

Re: Suggestions to reduce memory use when splitting a string

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2006-04/msg00268.html>

- *From:* tomb <tomb@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 03 Apr 2006 16:10:12 -0400
-

klineb wrote:

Good Day,

I have written a utility to convert our DOS COBOL data files to a SQL Server database. Part of the process requires parsing each line into a sql statement and validating the data to keep the integrity of the database. We are parsing roughly 81 files and range in size 1 kb to 65 MB files (Average of 400,000 lines in the larger files).

I have written this utility with VB.NET 2003 and when I parse all of the files I run out of memory. The following functions seem to be the main source of my leak. Any help optimizing this code is appreciated.

Public Function SplitDelimitedLine(ByVal CurrentLine As String, ByVal

SplitString() As String) As Boolean

'7-25-2005

'BJK

,

'---removed the use of Char replaced with: CurrentLine.Chars(i)

,

'-----

Dim i As Integer

Dim CountDelimiter As Boolean

Dim Total As Integer

Dim lbResult As Boolean

Dim Section As New String Builder

Dim liLen As Integer

Dim liCommaPos As Integer

Dim liQuotePos As Integer

Try

'We want to count the delimiter unless it is within the text qualifier

CountDelimiter = True

Re: Suggestions to reduce memory use when splitting a string

```
Total = 0
liLen = CurrentLine.Length - 1
For i = 0 To liLen
Select Case CurrentLine.Chars(i)
Case gsDoubleQuote
If CountDelimiter Then
CountDelimiter = False
Else
CountDelimiter = True
End If
Case gsComma
If CountDelimiter Then
' Add current section to collection
SplitString(Total) = Section.ToString.Trim
Section = Nothing
Section = New StringBuilder
Total = Total + 1
Else
Section.Append(CurrentLine.Chars(i))
End If
Case Else
Section.Append(CurrentLine.Chars(i))
End Select
Next
' Get the last field – as most files will not have an
ending delimiter
If CountDelimiter Then
' Add current section to collection
SplitString(Total) = Section.ToString
End If
lbResult = True
Catch ex As Exception
ps_LastErrSource = ex.Source
ps_LastErrDesc = ex.ToString
lbResult = False
Dim loSB As New StringBuilder(ps_LastErrDesc)
UpdateLog(loSB)
End Try
Return lbResult
End Function
```

This function is stored in a class and is called from other function within this class.

Thanks
Brian

It looks to me like your field separator is always a comma. So why not just use `SplitString = split(CurrentLine, ",")`, then use `Replace` on each string in the resultant array if you want to get rid of the

Re: Suggestions to reduce memory use when splitting a string

quotes. I think this will be much faster than going char by char.

The parameter `SplitString` will have to be declared as `byRef SplitString as Array`, rather than `SplitString() as String`.

Tom

.