

Forum:[Hack Forums](#)

Thread: [GZip Class with extended functionality](#)

Barret - February 4, 2017, 9:37 am

GZip extension

This GZip extension is a class based on a memory stream with the addition that it has a few extra functions that can either decompress or compress your byte array.

You really don't need to call any functions from the memory stream itself, its all handled as is.

Since it is a bit of a challenge to get a clear indication how far a stream is with reading or writing i made this extension.

GZip in action in my own project (Archiver)

Declarations

Remark: Can also be used within the USING .. END USING block since its base supports IDisposable interface obviously

```
Code New GZipStream(BufferSize)
New GZipStream(AutoSize)
New GZipStream(CompressionLevel)
New GZipStream(CompressionLevel, BufferSize, AutoSize)
Progress Callback
```

The progress delegate is as follows

```
Code Public Delegate Sub ProgressCallback(Position As Int64, Total As Int64)
```

Examples

You can either hook your own sub method or lambda the call back and raise your event.

Code Public Class Example

```
Public Event Progress(Position As Int64, Total As Int64)
```

```
Public Sub ShowProgression(Position As Int64, Total As Int64)
```

```
'...code...
```

```
End Sub
```

```
Public Function Compress(src() As Byte) As Byte()
```

```
Using gzip As New GZipStream(CompressionLevel.Fastest, Buffer.VeryLarge, True)
```

```
Return gzip.Compress(src, AddressOf Me.ShowProgression)
```

```
End Using
```

```
End Function
```

```
Public Function Decompress(src() As Byte, Callback As GZipStream.ProgressCallback) As
Byte()
```

```
Using gzip As New GZipStream(CompressionLevel.Fastest, Buffer.Standard, True)
```

```
Return gzip.Decompress(src, Sub(x, y) RaiseEvent Progress(x, y))
End Using
End Function
```

End Class

Example of getting and showing the progress

Remark: When the progress is done, the GZip extension event will raise a zero for both parameters

```
Code Private Sub UpdateGUI(n as int64, t as int64)
    If (Me.Textbox.InvokeRequired) Then
        Me.Textbox.Invoke(Sub() Me.UpdateGUI(n,t))
    Else
        If (n = 0 AndAlso t = 0) Then
            Me.Textbox.Text = "Ready"
        Else
            Me.Textbox.Text = String.Format("Busy {0}%", Math.Floor((n / t) * 100))
        End If
    End If
End Sub
```

Last note

Since this is possibly a long process of large files etc... you obviously want this to be ran in a separate thread.

Source

```
Code Imports System.IO
Imports System.IO.Compression
Imports System.Runtime.InteropServices
Public NotInheritable Class GZipStream
    Inherits MemoryStream
    Public Enum Buffer As Int32
        Small = &HFF
        Standard = &HFFF
        Large = &HFFFF
        VeryLarge = &HFFFFFF
    End Enum
    Public Delegate Sub ProgressCallback(Position As Int64, Total As Int64)
    Public Property Level As CompressionLevel
    Public Property BufferMaxSize As Int32
    Public Property BufferAutoSize As Boolean
    Sub New()
        Me.Level = CompressionLevel.Fastest
        Me.BufferMaxSize = DirectCast(Buffer.Standard, Int32)
        Me.BufferAutoSize = True
    End Sub
    Sub New(Buffer As Buffer)
        Me.Level = CompressionLevel.Fastest
        Me.BufferMaxSize = DirectCast(Buffer.Standard, Int32)
        Me.BufferAutoSize = False
    End Sub
```

```

Sub New(Level As CompressionLevel)
    Me.Level = Level
    Me.BufferMaxSize = DirectCast(Buffer.Standard, Int32)
    Me.BufferAutoSize = False
End Sub
Sub New(BufferAutoSize As Boolean)
    Me.Level = Level
    Me.BufferMaxSize = DirectCast(Buffer.Standard, Int32)
    Me.BufferAutoSize = BufferAutoSize
End Sub
Sub New(Level As CompressionLevel, BufferSize As Buffer)
    Me.Level = Level
    Me.BufferMaxSize = DirectCast(BufferSize, Int32)
    Me.BufferAutoSize = False
End Sub
Sub New(Level As CompressionLevel, BufferSize As Buffer, BufferAutoSize As Boolean)
    Me.Level = Level
    Me.BufferMaxSize = DirectCast(BufferSize, Int32)
    Me.BufferAutoSize = BufferAutoSize
End Sub
Public Function Compress(source() As Byte, Progress As ProgressCallback) As Byte()
    Dim p As Integer = 0, Buffer As Byte() = New Byte(Me.AutoAdjustBuffer(source.Length - 1)) {}
    Using deflate As New DeflateStream(Me, Me.Level, True)
        Do
            Buffer = Me.Copy(source, p, Buffer, 0)
            If (p < source.Length) Then
                deflate.Write(Buffer, 0, Buffer.Length)
                p += Buffer.Length
                Progress.Invoke(p, source.Length)
            End If
            Loop While p < source.Length
        End Using
        Progress.Invoke(0, 0)
        Return Me.ToArray
    End Function
Public Function Decompress(source() As Byte, Progress As ProgressCallback) As Byte()
    Dim p As Int32 = 0, buffer As Byte() = New Byte(Me.AutoAdjustBuffer(source.Length - 1)) {}
    Using deflate As New DeflateStream(New MemoryStream(source),
CompressionMode.Decompress)
        Do
            p = deflate.Read(buffer, 0, buffer.Length)
            If p > 0 Then
                Progress.Invoke(deflate.BaseStream.Position - 1, deflate.BaseStream.Length - 1)
                Me.Write(buffer, 0, p)
            End If
            Loop While p > 0
            Progress.Invoke(0, 0)
            Return Me.ToArray
        End Using
    End Function
Private Function AutoAdjustBuffer(Length As Integer) As Integer
    If (Not Me.BufferAutoSize) Then Return Me.BufferMaxSize
    Dim n As Integer = &H400

```

```

Do
    n += &H400
Loop Until n >= Length Or n >= Me.BufferMaxSize
Return n
End Function
Private Function Copy(source() As Byte, sourceindex As Int64, destination() As Byte,
destinationindex As Int64) As Byte()
    If (destination.Length > (source.Length - sourceindex)) Then
        destination = New Byte(Convert.ToInt32((source.Length - sourceindex) - 1)) {}
    End If
    System.Array.Copy(source.ToArray, sourceindex, destination, destinationindex,
destination.Length)
    Return destination
End Function
End Class

```

Have fun, -B

GruntStyl - February 4, 2017, 10:29 am

Awesome contribution Barret, I'm sure myself and other will find this useful. Thank you as always :)

Ninja - February 4, 2017, 11:58 am

Thanks for another awesome release Barret! :-)

Cairns - February 5, 2017, 7:10 am

Nice work, I might put this to use in a project at some point if you don't mind? let me know how you feel about that. Thanks for the contribution. I'm sick of browsing around here from time to time and seeing the same shitty posts.

Barret - February 5, 2017, 7:50 am

(Feb 5, 2017 08:10 AM)Cairns Wrote: [\[CITE\]](#) Nice work, I might put this to use in a project at some point if you don't mind? let me know how you feel about that. Thanks for the contribution. I'm sick of browsing around here from time to time and seeing the same shitty posts.

Its open and public domain bud, use it the way you see fit. If i publish source like this, there are no rules, demands or silly things that require to use, edit or reproduce it. ?

fec-ri.muhyayel - March 8, 2019, 9:30 am

which unarchive files ?

example : .bin,.arc