_color.g = 0;
        att.front_color.b = 255;
        cvxPartSetFaceAt(1, &idEnt, &att);
    }

    return 0;
}
```

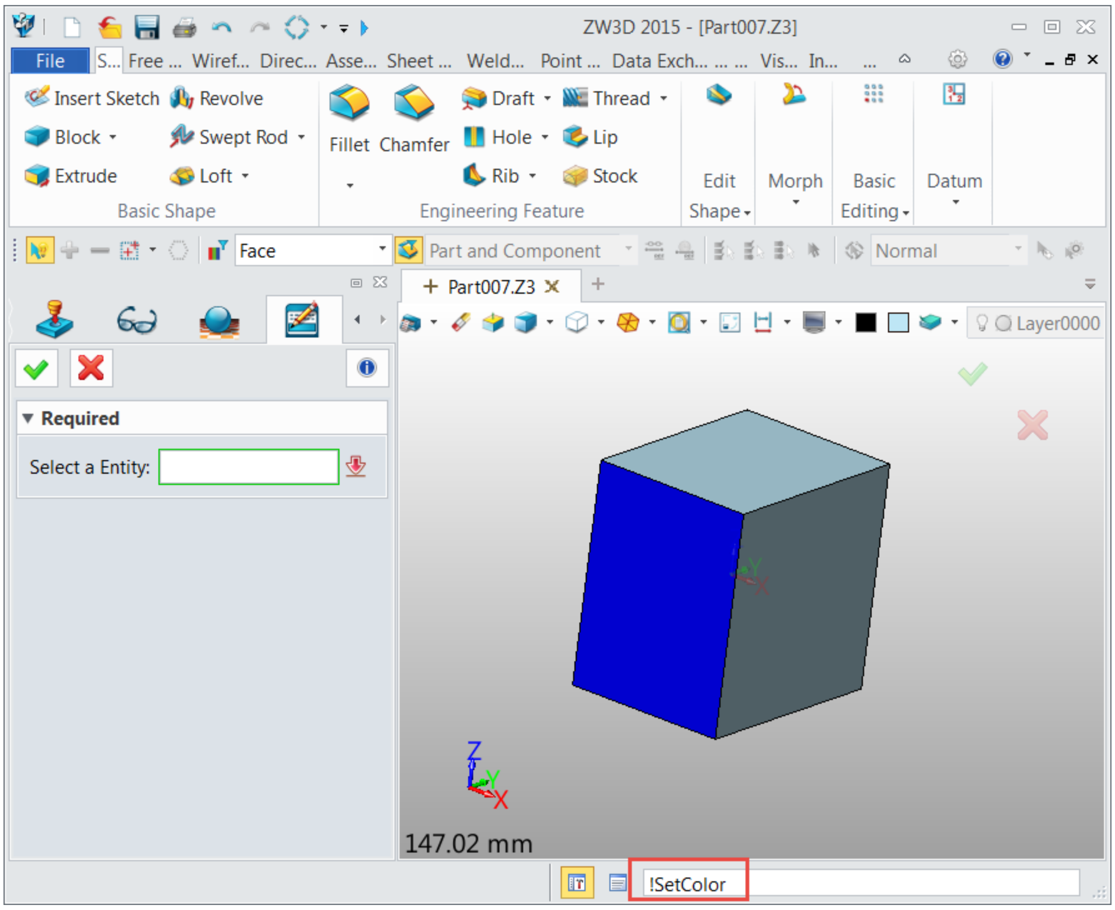
```
LIBRARY
EXPORTS
    SetColorInit
    SetColorExit
```

4. Refer to Chapter 1 to set the configuration of the project and build to get **SetColor.dll** and **SetColor.zrc**. Now, we have the dll and zrc. Copy all the file to ZW3D installation directory. Note: you must close ZW3D before you copy the files.

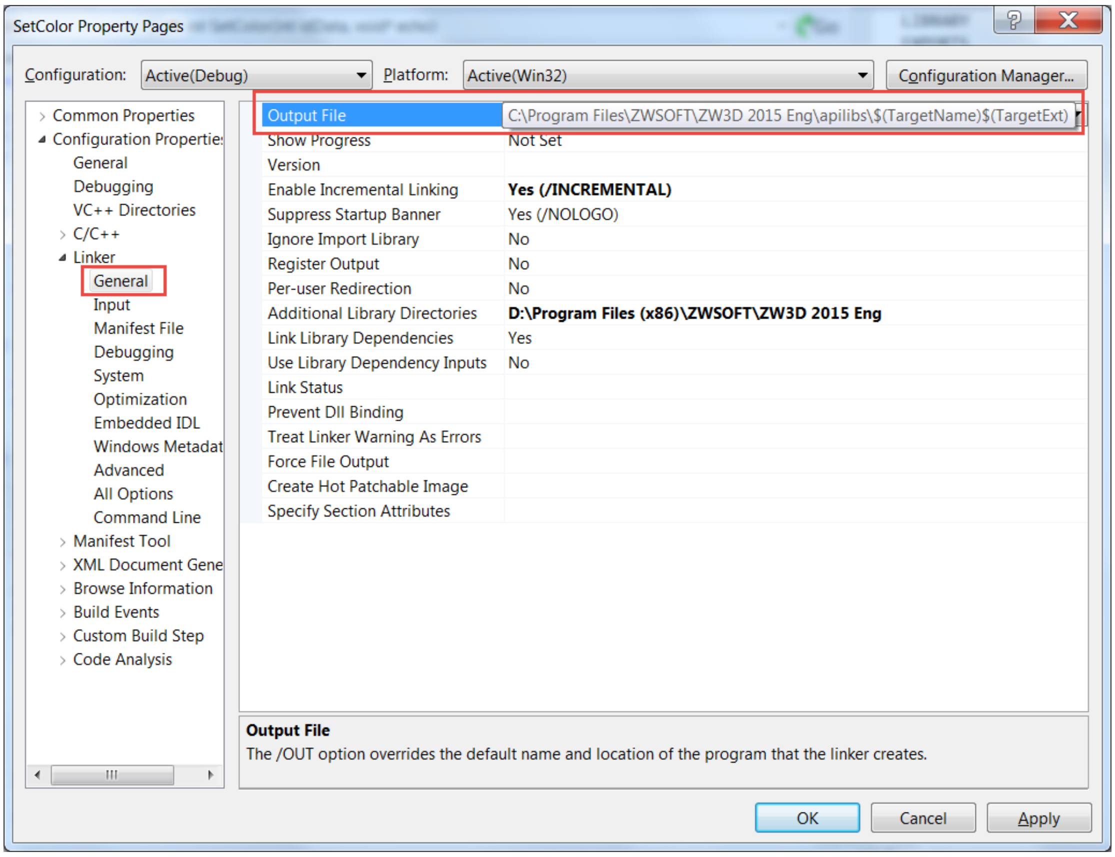


C:\Program Files\ZWSOFT\ZW3D 2015 Eng\apilibs

5. Start ZW3D, create a new part and draw a box. Then, input "!SetColor" in the command line. You can get the command dialog as follows. Select a surface and press OK, you can find the color of the surface was changed to blue.



6. Debug this project. Set the Output File to “C:\Program Files\ZWSOFT\ZW3D 2015 Eng\apilibs\$(TargetName)$(TargetExt)”. You can refer to the picture as follows. Then, rebuild the project. You can set the break point and debug this project.
Note: ZW3D will load all the DLL in the “apilibs” directory automatically.

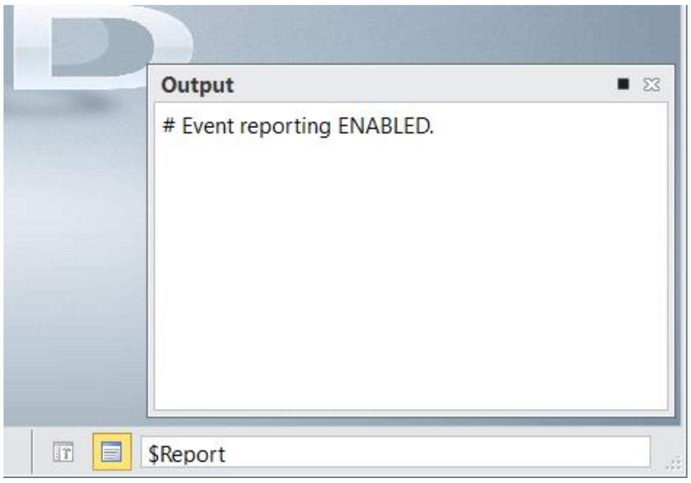


Output File	C:\Program Files\ZWSOFT\ZW3D 2015 Eng\apilibs\$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	**D:\Program Files (x86)\ZWSOFT\ZW3D 2015 Eng**
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Output File
The /OUT option overrides the default name and location of the program that the linker creates.

Chapter 5: How to call ZW3D functions?

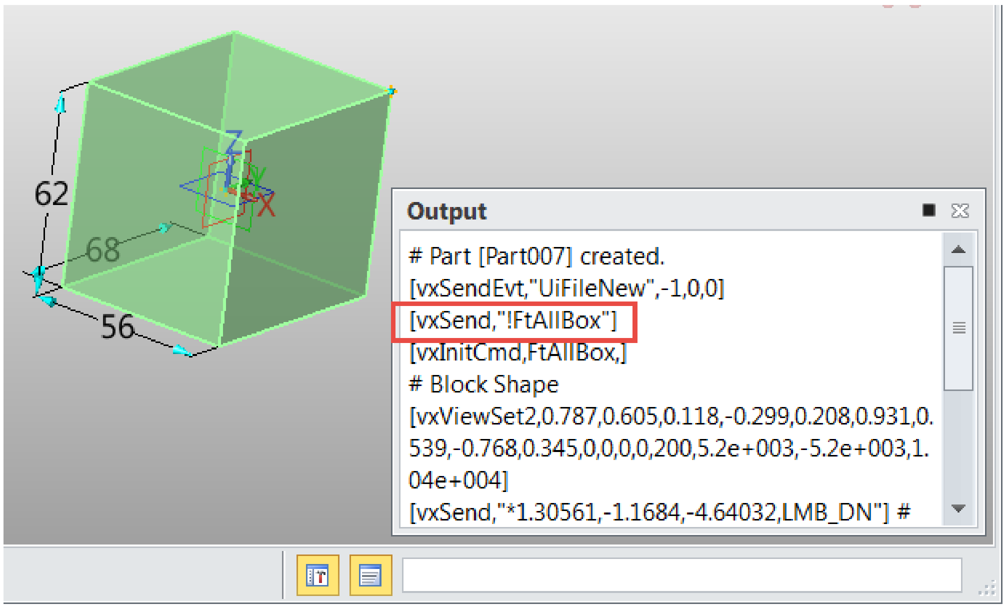
1. Get ZW3D command dialogs.
   Download all ZW3D command dialogs from the following link:
   https://dl.zwsoft.com/zw3d/Products/ZW3D_API/2023/zw3d-command-dialog_2023.rar
2. Study the command dialog of ZW3D.
   a) Run “$Report” in the command line of ZW3D.
      You will get the following message after you run the command successfully.



```
# Event reporting ENABLED.
```

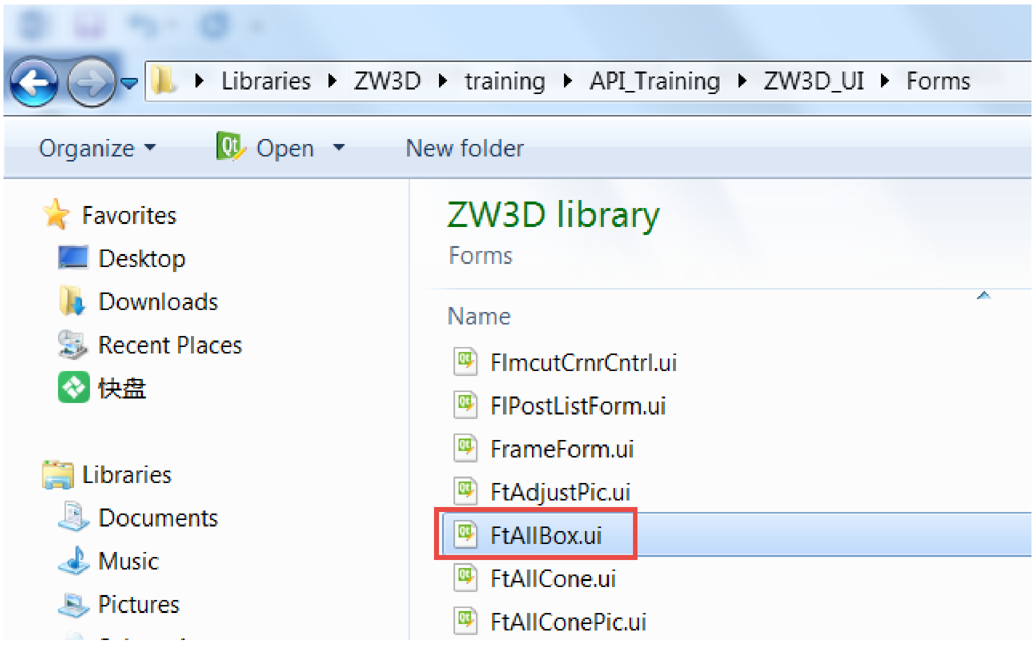
```
$Report
```

   b) Run the command you need to know, for example the function to create Box.
      You will get the message as below, “[vxSend,”!FtAllBox]”, which means ZW3D run “!FtAllBox” to create the box.

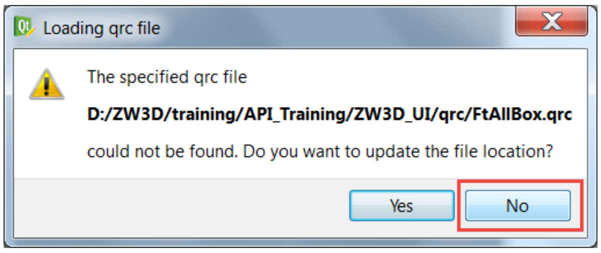


```
# Part [Part007] created.
[vxSendEvt,"UiFileNew",-1,0,0]
[vxSend,"!FtAllBox"]
[vxInitCmd,FtAllBox,]
# Block Shape
[vxViewSet2,0.787,0.605,0.118,-0.299,0.208,0.931,0.
539,-0.768,0.345,0,0,0,0,200,5.2e+003,-5.2e+003,1.
04e+004]
[vxSend,"*1.30561,-1.1684,-4.64032,LMB_DN"] #
```

   c) Go to ZW3D Command dialog folder, you can find a command dialog named as “FtAllBox.ui”.



d) Use ZW3D UI designer to open this dialog, you will get a warning. Press **No** to ignore it.

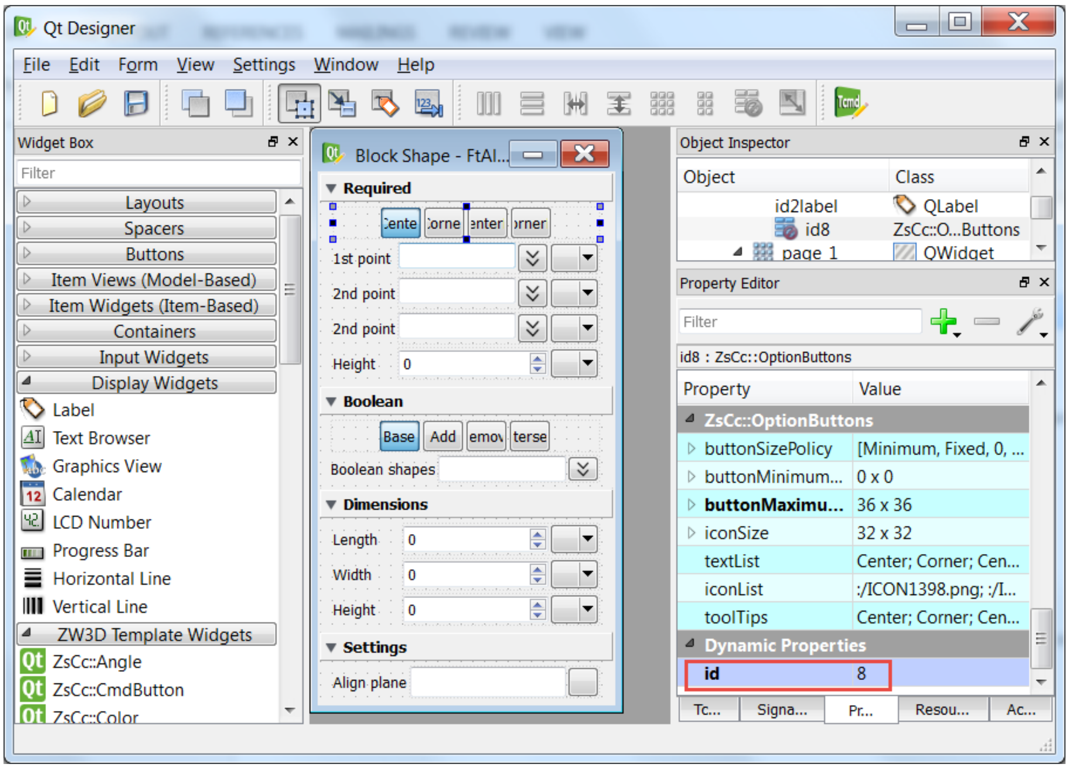

Loading qrc file

The specified qrc file

D:/ZW3D/training/API_Training/ZW3D_UI/qrc/FtAllBox.qrc

could not be found. Do you want to update the file location?

e) You will know all the IDs of the controls.

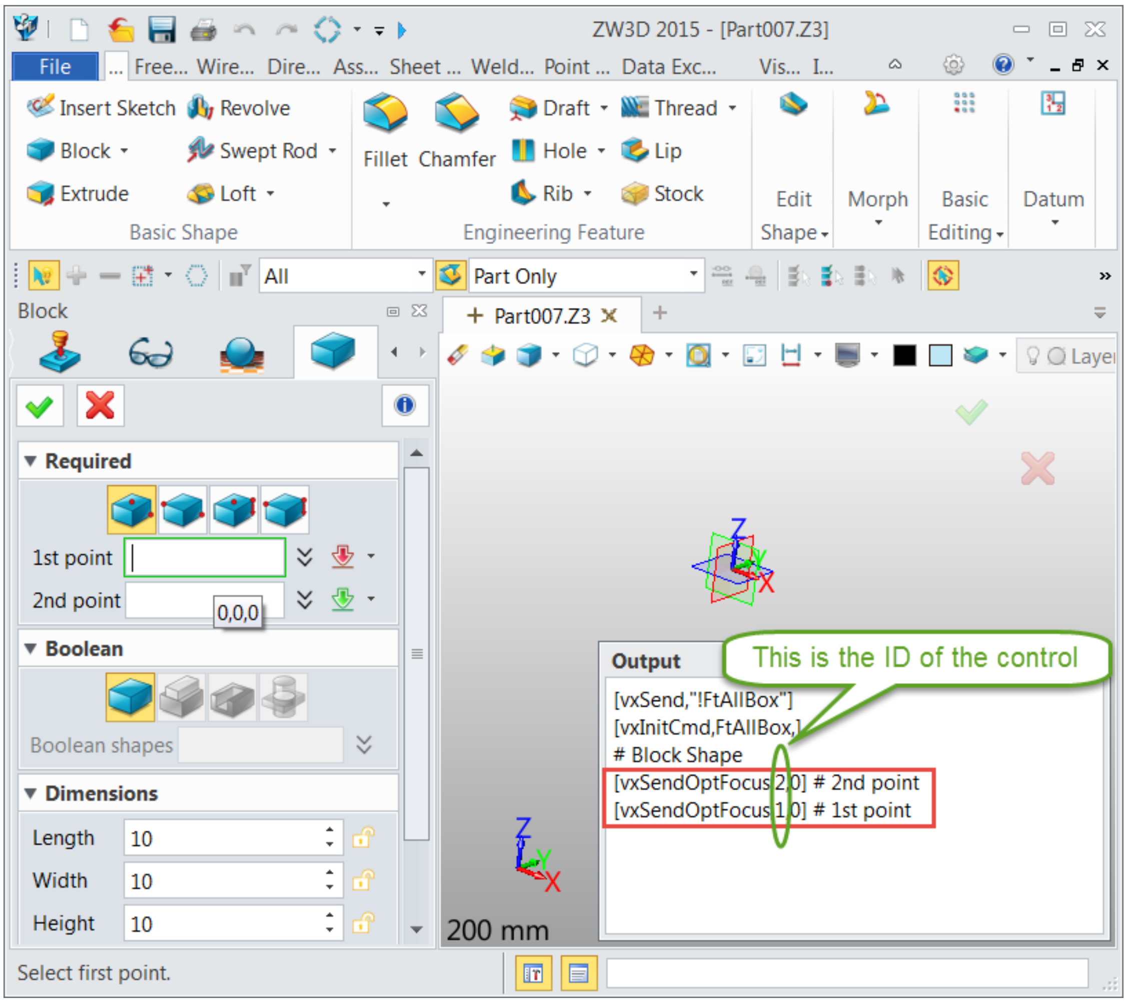


Object	Class
id2label	QLabel
id8	ZsCc::O...Buttons
page 1	QWidget

id8 : ZsCc::OptionButtons

Property	Value
ZsCc::OptionButtons	
buttonSizePolicy	[Minimum, Fixed, 0, ...
buttonMinimum...	0 x 0
buttonMaximu...	36 x 36
iconSize	32 x 32
textList	Center; Corner; Cen...
iconList	:/ICON1398.png; :/I...
toolTips	Center; Corner; Cen...
Dynamic Properties	
id	8

f) You will find there are two “2nd point” controls. To learn the difference, refer to step b) to know the function. Then, change your mouse on different controls in the command dialog. You can get different message if you change the controls. You can find the ID in the messages.



```
Output
[vxSend,"!FtAllBox"]
[vxInitCmd,FtAllBox,]
# Block Shape
[vxSendOptFocus,2,0] # 2nd point
[vxSendOptFocus,1,0] # 1st point
```

3. Now, you already know the function name and the control IDs. Let’s use ZW3D function to create a box although ZW3D also support the API function to create a box [cvxPartBox()].

a) Refer to chapter 1 to create a new project and named as “myBox”.

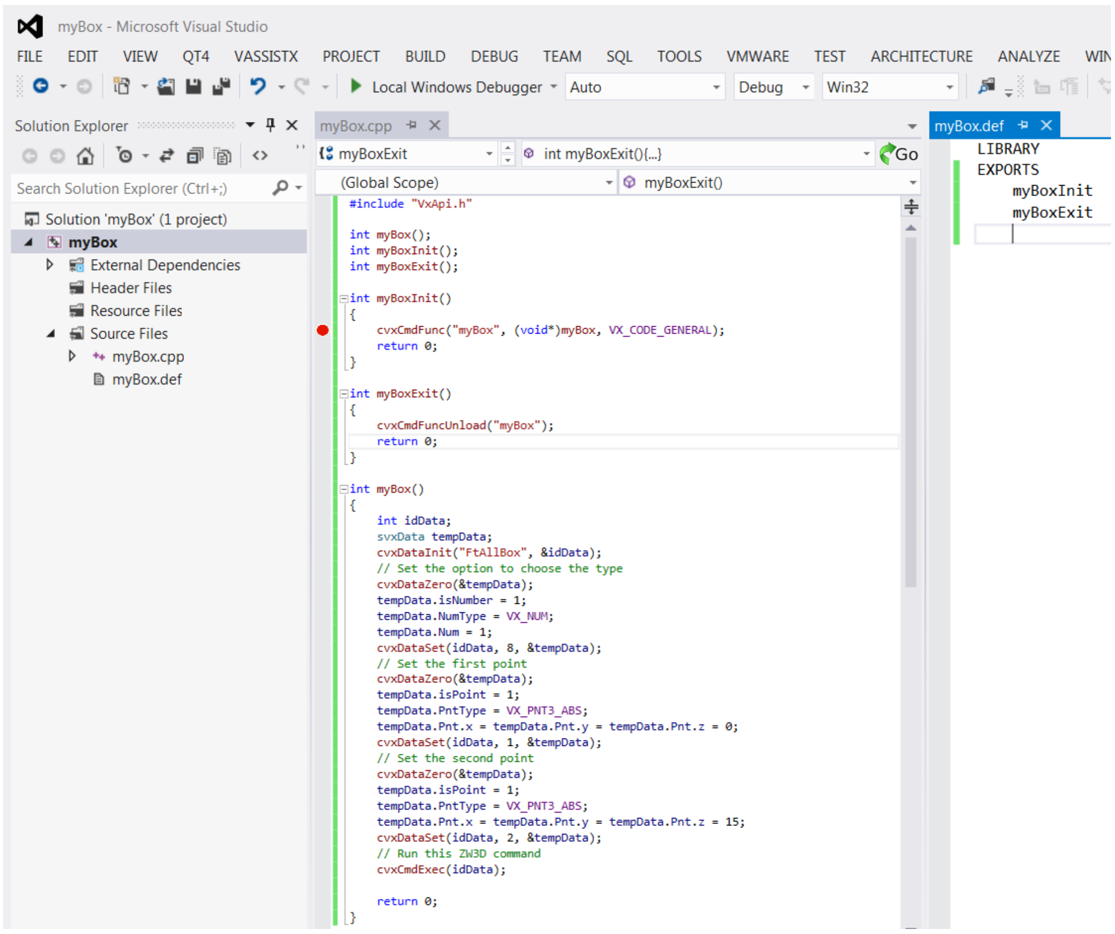
b) Add myBox.cpp, and copy the code as follows:

```
#include"VxApi.h"

intmyBox();

intmyBoxInit(int format, void *data)
{
	cvxCmdFunc("myBox", (void*)myBox, VX_CODE_GENERAL);
	return0;
}
```

```
intmyBoxExit(void)
{
        cvxCmdFuncUnload("myBox");
        return0;
}

intmyBox()
{
        intidData;
        svxDatatempData;
        cvxDataInit("FtAllBox", &idData);
        // Set the option to choose the type
        cvxDataZero(&tempData);
        tempData.isNumber = 1;
        tempData.NumType = VX_NUM;
        tempData.Num = 1;
        cvxDataSet(idData, 8, &tempData);
        // Set the first point
        cvxDataZero(&tempData);
        tempData.isPoint = 1;
        tempData.PntType = VX_PNT3_ABS;
        tempData.Pnt.x = tempData.Pnt.y = tempData.Pnt.z = 0;
        cvxDataSet(idData, 1, &tempData);
        // Set the second point
        cvxDataZero(&tempData);
        tempData.isPoint = 1;
        tempData.PntType = VX_PNT3_ABS;
        tempData.Pnt.x = tempData.Pnt.y = tempData.Pnt.z = 15;
        cvxDataSet(idData, 2, &tempData);
        // Run this ZW3D command
        cvxCmdExec(idData);

        return0;
}
```

c) Add the Module Definition file, myBox.def. Copy the code as follows:

```
LIBRARY myBox.dll
EXPORTS
    myBoxInit
    myBoxExit
    myBox
```



```
#include "VxApi.h"

int myBox();
int myBoxInit();
int myBoxExit();

int myBoxInit()
{
    cvxCmdFunc("myBox", (void*)myBox, VX_CODE_GENERAL);
    return 0;
}

int myBoxExit()
{
    cvxCmdFuncUnload("myBox");
    return 0;
}

int myBox()
{
    int idData;
    svxData tempData;
    cvxDataInit("FtAllBox", &idData);
    // Set the option to choose the type
    cvxDataZero(&tempData);
    tempData.isNumber = 1;
    tempData.NumType = VX_NUM;
    tempData.Num = 1;
    cvxDataSet(idData, 8, &tempData);
    // Set the first point
    cvxDataZero(&tempData);
    tempData.isPoint = 1;
    tempData.PntType = VX_PNT3_ABS;
    tempData.Pnt.x = tempData.Pnt.y = tempData.Pnt.z = 0;
    cvxDataSet(idData, 1, &tempData);
    // Set the second point
    cvxDataZero(&tempData);
    tempData.isPoint = 1;
    tempData.PntType = VX_PNT3_ABS;
    tempData.Pnt.x = tempData.Pnt.y = tempData.Pnt.z = 15;
    cvxDataSet(idData, 2, &tempData);
    // Run this ZW3D command
    cvxCmdExec(idData);

    return 0;
}
```

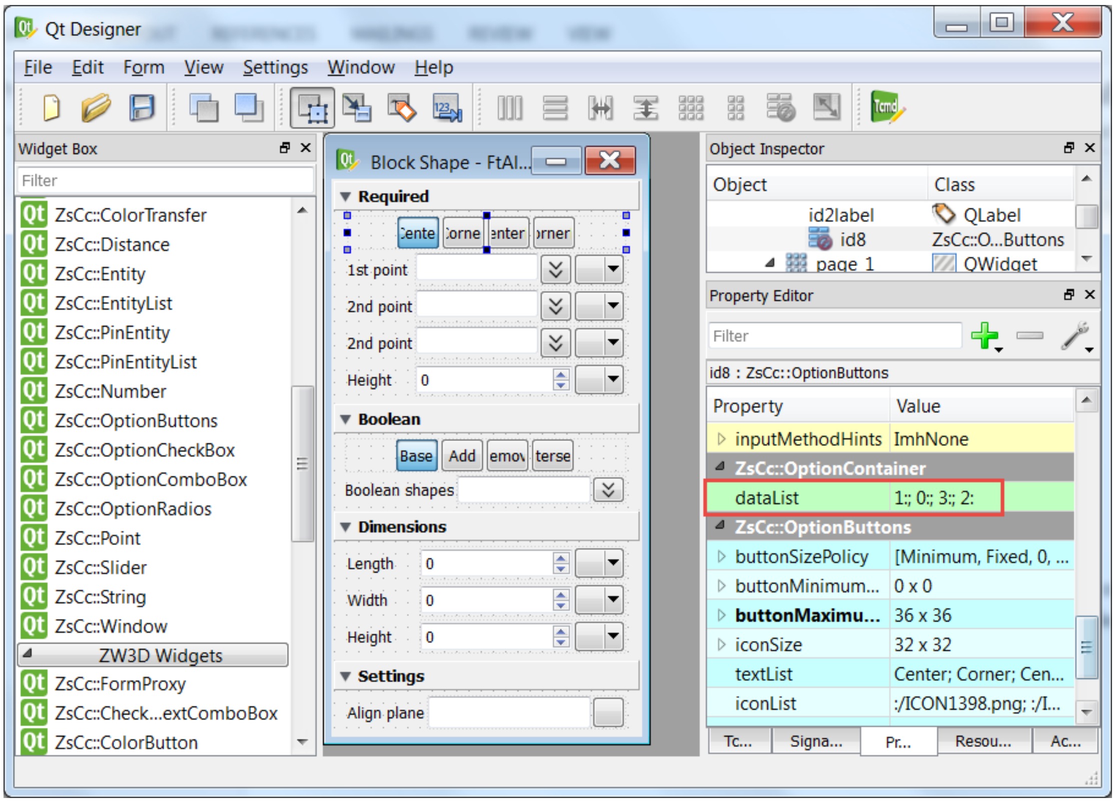
```
LIBRARY
EXPORTS
    myBoxInit
    myBoxExit
```

Build this project and load the DLL in ZW3D. Run “~myBox” in the command line after you create a new part. This command will create a box in the modeling space.

4. The value of the ZsCc::optionbuttons.

   After clicking on the control, you can find the **DataList** property, where the value order of the control should be 1, 0, 3, 2. So you need to set the right value in your code. The following code means this project use the first button. If you want to use the second button, you need to set the value to 0.

```
// Set the option to choose the type
    cvxDataZero(&tempData);
    tempData.isNumber = 1;
    tempData.NumType = VX_NUM;
    tempData.Num = 1;
    cvxDataSet(idData, 8, &tempData);
```



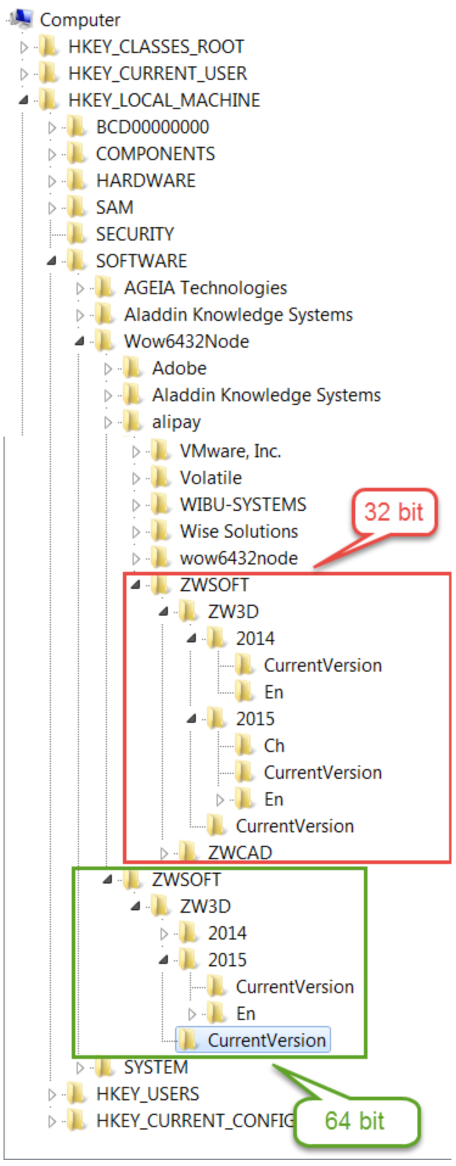
Object	Class
id2label	QLabel
id8	ZsCc::O...Buttons
page_1	QWidget

id8 : ZsCc::OptionButtons

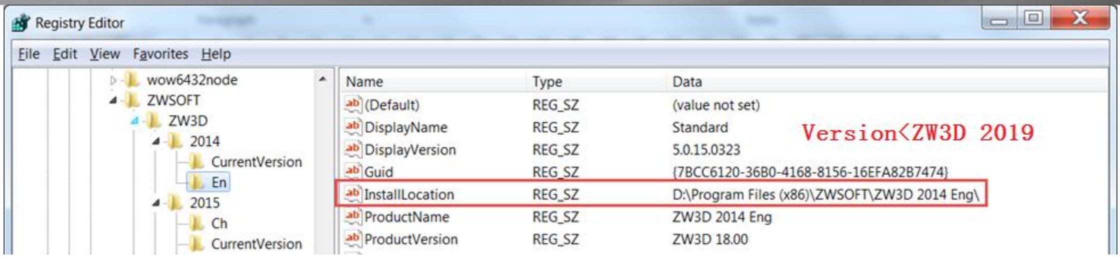
Property	Value
inputMethodHints	ImhNone
ZsCc::OptionContainer	
dataList	1:; 0:; 3:; 2:
ZsCc::OptionButtons	
buttonSizePolicy	[Minimum, Fixed, 0, ...
buttonMinimum...	0 x 0
buttonMaximu...	36 x 36
iconSize	32 x 32
textList	Center; Corner; Cen...
iconList	:/ICON1398.png; :/I...

Chapter 6: ZW3D Register information

Theregister of ZW3D is saved in:
64 bit: [HKEY_LOCAL_MACHINE\SOFTWARE\ZWSOFT\ZW3D]
32 bit: [HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\ZWSOFT\ZW3D]
The **CurrentVersion** in the root directory remembers the version run last time. The **CurrentVersion** in the special version directory remembers the language run last time.
You can also get the detailed information in each language directory, like installation path.



```
Computer
  HKEY_CLASSES_ROOT
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
    BCD00000000
    COMPONENTS
    HARDWARE
    SAM
    SECURITY
    SOFTWARE
      AGEIA Technologies
      Aladdin Knowledge Systems
      Wow6432Node
        Adobe
        Aladdin Knowledge Systems
        alipay
          VMware, Inc.
          Volatile
          WIBU-SYSTEMS
          Wise Solutions
          wow6432node
          ZWSOFT
            ZW3D
              2014
                CurrentVersion
                En
              2015
                Ch
                CurrentVersion
                En
              CurrentVersion
            ZWCAD
        ZWSOFT
          ZW3D
            2014
            2015
              CurrentVersion
              En
            CurrentVersion
      SYSTEM
    HKEY_USERS
    HKEY_CURRENT_CONFIG
```



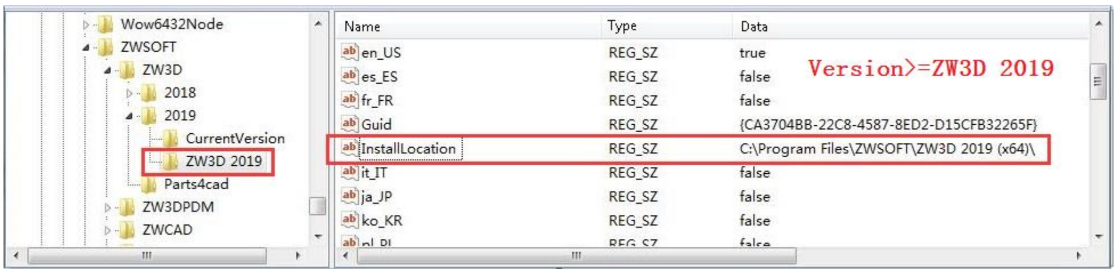
Registry Editor

File Edit View Favorites Help

- wow6432node
  - ZWSOFT
    - ZW3D
      - 2014
        - CurrentVersion
        - En
      - 2015
        - Ch
        - CurrentVersion

Name	Type	Data
(Default)	REG_SZ	(value not set)
DisplayName	REG_SZ	Standard
DisplayVersion	REG_SZ	5.0.15.0323
Guid	REG_SZ	{7BCC6120-36B0-4168-8156-16EFA82B7474}
InstallLocation	REG_SZ	D:\Program Files (x86)\ZWSOFT\ZW3D 2014 Eng\
ProductName	REG_SZ	ZW3D 2014 Eng
ProductVersion	REG_SZ	ZW3D 18.00

Version<ZW3D 2019



- Wow6432Node
  - ZWSOFT
    - ZW3D
      - 2018
      - 2019
        - CurrentVersion
        - ZW3D 2019
      - Parts4cad
    - ZW3DPDM
    - ZWCAD

Name	Type	Data
en_US	REG_SZ	true
es_ES	REG_SZ	false
fr_FR	REG_SZ	false
Guid	REG_SZ	{CA3704BB-22C8-4587-8ED2-D15CFB32265F}
InstallLocation	REG_SZ	C:\Program Files\ZWSOFT\ZW3D 2019 (x64)\
it_IT	REG_SZ	false
ja_JP	REG_SZ	false
ko_KR	REG_SZ	false
pl_PL	REG_SZ	false

Version>=ZW3D 2019