

Re: Asynchroner Download mit WebRequest

Source:

<http://www.tech-archive.net/Archive/German/Entwicklung/microsoft.public.de.german.entwickler.dotnet.vb/2004-04>

From: Arne Janning (*spam-me.here_arnolo_at_msn.com*)

Date: 04/06/04

Date: Wed, 7 Apr 2004 00:19:54 +0200

Hallo Stefan,

ich weiss nicht recht, was das soll. In einer Windows.Forms-Anwendung bist Du darauf angewiesen, dass die Oberfläche weiter bedienbar bleibt, während der WebRequest durchgeführt wird. Dies kannst Du auch dadurch erreichen, dass Du einfach einen Thread startest und ihm die Download-Funktion übergibst. Intern wird die Datei ohnehin asynchron heruntergeladen, wie man sieht, wenn man die System.Net.HttpWebResponse.GetResponse-Methode in MSIL anschaut:

```
[...]  
L_0020: callvirt WebRequest.BeginGetResponse  
[...]  
L_0030: callvirt IAsyncResult.get_IsCompleted  
[...]  
L_0038: callvirt IAsyncResult.get_AsyncWaitHandle  
[...]  
L_0044: callvirt WaitHandle.WaitOne  
[...]  
L_006c: callvirt WebRequest.EndGetResponse  
[...]
```

Wenn Du eine Benachrichtigung brachst, wenn der Download abgeschlossen ist, kann die Download-Methode auch ein Ereignis feuern, auf das Du dann in der Form reagieren kannst. Auch eine ProgressBar ist möglich, wie in dem Beispiel unten.

[getestet]

```
Imports System  
Imports System.Net  
Imports System.IO  
Imports System.Threading
```

```
Public Class frmMain  
    Inherits System.Windows.Forms.Form
```

```
Private Const SOURCE As String = "DEINE URL"
Private target As String = "test.txt"
Private WebReq As WebRequest
Private WebResp As WebResponse
Private bReader As BinaryReader
Private bWriter As BinaryWriter
Dim outStream As FileStream
Dim buffer() As Byte = New Byte(1024) {}
Dim bytesRead As Integer
Dim t As thread

#Region " Vom Windows Form Designer generierter Code "
Public Sub New()
    MyBase.New()
    InitializeComponent()
End Sub
' Die Form überschreibt den Löschvorgang der Basisklasse, um Komponenten zu
bereinigen.
Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
    If disposing Then
        If Not (components Is Nothing) Then
            components.Dispose()
        End If
    End If
    MyBase.Dispose(disposing)
End Sub
' Für Windows Form-Designer erforderlich
Private components As System.ComponentModel.IContainer
'HINWEIS: Die folgende Prozedur ist für den Windows Form-Designer
erforderlich
'Sie kann mit dem Windows Form-Designer modifiziert werden.
'Verwenden Sie nicht den Code-Editor zur Bearbeitung.
Friend WithEvents btnDownload As System.Windows.Forms.Button
Friend WithEvents progBar As System.Windows.Forms.ProgressBar
<System.Diagnostics.DebuggerStepThrough(> Private Sub InitializeComponent()
Me.btnDownload = New System.Windows.Forms.Button
Me.progBar = New System.Windows.Forms.ProgressBar
Me.SuspendLayout()
'
'btnDownload
'
Me.btnDownload.Location = New System.Drawing.Point(432, 102)
Me.btnDownload.Name = "btnDownload"
Me.btnDownload.Size = New System.Drawing.Size(163, 26)
Me.btnDownload.TabIndex = 0
Me.btnDownload.Text = "Start Download"
'
'progBar
'
Me.progBar.Location = New System.Drawing.Point(8, 24)
Me.progBar.Name = "progBar"
```

```
Me.progressBar.Size = New System.Drawing.Size(584, 24)
Me.progressBar.TabIndex = 1
'
'frmMain
'
Me.AutoScaleBaseSize = New System.Drawing.Size(6, 15)
Me.ClientSize = New System.Drawing.Size(604, 134)
Me.Controls.Add(Me.progressBar)
Me.Controls.Add(Me.btnDownload)
Me.Name = "frmMain"
Me.Text = "Downloader"
Me.ResumeLayout(False)
End Sub
#End Region

Private Sub btnDownload_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnDownload.Click
    t = New Thread(AddressOf Download)
    t.Name = "DownloadThread"
    t.IsBackground = True
    t.Start()
End Sub

Private Sub Download()
    Try
        WebReq = HttpWebRequest.Create(SOURCE)
        WebResp = WebReq.GetResponse
        progressBar.Maximum = CInt(WebResp.ContentLength)
        outputStream = New FileStream(target, FileMode.Create)
        bReader = New BinaryReader(WebResp.GetResponseStream())
        bWriter = New BinaryWriter(outputStream)
        Do
            bytesRead = bReader.Read(buffer, 0, 1024)
            bWriter.Write(buffer, 0, bytesRead)
            progressBar.Value += bytesRead
            progressBar.Update()
        Loop Until bytesRead = 0
        bReader.Close()
        bWriter.Close()
        outputStream.Close()
        MessageBox.Show("Download beendet")
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub
End Class
```

- > *Leider liest er nicht alle Bytes ein! D.h. die Callback-Funktion wird*
- aufgerufen bevor er*
- > wirklich fertig ist.*
- > Beim Debuggen hat das Byte-Array _abBuffer gegen das Ende nur noch Nullen*

drin. Komischerweise sind es auch nicht immer gleich viele

> *Bytes, die Nullen enthalten.*

>

> *Kann mir jemand einen Tipp geben was ich falsch mache?*

>

> *Danke und Gruss,*

> *Steff*

>