# Using the FoxPro module from ACS.NET library AcsLib.NET.dll AcsCppLib.NET.dll (for C++ support)

* #include “FoxPro.NET.h” and create a CFoxProBuffer instance (not a pointer)
  + Pass a file path to the constructor
* Iterate through records using a for loop and the CFoxProRecord class to read/write data

CFoxProBuffer fpBuffer("C:\\path\\file.dbf");

RecordArray fpRecords = fpBuffer.GetRecords();

for (int index = 0; index < fpBuffer.NumRecords(); index++)

{

// current record

CFoxProRecord currentRecord = fpRecords[index];

// do stuff with currentRecord  
 // see FoxPro.NET.h for full interface details  
}

* Call fpBuffer.Save() to commit changes to the DBF file (a backup is created in the *.backup* folder in the executable’s directory)
* Refer to FoxPro.NET.h for classes and type definitions.
* See example below for a simple but complete usage case. Includes project configuration for linking to the DLL

# Example: Code

/\*--------------------------------------------------

\* Example usage of FoxPro classes in native C++

\* ------------------------------------------------\*/

#include <stdio.h>

#include "FoxPro.NET.h"

using namespace AcsNetLib::FoxPro;

int main()

{

// create a CFoxProBuffer instance (give it a file path)  
 // relative paths work (i.e. if program EXE is in C:\Inv\Bin,  
 // you can say “..\\Invdata\\sys.dbf”)

CFoxProBuffer fpBuffer("C:\\Inv\\Invdata\\sys.dbf");

/\*--------------------------

accessing DBF records

---------------------------\*/

RecordArray record\_array = fpBuffer.GetRecords();

// always exit loop at NumRecords()!!

// using sys.dbf for this example

// - Show all lines at least 100 inches wide

printf("Lines at least 100 inches in width:\n");

for (int index = 0; index < fpBuffer.NumRecords(); index++)

{

// current record

CFoxProRecord record = record\_array[index];

// check the width; show lines with maxlen >= 100

char\* line\_name = record.Get("descr");

int line\_width = atoi(record.Get("maxlen"));

if (line\_width >= 100)

printf("%s: %d inches\n", line\_name, line\_width);

/\*--------------------------------------------------

modifying the DBF

example - change every line name to ACS

--------------------------------------------------\*/

record.Set("descr", "ACS");

}

// write fpBuffer changes to disk

fpBuffer.Save();

// pause, exit program

getchar();

return 0;

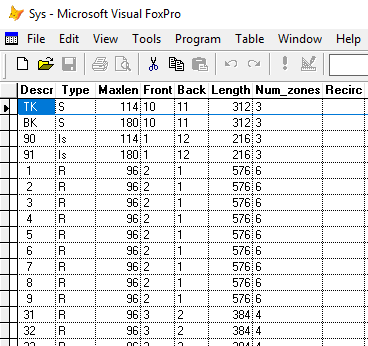
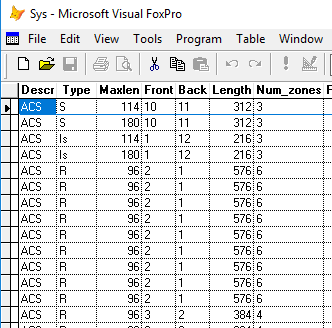
}

Notes

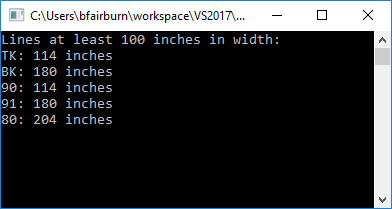
**Namespace**All ACS.NET classes are in the AcsNetLib namespace. Each module (SQL, FoxPro, etc.) has a sub-namespace under that.

**fpBuffer.GetRecords()**  
- returns a RecordArray (standard pointer-based array of   
 CFoxProRecords)  
  
 **record.Get(char\* field)**- returns a primitive string (char\*) with the record’s data in   
 the specified field  
- if an int value is required, conversion must be handled   
 separately (use atoi(record.Get(“field”)),   
 for example)  
  
**record.Set(char\* field, char\* value)**- field: which field (column) to update in the DBF  
- value: new value for this column  
  
  
**record.Save()**- commits all changes to the CFoxProBuffer to its file on   
 disk  
- saves a backup in the executable’s directory under the  
 *.backup* folder  
- Only necessary when exiting program, or if multiple   
 CFoxProBuffer instances are operating on the same file  
- .NET runtime automatically calls this through the   
 destructor (preventing data loss on a Winvent crash)

# Example: Results

*sys.dbf before running example code sys.dbf after running example code*



*example program output*

# Example: Project Configuration for Linking DLL

**Assuming ACS.NET library lives at C:\AcsNetLib**

Right click on project name in the Solution Explorer > left click *Properties*

|  |  |
| --- | --- |
|  | **VC++ Directories**   * update the highlighted areas as shown |
|  | **Linker > Input**   * add *AcsCppLib.NET.lib* to *Additional Dependencies* |
|  | **Build Events > Post-Build Event**   * Put the highlighted command in the *Command Line* property. This way the ACS.NET DLLs will automatically be pulled in to the build output. |

**Include FoxPro.NET.h**   
In C++ projects that need to use FoxPro classes, right click *Header Files* *> Add > Existing Item…* and choose the header – *C:\AcsNetLib\include\FoxPro.NET.h* in this example. This isn’t necessary but helps keep the project organized.

Class implementations are handled by the DLL. *FoxPro.NET.h* is the only source file that needs to be added to a C++ project.