24.2.7 Klasse DownloadManager

Die Klasse DownloadManager (gb.net.curl) implementiert einen Manager, der eine Liste von Downloads abarbeitet. Sie fügen ihm eine Liste der URLs hinzu, deren Inhalt heruntergeladen werden soll.

Der Download-Manager verarbeitet die Downloads transparent im Hintergrund und löst dabei Ereignisse aus, so dass Sie stets wissen, was passiert.

So erzeugen Sie einen DownloadManager und fügen zum Beispiel zwei Download-Adressen hinzu:

```
Private $hManager As DownloadManager

$hManager = New DownloadManager As "DLManager" ' "DLManager" ist der Event-Name
$hManager.MaxClient = 2

$hManager.Add("https://www.gambas-buch.de/lib/exe/fetch.php?media=24:24.1:24.1.4:dnsclient.tar.gz")
$hManager.Add("https://www.gambas-buch.de/lib/exe/fetch.php?media=24:24.4.1:smtp.client-0.6.1.tar.gz")
```

24.2.7.1 Eigenschaften

Die Klasse *DownloadManager* verfügt über folgende Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Count	Integer	Gibt die Anzahl der abzuarbeitenden Downloads zurück.
MaxClient	Integer	Legt die maximale Anzahl der Download-Clients fest oder gibt deren Anzahl zurück.
Progress	Float	Gibt den Fortschritt für alle Downloads als reelle Zahl zwischen 0 und 1 zurück.
Size	Integer	Gibt die Anzahl der herunter geladenen Bytes aller Downloads zurück.
Speed	Integer	Gibt die aktuelle Download-Geschwindigkeit in Bytes/s zurück.
State	Integer	Gibt den Status des DownloadManagers zurück. Die Werte sind NotReady (0), Ready (1), Downloading (2), Finnish (3) oder Error (4).

Tabelle 24.2.7.1.1 : Eigenschaften der Klasse DownloadManager

Hinweise:

- Alle Eigenschaften – bis auf MaxClient – können nur ausgelesen werden!
- Der Wert für die Integer-Eigenschaft MaxClient darf nur zwischen 1 und 32 liegen.
- Mit den Werten der Eigenschaft Progress im Intervall [0,1] können Sie zum Beispiel eine Progressbar direkt ansteuern. Wenn die Eigenschaft Progressbar.Label = True ist, dann wird der Download-Fortschritt zusätzlich mit einer Prozent-Angabe angezeigt.
- Beachten Sie: Wenn der Downloadmanager zum Beispiel 3 Download-Clients verwaltet und Download1.Status = Download.Ready ist, Download2.Status = Download.Downloaded und Download3.Status = Download.Finnish, dann wird als Status stets der höchste Wert ausgegeben → 4.

24.2.7.2 Methoden

Die Klasse *DownloadManager* verfügt über diese vier Methoden:

Methode	Beschreibung
Add (URL As String [, Key As String])	Fügt einen URL zur Liste der abzuarbeitenden Downloads hinzu. Der Parameter Key ist optional.
Clear()	Löscht die Liste (Datentyp Collection) aller vom Manager abzuarbeitenden Downloads.
Start()	Startet den DownloadManager.
Stop()	Stoppt den DownloadManager.

Tabelle 24.2.7.2.1 : Methoden der Klasse DownloadManager

Hinweise:

- Wenn der optionale Parameter 'Key' in der Add(…)-Methode nicht angegeben wird, dann wird der Key auf Key = CStr($iKey) gesetzt – mit "1" beginnend. Intern wird der Integer-Wert $iKey hochgezählt.
- Für jeden Download ist das primäre Zielverzeichnis Temp$().
- Sie finden zum Beispiel die Download-Datei für den 1. Download im temporären Verzeichnis: /tmp/gambas.1000/7533/1.tmp. Der Name des Unterordners 7533 entspricht der Prozess-ID des aktuellen Gambas-Prozesses.
- Achtung: Alle Dateien, die sich im Verzeichnis /tmp/gambas.<UserId>/<ProcessId> befinden, werden automatisch gelöscht, wenn das Gambas-Programm beendet wird.
- Daher sollten Sie diese geladenen Dateien – unter ihrem originalen Dateinamen – in einen Ordner Ihrer Wahl kopieren.

24.2.7.3 Ereignisse

Die Klasse *DownloadManager* verfügt über diese Ereignisse:

Ereignis	Beschreibung
Connect	Das Ereignis wird ausgelöst, wenn der Download-Manager gestartet wurde und sich der aktuelle Client zum Server verbunden hat.
End	Das Ereignis wird ausgelöst, wenn alle Downloads abgearbeitet wurden.
Error (Key As String)	Das Ereignis wird ausgelöst, wenn ein Fehler bei einem Download ausgelöst wird.
Finish (Key As String)	Das Ereignis wird ausgelöst, wenn ein Download beendet wurde.
Progress	Das Ereignis wird ausgelöst, wenn Daten geladen werden.
Size (Key As String)	Das Ereignis wird jeweils ausgelöst, wenn der Download-Manager die Größe der zu ladenden Download-Datei mit DownloadManager[Key].Size ausliest.

Tabelle 24.2.7.3.1 : Ereignisse der Klasse DownloadManager

Hinweise:

- Connect: Nur für das Protokoll HTTP können Sie auch den Status auslesen.
- Error (Key): Über den ausgelesenen Key können Sie feststellen, welcher Download fehlerhaft war.
- Finish(Key): Welcher Download beendet wurde, können Sie über den Key ermitteln.
- Size(Key): Für die Ermittlung der Dateigröße einer Download-Datei muss der DownloadManager nicht gestartet sein! Probleme können auftreten, wenn Weiterleitungen (Redirects) im Spiel sind. Dann lässt sich die Größe Download-Datei nicht ermitteln – es wird dann die Größe 0 zurückgegeben. Der Download wird aber ausgeführt!

24.2.7.4 Projekt

Im Projekt *DLManager* wird vom DownloadManager eine vorgegebene Liste von 3 Downloads verwaltet. Nach dem Programmstart wird zuerst geprüft, ob sich eine Verbindung in das Internet herstellen lässt. Dann werden alle drei Downloads ausgeführt. Der Download-Fortschritt wird für jeden Download (→ 69%) und subsummierend (→ 35%) optisch angezeigt:

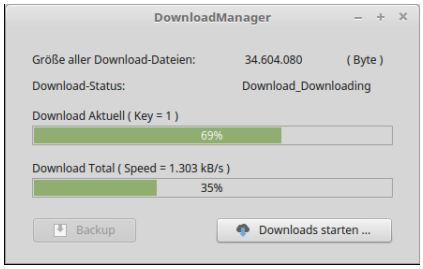

Abbildung 24.2.7.4.1: GUI DownloadManager

Nach dem Ende aller Downloads werden die im vorgegebenen Basis-Verzeichnis /tmp/gambas.<UserId>/<ProcessId> gespeicherten Dateien mit ihrem originalen Datei-Namen in das Verzeichnis /home/username/Downloads/DLD kopiert (Backup):

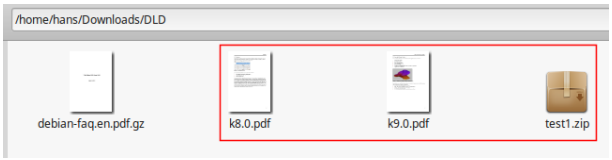

Abbildung 24.2.7.4.2: Inhalt des Download-Ordners DLD

Das Zielverzeichnis können Sie im Quelltext fest vorgeben oder im Dialog frei festlegen. Im Projekt wurde die zweite Möglichkeit nicht genutzt.

Der Quelltext wird vollständig angegeben:

```
' Gambas class file

Public hDownloadManager As New DownloadManager
Public sDownloadFile As New String[]

Private $iTotalSize As Integer
Private $iCurrentKey As Integer
Private $aURL As String[]
Private $aOriginalFilenames As String[]
Private $TempDirectory As String
Private $AverageSpeed As Integer

Public Sub Form_Open()

    Dim sURL As String

    FMain.Resizable = False
    MCheck.Check_Network()
    $aURL = New String[]
    hDownloadManager = New DownloadManager As "DLM"
    $iCurrentKey = 1
    btnBackup.Enabled = False

'-- Begin: Definition of the download URLs ------------------------------------------------------

    sURL = "https://www.gambas-buch.de/lib/exe/fetch.php?media=8:8.0:k8.0.pdf"
    $aURL.Add(sURL)
    hDownloadManager.Add(sURL)

    sURL = "https://www.gambas-buch.de/lib/exe/fetch.php?media=test1.zip"
    $aURL.Add(sURL)
    hDownloadManager.Add(sURL)

    sUrl = "https://www.gambas-buch.de/lib/exe/fetch.php?media=9:9.0:k9.0.pdf"
    $aURL.Add(sURL)
    hDownloadManager.Add(sURL)

'-- End: Definition of the download URLs ------------------------------------------------------

    Select hDownloadManager.Count
      Case 1
        tblTotalSize.Text = "Größe der Download-Datei:"
      Case 2
        tblTotalSize.Text = "Größe der beiden Download-Dateien:"
      Case 3
        tblTotalSize.Text = "Größe aller " & Str(hDownloadManager.Count) & " Download-Dateien:"
    End Select

End

Public Sub btnStart_Click()
    $AverageSpeed = 0
    hDownloadManager.Start()
End

Public Sub btnBackup_Click()
    Backup()
End

Public Sub DLM_Progress()
    Timer1.Start()
    ProgressBarCurrent.Value = hDownloadManager[$iCurrentKey].Current/ hDownloadManager[$iCurrentKey].Size
```

```
    ProgressBarTotal.Value = hDownloadManager.Progress
    SetStatus(hDownloadManager.State)
    lblCurrentDLSize.text = "Download Aktuell ( Key = " & $iCurrentKey & " )"
End

Public Sub DLM_Error(Key As String)
    If Not hDownloadManager[Key].ErrorText Then
       Print "No Error!"
    Else
       Print "Error = "; hDownloadManager[Key].ErrorText; " bei "; hDownloadManager[Key].Url
       hDownloadManager.Stop()
       hDownloadManager.State
    Endif
End

Public Sub DLM_End()
    SetStatus(hDownloadManager.State)
    hDownloadManager.Stop()
    btnBackup.Enabled = True
    Timer1.Stop()
End

Public Sub DLM_Finish(Key As String)
    Inc $iCurrentKey
    $TempDirectory = Split(hDownloadManager[Key].Path, "/")[3]
    Try ProgressBarCurrent.Value = hDownloadManager[Key].Current / hDownloadManager[Key].Size
End

Public Sub DLM_Size(Key As String)
    $iTotalSize += hDownloadManager[Key].Size
    lblTotalSize.Text = Format($iTotalSize, "###,###,##0")
End

Public Sub Timer1_Timer()
    If $AverageSpeed = 0 Then
       $AverageSpeed = hDownloadManager.Speed  'Initial speed
    Else
       $AverageSpeed = 0.9 * $AverageSpeed + 0.1 * hDownloadManager.Speed 'Floating average speed
    Endif
    lblTotalDLSize.text = "Download Total ( Speed = " & Format($AverageSpeed / 1024, "0,000") & " kB/s )"
End

Private Sub RenameFiles(Source As String)

    Dim sFilter, i As Integer
    Dim sFile, sPattern, sURL, sFilename, sLastSubFolder As String
    Dim aDir As String[]

    $aOriginalFilenames = New String[]
    aDir = New String[]

    For Each sURL In $aURL
      sLastSubFolder = Split(sUrl, "/").Last
      If String.InStr(sLastSubFolder, "?") <> 0 Or
         String.InStr(sLastSubFolder, "=") <> 0 Or
         String.InStr(sLastSubFolder, ":") <> 0 Then
         sFilename = Split(sLastSubFolder, "?,=,:", "", False, False).Last
         $aOriginalFilenames.Add(sFilename)
       Else
          $aOriginalFilenames.Add(sLastSubFolder)
       Endif
    Next

    sPattern = "*.tmp"
    sFilter = gb.File
    i = 0
    For Each sFile In Dir(Source, sPattern, sFilter).Sort(0)
      Move Source &/ sFile To Source &/ $aOriginalFilenames[i]
      Inc i
    Next

End

Private Sub CopyDir(Source As String, Destination As String)

    Dim sFile, sPattern As String
    Dim sFilter As Integer
    Dim FileInfo As Stat

    sPattern = "*.*"
    sFilter = gb.File

    If Not Exist(Destination) Then Mkdir Destination
```

```
    For Each sFile In Dir(Source, sPattern, sFilter)
      FileInfo = Stat(Source &/ sFile)
'-- The content of the error text file (with the original extension) is 'Not Found'
'-- if the download file was cannot be found. OR 'Bad request'
      If FileInfo.Size = 9 Then
         Message.Info("<b>The file " & sFile & " was not found!</b><hr>The content of the saved error text
file (with the original extension) is 'Not Found', if the download file was cannot be found.")
      Endif
'-- The content of the error text file (with the original extension) is 'Bad request'
'-- if the download file was cannot be found.
      If FileInfo.Size = 11 Then
         Message.Info("<b>The file " & sFile & " was not found!</b><hr>The content of the saved error text
file (with the original extension) is 'Bad request', if the download file was cannot be found.")
      Endif

      If Not Exist(Destination &/ sFile) Then
         Copy Source &/ sFile To Destination &/ sFile
      Endif
    Next

End

Private Sub Backup()

    Dim sSource, sDestination As String

    sSource = "/tmp/gambas." & System.User.Id &/ $TempDirectory
    sDestination = System.User.Home &/ "Downloads" &/ "DLD"
      RenameFiles(sSource)
      CopyDir(sSource, sDestination)
    Print "DONE!"

End

Private Function SetStatus(iStatus As Integer)

    Select iStatus
      Case Download.NotReady
        lblStatus.Text = "Download_NotReady"
      Case Download.Ready
        lblStatus.Text = "Download_Ready"
      Case Download.Downloading
        lblStatus.Text = "Download_Downloading"
      Case Download.Finish
        lblStatus.Text = "Download_Finish"
      Case Download.Error
        lblStatus.Text = "Download_Error"
    End Select

End

Public Sub Form_Close()
    hDownloadManager.Stop()
End
```