MSVC IDE Debugger List Tracing

Table of Contents

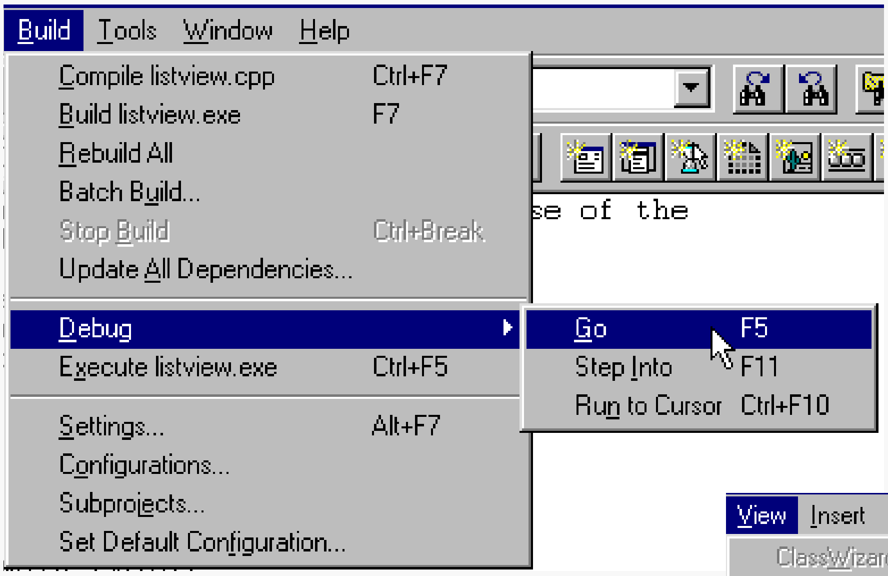
Watch Window

Using the Watch Window to View List Structures

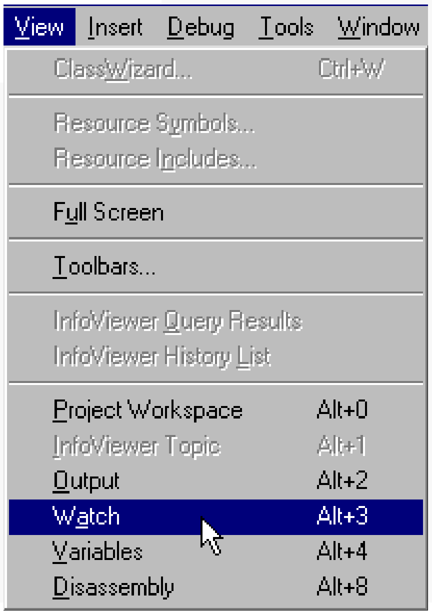
Once the program is successfully compiled and ready to execute:

– Set breakpoints at the locations(s) in the code where you wish to view the list's contents. Position the cursor on a line of code, right click, and select **Insert/Remove Breakpoint**.

– Select **Build -> Debug -> Go** from the menu or hit F5 to begin debugging.

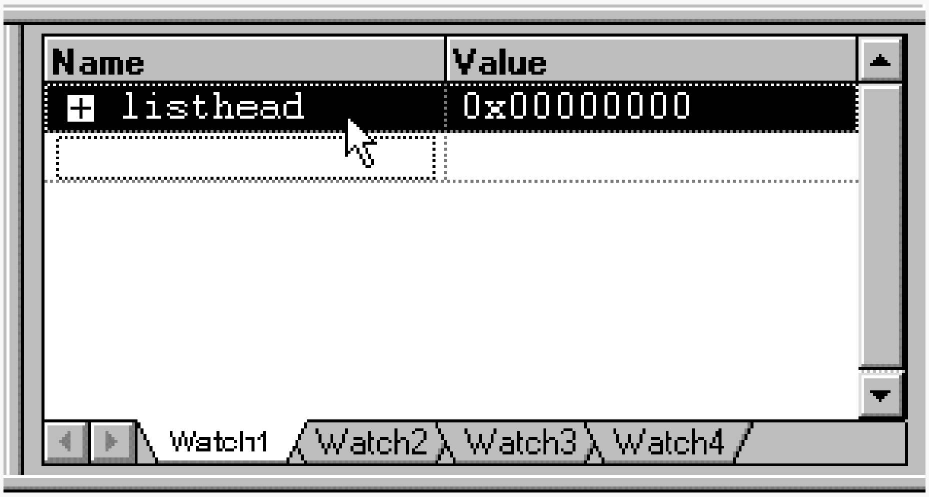


– The program will stop at the first breakpoint. If the Watch Window is not visible, select **View -> Watch** from the menu.



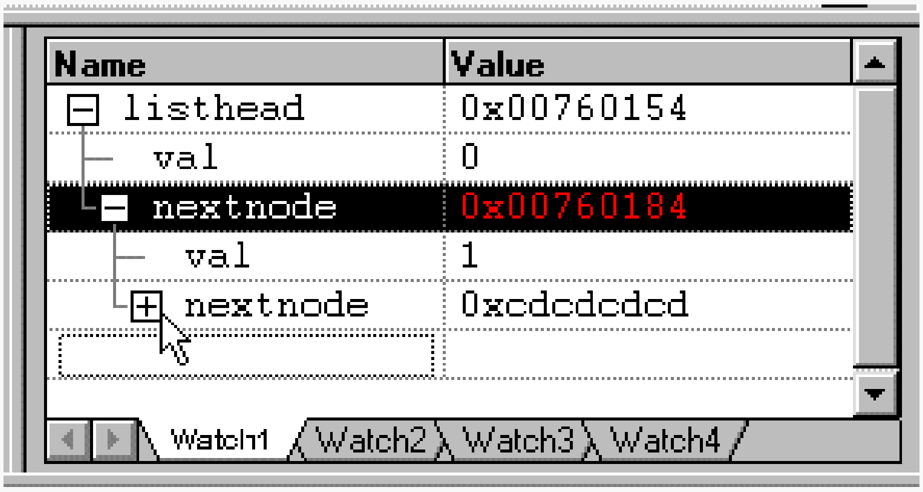
Viewing Linked List Contents (continued)

– Double-click the structure variable name in the Source window to highlight it, then drag and drop it into the Watch Window.



Name	Value
⊞ listhead	0x00000000

– There will be a plus (+) sign to the left of the variable name in the *name field* of the Watch Window. Click on the plus (+) to "**expand**" the variable. This will in effect display the "**contents**" of the structure with each subsequent expansion of the variable. A minus (-) sign indicates that the variable is already fully expanded.



Name	Value
⊟ listhead	0x00760154
val	0
⊟ nextnode	0x00760184
val	1
⊞ nextnode	0xcdcdcdcd

Watching Linked-List Variables

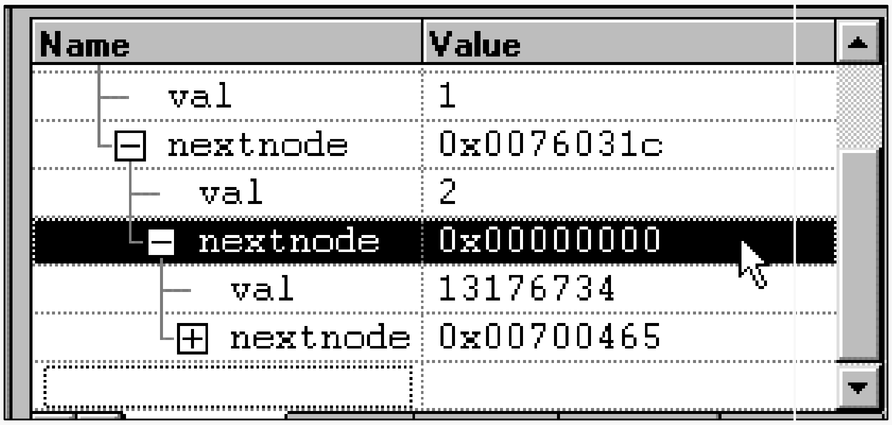
Sample Program

```
/* Sample program to demonstrate use of the
     variables window.  */

#include <iostream.h>

typedef struct node {
     int val;
     node* nextnode;
};

int i;
node* listhead, * currnode;

void main (void)
{
     listhead = new node;
     listhead->val = 0;
     currnode = listhead;

     for (i=1; i<=2; i++)  {
          currnode->nextnode = new node;
          currnode = currnode->nextnode;
          currnode->val = i;
     }
     currnode->nextnode = NULL;

     for (currnode = listhead;currnode != NULL;
                    currnode=currnode->nextnode)
          cout << currnode->val << " ";

     return;
}
```

create a list of size 3 and set the last node pointer to NULL

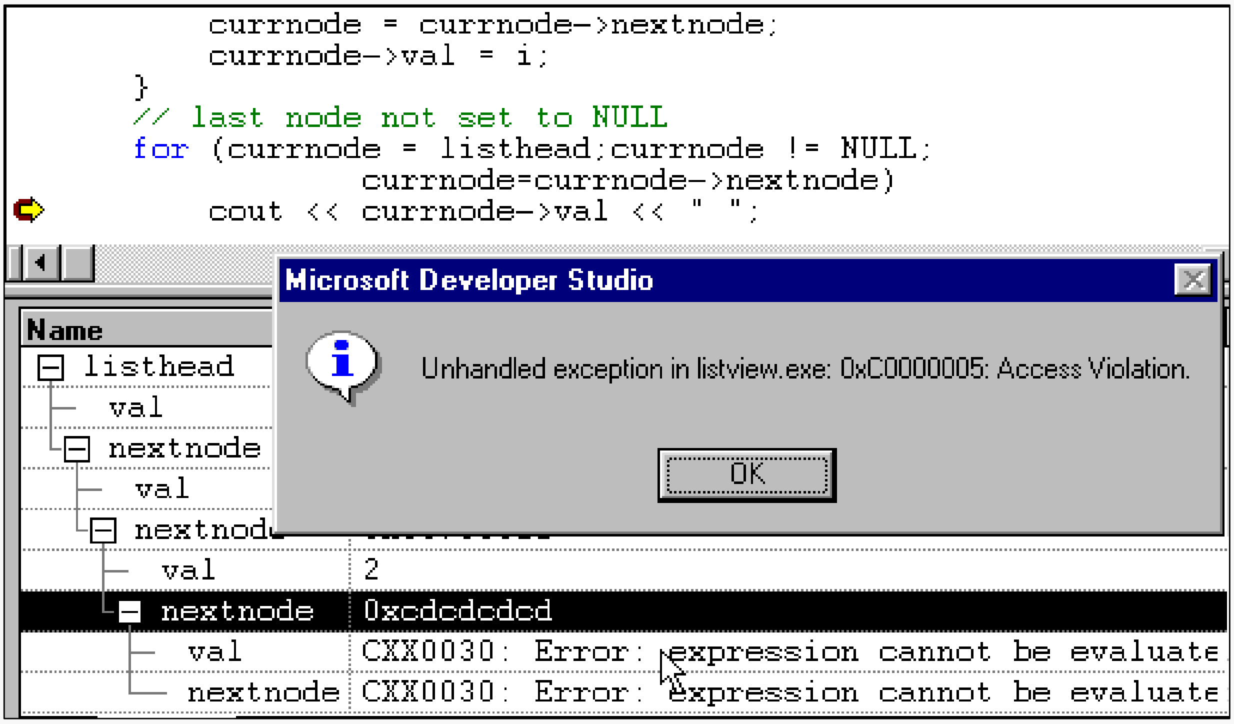
Dereferencing Invalid Addresses

– The tail of the list, when set to **NULL** will appear in the watch window as **0x00000000**. Any values at this point in the list will be inaccessible.



Name	Value
val	1
nextnode	0x0076031c
val	2
nextnode	0x00000000
val	13176734
nextnode	0x00700465

– If the last node is **NOT** set to **NULL**, the values will still be accessible but will most likely produce a run-time error or undesired program behavior, even in the debugger.



```
        currnode = currnode->nextnode;
        currnode->val = i;
    }
    // last node not set to NULL
    for (currnode = listhead;currnode != NULL;
            currnode=currnode->nextnode)
        cout << currnode->val << " ";
```

Microsoft Developer Studio

Unhandled exception in listview.exe: 0xC0000005: Access Violation.

OK

Name	Value
listhead	
val	
nextnode	
val	
nextnode	
val	2
nextnode	0xcdcdcdcd
val	CXX0030: Error: expression cannot be evaluate
nextnode	CXX0030: Error: expression cannot be evaluate