

Faster string comparison in vb6

Asked 11 years, 6 months ago

Modified 10 years, 7 months ago

Viewed 10k times



2

Am trying to make my vb6 app run faster, the reason is that am populating **vbaccelerators sgrid** with about 10k items all at once (this is a requirement from client).



I had to populate about 20 columns for each of the 10k items, and i have to perform string comparison in about more than half of them, so i wrote a string comparison function and did profiling



```
Function IsEqual(byval value1 as string, Byval value2 as string) as boolean  
    ' content, various versions are below  
End function
```

currently the items = 5000 and each of the time below shows the time it took and various versions of the function:

```
LCase$(Value1) = LCase$(value2)
```

time: 29149 ms

```
(StrComp(Value1, value2, 1) = 0 )
```

time: 30836 ms

```
If StrComp(Value1, value2, 1) = 0 Then  
    IsEqual = True  
Else  
    IsEqual = False  
End If
```

time 34180 ms

```
If StrComp(Value1, value2, 1) = 0 Then IsEqual = True
```

time 28387 ms

Timing is done with:

```
Declare Function timeBeginPeriod Lib "winmm.dll" (ByVal uPeriod As Long) As Long
Declare Function timeEndPeriod Lib "winmm.dll" (ByVal uPeriod As Long) As Long
Declare Function timeGetTime Lib "winmm.dll" () As Long
```

which returns the time in milliseconds.

Is there anyway to make the comparison faster?

performance vb6 comparison

Share

edited Jun 11, 2013 at 17:16

Improve this question

Follow

asked Jun 9, 2013 at 18:51



Smith

5,941 ● 17 ● 109 ● 163

I had thought calling an API might be faster, but my tests using CompareStringA showed it was 5 times slower. This was using an uncompiled version. One idea I stumbled across was to use byte arrays. I haven't tested it. Good luck. – [Tom Collins](#) Jun 9, 2013 at 19:59

The client requires VB6? – [John Saunders](#) Jun 10, 2013 at 0:06

5 You should take a look at the [VBSpeed](#) site. They have a textual string compare. – [Kelly Ethridge](#) Jun 10, 2013 at 6:23

2 Are you sure it is the string comparison? Typically it is the updates to the screen that take the most time. What grid are you using? Some grids have the ability to turn off updates, then you do your inserts, and then you turn updates back on for a single screen refresh. – [tcarvin](#) Jun 10, 2013 at 14:52

2 @TomCollins Remember that the A versions of API functions do implicit conversion from unicode to ANSI before passing the pointer. Try `CompareStringW()` declaring the string parameters as `Long` and using `StrPtr()`. – [Deanna](#) Jun 11, 2013 at 9:55 ✎

5 Answers

Sorted by: Highest score (default)



2



Things that might improve performance include..

- Change your parameters to ByRef
 - Using ByVal parameters copies the variables onto the stack. While this is a generally a good idea, if your comparison function is well-behaved and doesn't change the variables, making an extra copy of the data is not necessary.
- Populate the grid on demand,



- Only populate the parts of the grid that are showing on screen - track this with grid movement events. There are even grid controls for VB6 that facilitate this by letting you define "virtual" items and raising events to let you know which ones you need to populate. TList is the one I'm familiar with - I'll temper that suggestion with the caveat that it's licensing model can be a real PITA to work with.

Share Improve this answer Follow

answered Jun 9, 2013 at 22:54



Adrian

2,324 ● 18 ● 21

@Adrian what is Tlist? where can it be found. is it free control? – Smith Jun 10, 2013 at 10:57

Bennet-Tec TList ; not free, I'm afraid, but it's performance promises are accurate. bennet-tec.com/btproducts/TList/TList.htm ; I'm not affiliated, I just have some experience using it in the past. As I said, the licensing model is a total pain in the ass, but it's flexible and powerful.
– Adrian Jun 10, 2013 at 19:07

i am developing a free app, so i may not be able to use that – Smith Jun 12, 2013 at 8:50



This should be pretty fast.

1



Option Explicit

```
Private Declare Sub DerefByte Lib "msvbvm60" Alias "GetMem1" (ByVal Add As Long, ByRef Value As Byte)
```

```
Private Declare Sub DerefLong Lib "msvbvm60" Alias "GetMem4" (ByVal Add As Long, ByRef Value As Long)
```

```
Private Sub Form_Load()
```

```
    Debug.Print IsEqual("Hello", "hello")
```

```
    Debug.Print IsEqualB("Hello", "hello")
```

```
End Sub
```

```
Public Function IsEqualB(Str1 As String, Str2 As String) As Boolean
```

```
    Dim lpS1 As Long, lpS2 As Long
```

```
    Dim t1 As Byte, t2 As Byte
```

```
    Dim lSz As Long
```

```
    Dim i As Long
```

```
    IsEqualB = True
```

```
    lpS1 = StrPtr(Str1)
```

```
    lpS2 = StrPtr(Str2)
```

```
    DerefLong lpS1 - 4, lSz
```

```
    If lSz = LenB(Str2) Then
```

```
        For i = 0 To lSz - 1 Step 2
```

```

        DerefByte lpS1 + i, t1
        DerefByte lpS2 + i, t2
        If Not (t1 = t2) Then
            IsEqualB = False
            Exit For
        End If
    Next
Else
    IsEqualB = False
End If

End Function

Public Function IsEqual(Str1 As String, Str2 As String) As Boolean

    Dim lpS1 As Long, lpS2 As Long
    Dim t1 As Byte, t2 As Byte
    Dim lSz As Long
    Dim i As Long

    IsEqual = True

    lpS1 = StrPtr(Str1)
    lpS2 = StrPtr(Str2)
    DerefLong lpS1 - 4, lSz

    If lSz = LenB(Str2) Then
        For i = 0 To lSz - 1 Step 2
            DerefByte lpS1 + i, t1
            DerefByte lpS2 + i, t2
            If Not (t1 Or &H20) = (t2 Or &H20) Then
                IsEqual = False
                Exit For
            End If
        Next
    Else
        IsEqual = False
    End If

End Function

```

The basic premise here is to do byte by byte comparison mod 2 over the Unicode strings. One of the above functions is case sensitive, IsEqualB, then other is insensitive IsEqual.


Of course, it uses a couple of undocumented functions in the Visual Basic 6 runtime: but if you want speed, that's what you have to do, unfortunately.

Share

edited May 22, 2014 at 10:55

answered May 22, 2014 at 10:40

Improve this answer

 Occluded Sky

Follow

11 ● 2



Have you tried:

0

```
Function IsEqual(byval value1 as string, Byval value2 as string) as boolean
    Return StrComp(LCase$(Value1), LCase$(value2), vbBinaryCompare) = 0
End function
```

Share Improve this answer Follow

answered Jun 9, 2013 at 22:44



Tom Studee

10.4k ● 4 ● 41 ● 42

I get the sense that he needs case-insensitive comparison, since he goes out of his way to use it. – [Adrian](#) Jun 9, 2013 at 22:54

▲

You can probably cut your execution time in half by using "OPTION COMPARE TEXT". Place this line at the top of your code module.

0

```
OPTION COMPARE TEXT
```

This line, when used, will cause string compare within the code module be case insensitive. Because of this, you can simply use:

```
Function IsEqual(byval value1 as string, Byval value2 as string) as boolean
    IsEqual = (Value1 = Value2)
End Function
```

Share Improve this answer Follow

answered Jun 10, 2013 at 15:29



George Mastros

24.5k ● 5 ● 54 ● 62

Binary comparison is faster, but from the above, case-insensitive comparison is what is required. – [Adrian](#) Jun 10, 2013 at 19:23



look at the LockWindowUpdate() WinAPI call. This can really help grids when you are populating them . Make sure you call it once to lock the window and once to unlock it.

0

[Share](#) [Improve this answer](#) [Follow](#)

answered Jun 11, 2013 at 15:54



[gbrooksman](#)

115 ● 1 ● 6

