VqStrings

Huge Variable and Fixed Length String Arrays

for

Visual Basic

by

Vi Qual Software

** Installation **

Create a sub directory of your Visual Basic directory called VQSTRING and copy or move VQSTRG.ZIP to that directory. Unzip the file and copy or move VQSTRING.DLL and VQSTRING.HLP to your WINDOWS or WINDOWS\SYSTEM directory. Load Visual Basic, open the project, VQDEMO.MAK, and run it as you would any other Visual Basic program.

** Quick Start **

In order to be able to access huge variable and fixed length strings as quickly as possible, either start with the VQSHELL application provided or cut the necessary declarations and functions from the VQSHELL application and paste them into your application.

(1) Enter the VqString declarations and constants into the Global module of your application. Copy these from VQSHELL.BAS.

(2) If you're going to use variable length strings, add the VqGetVarString and VqPutVarString functions to the 'General' section of your form or module. These functions are not necessary for fixed length strings.

(3) Use Visual Basic's 'On Error GoTo' to trap errors returned by the VqString functions.

(4) Use the calls shown in the VQSHELL application to initialize, access, and erase the huge VqString arrays.

** Introduction - Variable Length Strings **

If you're a Basic programmer, you already know that one of the biggest advantages of Basic over other programming languages is variable length string arrays. String lengths don't have to be pre-determined and data doesn't have to be limited to arbitrary lengths. With variable length strings, your programs can be more flexible. Names, addresses, pages of text, and much, much more can be of any length and can change dynamically as needed by your application.

Visual Basic provides variable length VqString arrays but they're limited to a total of 64K of string space. VqStrings are not limited to 64K but can use all memory available to Windows. VqStrings can be dimensioned in multiple megabytes, if the memory is available.

Calls to VqString variable length string functions are simple. A Basic string, a handle, an element number, and a Mode variable are passed to the function. The function declaration for variable length strings as follows:

Declare Function VqVarLenStr Lib "VqString.dll" (ByVal Strng As String, ByVal Handle As Integer, ByVal Page As Long, ByVal Mode As Long) As Long

The Mode argument can be any one of the following values:

```
Global Const VqVarGetSize = -1
Global Const VqGetString = -2
Global Const VqPutString = -3
Global Const VqUbound = -4
Global Const VqVarMemUsed = -5
Global Const VqVarMemFree = -6
Global Const VqEraseString = -7
```

The constants containing the characters, 'Var' only apply to variable length strings.

** Initialize/Allocate (Variable Length) **

If the value of Mode is a positive value, the function considers the call to be an initialization. In this case, the values of the arguments are interpreted as follows:

The Page argument will contain the number of elements that the huge array can access.

The Mode argument will contain the amount of memory to allocate for the array. The arguments both use a long integer so that the values can be large enough to dimension arrays limited only by the amount of memory available to the system. An array of long pointers is allocated to store the address of the elements in the array.

The Strng argument is ignored during initialization.

The Handle argument will be the number of the current array to be initialized. There is a limit of five huge variable length VqString arrays (accessed as 1 through 5).

Note: If an initialization is requested, on an array that has already been initialized, it will be re-initialized and any previous data will be erased.

** Using Variable Length VqString Arrays **

Calls to variable length VqString arrays are accomplished through two special functions, provided in VQDEMO and VQSHELL:

```
Function VqGetVarString (Strng$, Handle%, Page&)

x% = VqVarLenStr(Strng$, Handle%, Page&, VqVarGetSize)
If x% < 0 Then
  VqGetVarString = x%
  HugeError = x%
  Exit Function
End If
Strng$ = Space$(x%)
x% = VqVarLenStr(Strng$, Handle%, Page&, VqGetString)
If x% < 0 Then
  VqGetVarString = x%
  HugeError = x%
  Exit Function
End If

VqGetVarString = 0
HugeError = 0

End Function
```

```
Function VqPutVarString (Strng$, Handle%, Page&)

'------ Need to append Chr$(0) to end of string.
Strng$ = Strng$ + Chr$(0)
x% = VqVarLenStr(Strng$, Handle%, Page&, VqPutString)
If x% < 0 Then
  VqPutVarString = x%
  HugeError = x%
  Exit Function
End If

VqPutVarString = 0
HugeError = 0

End Function
```

-- Strng Argument --

Since Visual Basic sets up its own environment, the strings are not compatible with strings used by a DLL written in C. Strings can be passed to DLL's and modified, but the length cannot be altered. Therefore, when 'getting' strings form the variable length string functions, it is necessary to send a string of sufficient length to hold the data. A 'VqVarGetSize' Mode is provided to accomplish this. A call is made to the VqString function to obtain the size of a particular element and a Basic string is created, or sized, to accept the data on the actual VqGetString call. Sample procedures are provided in both the demonstration and shell programs.

Because of the need to obtain the length of the string before actually retrieving the data, a small amount of Basic code overhead has been introduced. A method that would increase the speed of string access from the VqString functions would be to size a string of sufficient length to hold any possible data stored in the array. The code to get the size of the string:

```
x% = VqVarLenStr(Strng$, Handle%, Page&, VqVarGetSize)
If x% < 0 Then
  HugeError = x%
  Exit Function
End If
Strng$ = Space$(x%)
```

would be replaced with:

```
Strng$ = Space$(100)
```

replacing the 100 shown here with a value specific to your application.

Size the string to a value that is sure to be large enough for any data retrieved from the VqString function. This would eliminate the need to obtain the length of the string first and, because the C function would place a terminating zero on the end of the string, the resulting string would be the exact size of the data retrieved. This method can only be used if you can predict an absolute maximum length for the data retrieved from the huge array.

Under most circumstances, data retrieval would be fast enough and the method that uses the 'VqVarGetSize' argument would be fine. It may amount to a matter of preference but both methods are provided.

-- Handle Argument --
The Handle argument will be the number of the current array to be accessed. There is a limit of five huge variable length VqString arrays (accessed as 1 through 5).

-- Page Argument --
The Page argument is the element of the array to be accessed.

-- Mode Argument --
The Mode argument determines which of the functions are performed on the VqString array, by the DLL.

Global Const VqVarGetSize = -1
The function returns the size of the element whose number is specified in the Page argument (for the VqString array specified in the Handle argument).

Global Const VqGetString = -2
The data contained in the element specified in the page argument is copied to the Strng argument (for the VqString array specified in the Handle argument).

Global Const VqPutString = -3
The data in the Strng argument is stored in the element specified in the Page argument (for the VqString array specified in the Handle argument).

Global Const VqUbound = -4
The function returns the number of elements dimensioned in the array specified in the Handle argument. All other arguments are ignored.

Global Const VqVarMemUsed = -5
The function returns the amount of memory that has actually been used to store data in the array specified in the Handle argument. All other arguments are ignored.

Global Const VqVarMemFree = -6
The function returns the amount of memory that remains available to store data in the array specified in the Handle argument. All other arguments are ignored.

Global Const VqEraseString = -7

This argument frees the memory that was allocated for the VqString array specified in the Handle argument. All other arguments are ignored.

** Introduction - Fixed Length Strings **

Fixed length strings are not as flexible as variable length strings but they are slightly faster and somewhat more memory efficient. It isn't necessary to obtain the length of the string before retrieving the data and there is no need for the DLL to keep an array of pointers to the elements.

Calls to VqString fixed length string functions are simple. A Basic string, a handle, an element number, and a Mode variable are passed to the function. The function declaration for variable length strings as follows:

Declare Function VqFixLenStr Lib "VqString.dll" (ByVal Strng As String, ByVal Handle As Integer, ByVal Page As Long, ByVal Mode As Long) As Long

The Mode argument can be any one of the following values:

Global Const VqGetString = -2
Global Const VqPutString = -3
Global Const VqUbound = -4
Global Const VqEraseString = -7

** Initialize/Allocate (Fixed Length)**

If the value of Mode is a positive value, the function considers the call to be an initialization. In this case, the values of Page and Mode are interpreted as follows:

The Page argument will contain the number of elements that the huge array can access.

The Mode argument will be the length of each element.

The Strng argument is ignored during initialization.

The Handle argument will be the number of the current array to be initialized. There is a limit of five huge variable length VqString arrays (accessed as 1 through 5).

Note: If an initialization is requested, on an array that has already been initialized, it will be re-initialized and any previous data will be erased.

** Using Fixed Length VqString Arrays **

-- Strng Argument --

The Strng argument must be a string of the same length as the Mode argument when the VqString array was initialized. This can be done with the Basic statement:

Strng$ = Space$(nn)

-- Handle Argument --

The Handle argument will be the number of the current array to be accessed. There is a limit of five huge variable length VqString arrays (accessed as 1 through 5).

-- Page Argument --

The Page argument is the element of the array to be accessed.

-- Mode Argument --

The Mode argument determines which of the functions are performed on the VqString array, by the DLL.

Global Const VqGetString = -2

The data contained in the element specified in the Page argument is copied to the Strng argument (for the VqString array specified in the Handle argument).

Global Const VqPutString = -3

The data in the Strng argument is stored in the element specified in the Page argument (for the VqString array specified in the Handle argument).

Global Const VqUbound = -4

The function returns the number of elements dimensioned in the array specified in the Handle argument. All other arguments are ignored.

Global Const VqEraseString = -7

This argument frees the memory that was allocated for the VqString array specified in the Handle argument. All other arguments are ignored.

** Common to Variable and Fixed length strings **

-- Subscripts --

VqString arrays are 'one origin' and there is no zero'th element. An 'Illegal Function Call' error code is returned if an attempt is made to access the zero'th element or an element beyond the number of elements that were allocated during initialization.

-- Error handling --

VqString functions return error codes compatible with the Visual Basic Error$ function. Errors can be trapped with Visual Basic's 'On Error GoTo' statements or handled individually by the programmer. The source code for the demonstration program is included and a shell, for development of other programs that will use VqString arrays, is also provided to make getting started easier. These can both be used as an example of how to handle errors returned from the VqString functions. A detailed explanation of the error handling as follows:

The following constants can be declared, in the Global module of your Visual Basic program, if you prefer to handler errors yourself:

Global Const IllegalFunctionCall = -5

(1) Returned if an invalid handle is sent to the function. Handles are limited to 1 through 5.

(2) Returned if an attempt was made to access an array that has not been initialized.

(3) Returned if an attempt was made to erase a VqString array that was not initialized.

Global Const OutOfMemory = -7

Returned if there is not enough memory available to the system to allocate the VqString array requested.

Global Const SubscriptOutOfRange = -9

(1) Returned if an attempt was made to access an element of a VqString array that is beyond the number of elements dimensioned during initialization.

(2) Returned on attempt to access element zero. VqString arrays are 'one origin' and are access from 1 though the number of elements dimensioned.

Global Const OutOfStringSpace = -14

This error pertains only to variable length VqString arrays. It is returned if there is not enough free space in the array to store the element specified.

These error constants need not be declared if you use only Visual Basic's 'On Error GoTo'. The value returned by the VqString function will generate an error message compatible with errors returned by Visual Basic's own string functions.

** Limitations of Demonstration DLL **

The DLL provided with the demonstration program is limited to 10,000 elements and 131,072 bytes (128K) of memory for the variable length strings and the fixed length VqString arrays are limited to 131,072 bytes (128K) in any combination of elements and string lengths. An 'Illegal Function Call' error code is returned if the dimensions are greater than these limits. The DLL provided with the registered version is not limited in any way.

** Quick Reference **

Function declarations:

```
Declare Function VqVarLenStr Lib "VqString.dll" (ByVal Strng As String, ByVal Handle As Integer,
ByVal Page As Long, ByVal Mode As Long) As Long

Declare Function VqFixLenStr Lib "VqString.dll" (ByVal Strng As String, ByVal Handle As Integer,
ByVal Page As Long, ByVal Mode As Long) As Long
```

The Constants for the Mode argument:

```
Global Const VqVarGetSize = -1
Global Const VqGetString = -2
Global Const VqPutString = -3
Global Const VqUbound = -4
Global Const VqVarMemUsed = -5
Global Const VqVarMemFree = -6
Global Const VqEraseString = -7
```

Function return error codes:

```
Global Const IllegalFunctionCall = -5
Global Const OutOfMemory = -7
Global Const SubscriptOutOfRange = -9
Global Const OutOfStringSpace = -14
```

** VqString Registration Form **

Registration entitles you to technical support and discounts on upgrades.

To: **Vi Qual Software**
28 Hill Drive
Rochester, NY 14626-1810

From: (please print or type)

Name: ____

Company: ____

Street: ____

City, ST, Zip: ____

Country: ____

Where did you hear about VqStrings? _______________________________

Disk size: (check one) ____ 5 1/4" ____ 3 1/2"

Quantity: ______ @ $29.95 ea. Sub Total $ __________

US and Canada	$2.50 shipping and handling.
European Countries	$5.00 shipping and handling.
Far East addresses	$7.50 shipping and handling.

Shipping and handling $ __________

New York State residents add 7% sales tax $ __________

Total enclosed $ __________

Make check or money order (in US currency) payable to:

Vi Qual Software