NetCode Control (TM) Version 1.30

Properties Events Exported Functions Error Codes Copyright & Registration

Description

The **NetCode** Control can be used to encode or decode files or strings. **UUEncode** as well as MIME's **Base64** and **Quoted-Printable** formats are currently supported.

File Name

NETCODE.VBX

Object Type

NetCode

Remarks

NetCode's operation is controled by assigning a value to the Action Property. The encoding format should be given at Format. The binary data (or their filename) is assigned to DecodedData Property and the encoded ones to the EncodedData Property. The FileName Property may be used to override the default filenaming or to specify a directory during UUEncoding. After UUDecoding it contains the full specified file name of the decoded file.

Because of the restrictions put on the length of a Visual Basic String (64K), two functions are exported: Encode and Decode.

If you have any questions, suggestions, or need any assistance, you can contact us via email at **devsoft@aol.com**. We will try to answer all messages, however, messages from registered users will have higher priority, so please include your serial number in your message for faster service.

We also strongly recommend that you visit our WWW site at **http://www.dev-soft.com/devsoft** . There you will find the latest versions of our shareware products as well as other helpful information.

Understanding Encoding/Decoding

Most of mail systems use only 7 bits to transmit mesages. A binary file such an archive or a non-plain text formated file produced from a text processor should be first encoded in 7 bit code before EMail transfer.

The following encoding formats are currently supported:

UUDecoding

Base64 Decoding

Quoted Printable Encoding

Decoding is the inverse process: creation of the original file from the encoded data. Encoded data are often split over several files because of the size limit put to the EMail message. Each message is preceded by information on splitting and also the mail header. **NetCode** supports this case in both directions:

- During **Encoding** setting the MaxFileSize Property to the maximum size of a message body instructs **NetCode** to split the encoded data over several files. Multiple filenames can be specified using question marks **"?"**.
- During **Decoding**, the body messages can be saved in separate files named *namexxx.ext* where

xxx is a numeral starting at 000 and *ext* is **".uue"**, **".b16"**, or **".q_p"** according to Format. This multiple filename should be given to EncodedData. Currently only while UUDecoding if Intellicode Property is set to **True**, **NetCode** tries to filter the data that don't belong to the pure uuencoded ones (also the mail header).

UUEncoding codes 3 bytes of 8 bit code into 4 bytes of 7 bit ASCII code that are ensured to be correctly interpreted from the receiving system. An uuencoded file starts with **"begin "** followed by the *filemask* (which is ignored by **NetCode**) and the FileName. The encoded data follow as fixed length textlines and end with an encoded zero-length line and **"end"**. The encoded data take thus some 35% more space.

Base64 is the standard encoding scheme for binary data in MIME (Multipurpose Internet Mail Extensions - RFC 1521). It is much like UUEncoding but it uses another subset of printable characters which aren't misinterpreted by gateways using EBCDIC code.

Quoted Printable is the standard encoding scheme in MIME for formated ASCII text containing 8 Bit characters. Special characters are encoded as their hexadecimal code value preceded by an equal sign **"="**. Encoded data take thus much less space than UUEncoding or Base64 Encoding if only few characters are 8 Bit ones. It should **not** be used for binary data.

AcceptDataPREF_AcceptData
ActivePREF_Active
BytesSentPREF_BytesSent
ConnectedPREF_Connected
DataInPREF_DataIn
DataToSendPREF_DataToSend
EOLPREF_EOL
HostPREF_Host
HostAddressPREF_HostAddress
HostNamePREF_HostName
InBufferSizePREF_InBufferSize
LingerPREF_Linger
ListeningPREF_Listening
LocalHostPREF_LocalHost
LocalHostNamePREF_LocalHostName
LocalPortPREF_LocalPort
NullsToSendPREF_NullsToSend
OutBufferSizePREF_OutBufferSize
PortPREF_Port
RemoteHostPREF_RemoteHost
RemotePortPREF_RemotePort
WinsockInfoPREF_WinsockInfo
ActionPREF_Action
EncodedDataPREF_EncodedData
DecodedDataPREF_DecodedData
FileNamePREF_FileName
FileCntPREF_FileCnt
FileCntPREF_FileCnt
FormatPREF_Format
IntellicodePREF_Intellicode
MaxFileSizePREF_MaxFileSize
OverwritePREF_Overwrite
ProgressStepPREF_ProgressStep

DecodingUREF_ENCODING
EncodingUREF_ENCODING
UUDecodingUREF_UU_ENCODING
UUEncodingUREF_UU_ENCODING
Base64 DecodingUREF_BASE64_ENCODING
Base64 EncodingUREF_BASE64_ENCODING
Quoted Printable DecodingUREF_QP_ENCODING
Quoted Printable EncodingUREF_QP_ENCODING

ConnectedEREF_Connected
ConnectionRequestEREF_ConnectionRequest
DataInEREF_DataIn
DisconnectedEREF_Disconnected
ReadyToSendEREF_ReadyToSend
ProgressEREF_Progress

EncodeFREF_Encode
DecodeFREF_Decode

True
False
Boolean (Integer)

Integer
Long
String
"" *(empty string)*

".uue", **".b16"**, or **".q_p"**

Error Codes~~ERROR_CODES~~
Exported Functions~~EXPORTED_FUNCTIONS~~

NetCode

NETCODE.VBX

netcodecontrol

1.30

$30

#11967

5420

Copyright Notice

Registration Procedure

The prices below are for the ***licenses only*** and do not include media distribution. We only send you a set of keys to unlock the software and verify registration by e-mail. All technical support questions should be directed to:

INTERNET:	**devsoft@aol.com**
COMPUSERVE:	**75244,2736**

The cost of a single user developer is **$30**. You can order via any one of the following channels:

i) ordering through CompuServe Software Registration Service (SWREG)

You can register via CompuServe by going to the Shareware Registration Forum (**GO SWREG**) and following the forum instructions. The Registration ID for **NetCode** is **5420**. You can also do a keyword search using the keyword **NetCode**.

ii) ordering by Check or Money Order

To order by check or money order please send the attached order form and a check or a money order **(payments must be in US Dollars drawn on a US Bank)** to:

devSoft Inc.
P.O. Box 13821
Research Triangle Park, NC 27709 U.S.A.

iii) ordering by Credit Card

To order by Visa or MasterCard by E-mail, fax or snail mail send the attached order form to the above address or:

INTERNET:	**devsoft@aol.com**
COMPUSERVE:	**75244,2736**
FAX:	**(919) 493-5805**

Where To Find Our Shareware

The first place to look at is **http://www.dev-soft.com/devsoft .** There you will find the latest versions of our products, release notes, questions and answers, documentation, press releases, everything you would want to know about us and our products. We strongly recommend that you access that site before contacting us directly.

We will also upload our products in the CompuServe MSBASIC Forum (GO MSBASIC) in Library 17 (3rd Party Tools), as well as in America Online. Usually, the name of the product will be listed as a keyword, so if you try it, you will certainly get a hit.

We will also announce our new releases to the newsgroups of the comp.lang.basic.visual hierarchy, and **comp.lang.basic.visual.3rdparty** in particular.

Licensing

i) shareware version

You may use the shareware version of **NetCode** for up to 30 days in your design environment and for evaluation purposes only. You may copy and distribute it freely as long as all the files in the package, including the demo programs and this help file are distributed with it and no changes or additions of any kind are made to the original package.

There is no charge for any of the above, however, you are specifically prohibited from charging, or requesting donations for any copies, however made, and from distributing **NetCode** and/or it's accompanying files with other products (commercial or otherwise) without prior written permission from **devSoft Inc**.

ii) registered version

As a registered user, you can use **NetCode** in your design environment as well as distribute executables that use **NetCode** as a runtime component. **devSoft** asks for no royalties or runtime fees for such distribution. The only requirement is that you distribute a license file which will bear your unique serial number. You will obtain that file upon registration. We also ask you as a courtesy to distribute this help file with your application, but you are not required to do so.

Please note that the rights to the license file are not transferable: users of your application cannot legally use the license for their own applications, or distribute their own code using the a license file with your serial number on it. Only registered users can distribute executables using **NetCode**.

You may install only one registered copy of **NetCode** in a single workstation at any time. Use of a registered copy in more than one workstation is against the terms of this licensing agreement. In particular, you are specifically prohibited from distributing a registered version of **NetCode** except as a runtime component of one of your applications.

Limitation of liability:

THIS SOFTWARE AND THE ACCOMPANYING FILES ARE SOLD "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OF MERCHANTABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED OR IMPLIED. THE EXTENT OF LIABILITY OF THE SELLER IS HEREBY LIMITED EXCLUSIVELY TO PRODUCT REPLACEMENT OR REFUND OF PURCHASE PRICE. IN PARTICULAR, IN NO EVENT SHALL DEVSOFT BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOSS OF PROFITS, LOSS OF DATA, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM THE USE OF THIS SOFTWARE.

devSoft Inc.

P.O. Box 13821 , Research Triangle Park, NC 27709 U.S.A.

ORDER FORM / INVOICE

Prices are guaranteed through December 1995.

Registration codes will be sent by electronic mail only. If you need a disk (3.5") or a paper copy of your license, please enclose an additional $5.00 with your order.

Product	No. of Copies	Price	Total
IPPort	_____ x	$25.00 =	_____
IPDaemon	_____ x	$25.00 =	_____
UDPPort	_____ x	$25.00 =	_____
NetCode	_____ x	$30.00 =	_____
Add Disk and/or Paper copy of License		$5.00 =	_____
		Total	_____

Name: Phone: Date:

Credit card used /Exp. Date

Company Name:

Address:

City, State, Zip/Country:

Name of registrant:

E-Mail address:

Comments:

Decode Exported Function

Description

Decodes according to the format **sFormat** the data of length **ulEncLen** pointed by **lpEncoded**. The decoded data are written at the space pointed by **lpDecoded**. The caller has allocated **ucDecLen** bytes. **lpFileName,** if not NULL, receives the filename of the decoded data.

Please refer to Format for the legal values of **sFormat**.

Syntax

```
#include <vbapi.h>
ULONG FAR PASCAL UUDecode(
        SHORT sFormat,
        LPVOID lpDecoded, ULONG ulDecLen,
        LPVOID lpEncoded, ULONG ulEncLen,
        LPSTR lpFileName, ERR FAR* lpErr, BOOL bIntellicode);
```

Return Value

Length of **DecodedData** written in **lpDecoded** or **0** if an error occurred. In that case the integer pointed by **lpErr** (if the pointer is not NULL) receives the error code.

Remarks

Same as assigning **DecodeToString** to the Action, beside that the lengths of the data is limited only by ULONG's highest value: **4294967295. ulDecLen** and **ulEncLen** denote the amount of available space the pointers point at. An error occurrs if there's not enough place.

The error value is written at the short integer pointed by **lpErr** -- if not NULL. Please refer to Error Codes for a complete list of errors.

For a description of Format, DecodedData, EncodedData, FileName, and Intellicode see the recpective sections.

This function is offered for C users to overcome the limitations put on the length of Visual Basic Strings.

See Also

Encode

Encode Exported Function

Description

Encodes according to the format **sFormat** the data of length **ulDecLen** pointed by **lpDecoded** with the filename given by **lpFileName**. The encoded data are written at the space pointed by **lpEncoded**. The caller has allocated **ucEncLen** bytes.

Please refer to Format for the legal values of **sFormat**.

Syntax

```
#include <vbapi.h>
ULONG FAR PASCAL UUEncode(
        SHORT sFormat,
        LPVOID lpDecoded, ULONG ulDecLen,
        LPVOID lpEncoded, ULONG ulEncLen,
        LPSTR lpFileName, ERR FAR* lpErr, BOOL bIntellicode);
```

Return Value

Length of **EncodedData** written in **lpEncoded** or **0** if an error occurred. In that case the integer pointed by **lpErr** (if the pointer is not NULL) receives the error code.

Remarks

Same as assigning **EncodeToString** to the Action Property, beside that the lengths of the data is limited only by ULONG's highest value: **4294967295. ulDecLen** and **ulEncLen** denote the amount of available space the pointers point to. An error occurrs if there's not enough place.

The error value is written at the short integer pointed by **lpErr** - if not NULL. Please refer to Error Codes for a complete list of errors.

For a description of Format, DecodedData, EncodedData, FileName, and Intellicode see the recpective sections.

This function is offered for C users to overcome the limitations put on the length of Visual Basic Strings. Therefore, unlike in VB Calls with VB Strings, **NetCode** produces line breaks as single linefeeds **"\n"** in lpEncoded.

See Also

Decode

Properties

*Action	*Intellicode
*DecodedData	Left
*EncodedData	*MaxFileSize
*FileCnt	Name
*FileName	*Overwrite
*Format	*ProgressStep
Index	Top

Action Property

Description

Controls operation of **NetCode**.

Usage

[*form*.][*netcodecontrol*.]**Action**[= *value*]

Default Value

0 *(Idle)*

Remarks

The following table lists the possible values for the **Action** property.

Setting	Description
0 (Idle)	Default
1 (DecodeToFile)	Decode with the format given in Format: Data is read from the EncodedData file(s) and written in the filename specified in the encoded data. Multiple filenames should exist as *name000.ext*, *name001.ext* etc. and specified as **name???** at EncodedData. (The number of digits/question marks is optional). Extension's value *ext* is one of the following: **uue** for UUDecoding, **b64** for Base64 Decoding or **q_p** for Quoted Printable Decoding according to the value of Format While UUDecoding the created filename will -- by default -- be the one specified in the uuencoded data. DecodedData or, if empty, FileName can be used to override this value. If the string terminates with a backslash **"\"** it is interpreted as a directory and the filename contained in the uuencoded data is tried to be created in this directory. FileName contains thereafter the filename of the (attemptedly) created file.
2 (EncodeToFile)	Encode with the format given in Format: Data is read from the file DecodedData and is encoded in the file(s) EncodedData. The uuencoded data contain as filename the value given in FileName or, if empty, the value of DecodedData. While UUEncoding its recomended to give the full file specification in EncodedData and the filename at FileName. The current version of **NetCode** creates no message headers. If -- via Format -- a MIME standard encoding is used such as Base64 Encoding or Quoted Printable Encoding the user should fill the header values appropriately. If MaxFileSize is set and the encoded data take more

	place than MaxFileSize the data is split over several files. The user can specify in this case more than one filename by ending the filename with question marks **"?"**. **NetCode** expands them into numerals starting from 000 (as many digits as question marks specified) and appends the extension **".uue"**, **".b16"**, or **".q_p"** according to the value of Format to the filename.
3 (DecodeToString)	Decode according to Format: The string EncodedData is encoded into DecodedData. FileName contains the filename contained in the uuencoded data. No MIME Header interpretation is currently supported.
4 (EncodeToString)	Encode according to Format: The string DecodedData (may contain also binary data) is encoded into EncodedData. While UUEncoding the filename is taken from FileName.

Please refer to Error Codes for a complete list of errors.

Data Type

Integer

DecodedData Property

Description

Filename of the decoded data or the decoded data itself. Setting the Action property tells **NetCode** whether a file or a string is meant.

Usage

[*form*.][*netcodecontrol*.]**DecodedData**

Default Value

"" *(empty string)*

Remarks

While UUEncoding, the value of FileName is the filename that will be written in EncodedData. If FileName is empty, **DecodedData** is taken. It's reccomended that EncodedData should contain the full path and FileName only the filename, so that no problems occurr while UUDecoding in a foreign system.

While Decoding, **NetCode** tries to generate the name for the created file in the following order: **DecodedData**, FileName or, while UUDecoding, the specified filename in the uuencoded data itself. If **DecodedData** or FileName end with a backslash **"\"** they are interpreted as directories and **NetCode** tries to create the given filename in this directory. FileName contains thereafter the name of the (attemptedly) created file.

Data Type

String

EncodedData Property

Description

Filename of the encoded data or the encoded data itself. Setting the Action property tells **NetCode** whether a file or a string is meant. **NetCode** expects multiple filenames in the form **name???.*ext*** where *name* is the root filename, possibly with a path specification, the number of question marks **'?'** shows the format of numbers and the extension **should** be one of the followings according to the value of Format: **".uue"**, **".b16"**, or **".q_p"**.

Usage

[*form*.][*netcodecontrol*.]**EncodedData**

Default Value

"" *(empty string)*

Remarks

If the encoded data are spread over several files, these filenames should exist as *path\file000.ext* and passed to **EncodedData** in the form **name???**. Question marks **"?"** are expanded to numerals starting with 0000 (the number of zeros **"0"** equals that of question marks **"?"**). **NetCode** appends the extension **".uue"**, **".b16"**, or **".q_p"** according to Format) immediately after the generated numbers. FileCnt receives -- if not zero -- the number of encoded files treated.

See also the Format, Intellicode and MaxFileSize Properties when treating multiple files.

Data Type

String

FileCnt Property

Description

Tells -- if not 0 -- the number of encoded files **NetCode** has read from or written into. If the user specifies one or more questions marks **"?"** in EncodedData these will be expanded to 000 -- **FileCnt** - 1 (the number of question marks **"?"** specifies the number of digits.

Usage

[*form*.][*netcodecontrol*.]**FileCount**

Default Value

0

Remarks

Please refer to EncodedData for filenaming conventions.

Data Type

Integer (read only)

FileName Property

Description

While UUEncoding it contains the filename that is written in the uuencoded data.

While Decoding -- if not empty -- it instructs **NetCode** where to write the decoded data. It can be either the filename or the directory where the file should be written. In this case it should end with a backslash **"\"**

After an **Decode** Action, it contains the filename of the (attemptedly) created file. If the file couldn't be created because of an illegal filename, examining **FileName** might give hint to the reason of the failure. Remeber to set **FileName** to **""** (empty string) after each **Decode** operation since it contains the full specification of the file.

Usage

[*form*.][*netcodecontrol*.]**FileName**

Default Value

"" *(empty string)*

Remarks

An error occurrs if **FileName** is empty and the UUEncode Format as well as EncodeToString Action are selected.

If you want to first check the uuencoded filename before creating it in the disk, you may assign an illegal directory name to **FileName** (always use a closing backslash **"\"** to denote it as directory), then trap the produced error and check the filename appended to **FileName**.

Data Type

String

Format Property

Description

Shows the type of coding to be used.

Usage

[*form*.][*netcodecontrol*.]**Format**[= *value*]

Default Value

0 *(UUEncode)*

Remarks

The following table lists the possible values for the **Format** property.

Setting	Description
0 (UUEncode)	The most used (unwritten) standard. 3 Bytes are encoded into 4 readable characters. If multiple filenames are specified the extension **".uue"** is used/expected.
1 (Base64)	Encoding format of MIME. Much like UUEncode but another subset of printable characters is used. If multiple filenames are specified the extension **".b64"** is used/expected.
2 (Quoted_Printable)	Another MIME format coding only special characters. Used mostly if the text contains special accented characters. If multiple filenames are specified the extension **".q_p"** is used/expected.

Please refer to Encoding for more details on encoding schemes.

Data Type

Integer

IntelliCode Property

Description

Controls whether **NetCode** should try to interpret several concatenated messages while UUDecoding. No interpretation of MIME headers is currently made.

Usage

[*form*.][*netcodecontrol*]**IntelliCode**[= *value*]

Default Value

True

Remarks

If a file is uuencoded and split over several mail messages, **NetCode** does normally attempt to filter the opening and closing lines such as message headings and/or decode scripts which don't belong to the encoded data. You should set **IntelliCode** to **False** only in the improbable case that the data is decoded improperly.

Important: The uuencoded messages should appear in the **proper order**. The current version handles only the filtering of *packing* information and does no sorting yet.

Important: This property currently applies only to UUEncode Format. The MIME headers should be treated by the user.

Data Type

Boolean (Integer)

MaxFileSize Property

Description

Controls while Decoding whether **NetCode** should split the encoded data into several files and gives the maximum allowed size for these files.

Usage

[*form*.][*netcodecontrol*.]**MaxFileSize**[= *value*]

Default Value

0 *(no splitting)*

Remarks

Some mailing systems require that the message size shouldn't exceed a certain size. Assigning a nonzero value to **MaxFileSize** tells **NetCode** to split the encoded data into several files. FileCnt will contain, if not zero, the number of files created. Please refer to EncodedData for filenaming conventions.

Important: If a non-zero value is assigned to MaxFileSize then a multiple filename should also be assigned to EncodedData (end with question marks **"?"**) so that **NetCode** can expand the filenames.

Please refer to Error Codes for a complete list of possible errors**.**

Data Type

Long *(0 or larger than 200)*

Overwrite Property

Description

Controls whether created file(s) should overwrite already existing ones.

Usage

[*form*.][*netcodecontrol*.]**Overwrite**[= *value*]

Default Value

False

Remarks

None.

Data Type

Boolean (Integer)

ProgressStep Property

Description

Controls the granularity at which the Progress Event is fired.

Usage

[*form*.][*netcodecontrol*.]**ProgressStep**[= *value*]

Default Value

1

Remarks

The Progress Event will be fired when 0%, n***ProgressStep**% and 100% of input data is read if **ProgressStep** is set. The Progress Event is fired exactly (100 + **ProgressStep** -- 1)\ ProgressStep + 1 times.

Setting **ProgressStep** to 0 desables firing Progress Event.

Data Type

Integer *(0 to 100 enumerated)*

Events

*Progress

Exported functions

Encode

Decode

Progress Event

Description

Occurrs when *PercentDone* of the input is read.

Syntax

Sub *netcodecontrol*_**Progress(***PercentDone* **As Integer)**

Remarks

The ProgressStep property shows how often the event is fired.

Events are fired at 0%, 100% and at multiples of ProgressStep. The event will be fired exactly (100 + ProgressStep -- 1) \ ProgressStep + 1 times if ProgressStep is not zero.

Error Codes

The following is a list of the trappable errors fired by NetCode**:**

20001 (NTCERR_BEGIN_NOT_FOUND)	The starting **"begin "** was not found (only while UUDecoding)
20002 (NTCERR_SHORT_FILE)	The input ended unexpectedly
20003 (NTCERR_NO_END)	The closing **"end"** was not found (uudecoded file may be too short -- only while UUDecoding)
20004 (NTCERR_FILE_CREATE)	Can't create a file (either illegal filename or disk is write-protected)
20005 (NTCERR_FILE_OPEN)	Can't open for read the input file (file doesn't exist?)
20006 (NTCERR_FILE_READ)	Can't read from input file
20007 (NTCERR_FILE_WRITE)	Can't write to file (disk full?)
20008 (NTCERR_NO_FILENAME)	No filename was given while encoding
20009 (NTCERR_FILE_EXISTS)	File exists and Overwrite was set to **False**
20010 (NTCERR_NOT_ENOUGH_SPACE_IN_STRING)	The given pointer had not enought space to contain the output (only when using the Exported Functions)
20012 (NTCERR_NO_ENC_FILE)	No filename was given where to write the encoded data
20017 (NTCERR_NO_SUCH_FILENAME)	No such filename
20018 (NTCERR_NO_MORE_FILES)	No more filenames where to read from or write to