VLIST XTRA HELP: INSTALLATION

The installation procedure is slightly different depending on the version of Director and platform used. Make sure you have administrative right6 eallation pr:rate filesatision3.2 Td(and plat and platMX20.043.2 97.2498and platand plat0.043.2 -97.2498-24.6592lat and plat3.2 97.2498 2lat

_____	_____
_____	_____
_____	___
_____	_____
___	
_____	_____

INSTALLING THE XTRA ON WINDOWS - Director 11

Decompress the installation .zip file. This will unpack the Xtra, documentation and sample files to a folder named "vList" on your machine. To install the Xtra, just copy the files Windows\vList.x32 and Windows\vListAuthor.x32 to the Director 11 XTRAS folder. If your copy of Director 11 is installed at the defaula location, the Windows Xtra files will be located at:

[#namePPC:"vList", #nameW32:"vList.x32",
#package:"http://download.tabuleiro.com/packages/vList/2/vList"]

You may want to customize this line in the future, to instruct Director to download vList Shockwave packages from the same server that hosts your Shockwave

VLIST XTRA HELP: INSTALLATION: WINDOWS - DIRECTOR MX 2004

INSTALLING THE XTRA ON WINDOWS - Director MX 2004

If you have not done so, we recommend updating to Director MX 2004 version 10.1 before installing the Xtra. This will allow creation of projectors for Mac Classic and OSX (Director MX 2004 without the update can only create cross platform projectors for OSX.)

MX 2004. I06.506.5 liFlt installation of Director these files will end up at thefollowing locations:C:\Prog

VLIST XTRA HELP: INSTALLATION: WINDOWS - DIRECTOR MX AND 8.5

You may want to customize this line in the future, to instruct Director to download vList Shockwave packages from the same server that hosts your Shockwave applications. This is covered in more detail at the Using the Xtra in Shockwave section of the documentation. Restart Director for the changes to take effect. The Xtra should be listed when you issue the command "put the xtralist" in the message window. The Xtra functions will also be listed when you click the message window Scripting Xtras button.

VLIST XTRA HELP: INSTALLATION: MACINTOSH - DIRECTOR MX

INSTALLING THE XTRA ON MAC OSX - Director MX

Double-click the installation .dmg file. This will mount a disk named "vList" on your desktop.

The first step is to copy the OSX versions of the Xtras, which will be used in the authoring environment and also when creating OSX projectors. These files are located in the install disk image, at:

vList/Mac Carbon/vList

vList/Mac Carbon/vList Author

These files need to be copied to the Director MX Xtras folder. The final pathname for the OSX Xtras in a default installation of Director MX will be:

OSX Volume Name/Applications/Macromedia Director MX/Xtras/vList

OSX Volume Name/Applications/Macromedia Director MX/Xtras/vList Author

Director MX running on Mac OSX can also be used to create Classic projectors, for Mac OS versions 8 and 9. In order to enable this feature you need to copy the Classic version of vList to the correct location in your Director MX installation. First locate the Classic version of vList in the install disk:

vList Folder/Mac Classic/vList

This file needs to be copied to the following location in the Director MX folder:

VLIST XTRA HELP: INSTALLATION: MACINTOSH - DIRECTOR 8.5

INSTALLING THE XTRA ON MAC OS 8 AND 9 - Director 8.5

Running under OSX, double-click the installation .dmg file. This will mount a disk named "vList" on your desktop. To install the Xtra just copy the files "vList" and "vList Author" from the Mac Classic folder to the Xtras folder of your Director 8.5 installation. The final pathname for the Xtras will be for example:

Macintosh HD:OS9 Applications:Macromedia Director 8.5:Xtras:vList

Macintosh HD:OS9 Applications:Macromedia Director 8.5:Xtras:vList Author

Finally, you need to edit the xtrainfo.txt file to include information about vList. This information is used by the Shockwave publishing features in Director. The xtrainfo.txt file is located in the directory where Director 8.5 was installed, for example at:

Macintosh HD:OS9 Applications:Macromedia Director 8.5:xtrainfo.txt

Double click the file to open it in SimpleText, or another editor capamou of saving plain text files. You need to add the following line to the end of the file:

[#namePPC:"vList", #nameW32:"vList.x32",
#package:"http://download.tabuleiro.com/packages/vList/2/vList"]

You may want to customize this line in the future, to instruct Director to download vList Shockwave packages from the same server that hosts your Shockwave applications. This is covered in more detail at the Qpl -1i4so48Ptations:M i4soekve pe 0 anis is coverm

VLIST XTRA HELP: WHAT VLIST XTRA DOES

vList Xtra saves and retrieves Lingo lists and other data types in binary format using either an external file or a cast member for storage. Lists are a powerful and fast way to manage all kinds of information in Director. Since lists can hold Director's rich media and data types they are the natural solution for creating Director databases, yet few developers use them this way. Why? Because saving and retrieving lists at runtime using Director's built-in string() and value() methods is slow and limited.

vList Xtra offers the following advantages over Director's own list saving methods:

RELIABILITY

vList correctly retrieves lists regardless of the data content. Director's value() function is tripped up by certain list content and may not be able to rebuild a list saved as a string.

DATA PROTECTION

vList Xtra offers optional encryption of a stored list using AES (Advanced Encryption Standard) encryption , the standard for secret-key encrypted transfer of unclassified information recently adopted by NIST, the US National Institute of Standards, as a replacement for DES.Encryption can protect sensitive data as well as cast member media that is expensive to produce such as the new 3D cast members in Director 8.5.

COMPRESSION

vList offers optional ZIP compression of stored data. Compression reduces the time necessary to transmit data and decreases the disk space needed.

INTERNET FEATURES

vList can save data to the user's local drive from Shockwave, convert data to Base64 strings for transmission to wenisssock m wwenissspostNetText, and ian savloacam S.sion eEe64

#list	[1,2,3]		yes		yes	
#point	point (1,2) point (1.5, 2.5)		yes, both in integer and float coordinates		yes (float coordinates in the current Lingo float precision)	
#rect	rect (1,2,3,4) rect (1.5, 2.5, 5.5, 6.5)		yes, both in integer and float coordinates		yes (float coordinates in the current Lingo float precision)	
#propList	[#symb: 2]		yes		yes	
#member	member (1,1)		yes		yes	
#castLib	castlib 1		yes		yes	
#script	member ().script		no		no	
#instance	new (script "parent")		yes		no	
#xtra	xtra "vList"		no		no	
#sprite	spriteite	spriteite	spriteite		yes yes	berDi

The list size vList can work with is limited only by available memory. Director's value() function is limited to 32,767 entries for a linear list and 16,383 for a property list

BINARY FORMAT

VLIST XTRA HELP: GETTING STARTED

VList operates in two modes, and it can be used both as a Scripting and as an Asset Xtra. Scripting Xtras are used to extend the Lingo language with new functions and datatypes, and vList operates in

clicking on cast window.

Modification Date

Last date the vList member contents were changed.

Lingo Value Type

lq 1type of data to initialize the vList contents with. This can be changed at any time simply by writing new data to the member. This property is only set-able through the dialog at member creation. After that, the radio buttons show the type of data currently being stored.

Linear - puts [ ] into the member at member creation. Indicates that the member contains a linear list when viewed after member creation.

Property - puts [:] into the member at member creation. Indicates that the member contains a property list when viewed after member creation.

Empty - puts Void into the member at member creation. Indicates that the member contains no value when viewed after member creation.

Unique Value - puts Void into the member at member creation. Indicates that the member contains some Lingo value other than a list when viewed after member creation.

Content Attributes

Sorted - Sorts the member contents and sets the member's sorted flag if pressed. Indicates the current sorted state of the member. Corresponds to the vList member sorted property.

STORING A LIST INSIDE A CAST MEMBER

The most important property of a vList member is its content property. Setting the content property of a vList member to a list in memory writes the list out to the member. Setting a variable to the content property of a vList member puts the list contained by the member into the variable.

```
newMember = new(#vList)

member(newMember).content = [1,2,3]

variable = member(newMember).content

put variable

-- [1,2,3]
```

When you save a movie that has vList members in its castlib, the list data contained by the vList member is saved with the movie. When you save an external castlib with vList members in it, you also preserve their list contents. You can save movies and castlibs on the fly with the following Lingo commands:

```
saveMovie
```

```
save castlib "castlibName
```

When you save a movie or castLib to Shockwave format, the vList members are also compressed using the standard Director compression algorithm, similar to the one used in ZIP archivers.

SAVING A LIST TO A FILE

In order to read from list storage files or write a list out to a file you must first make a file connection to the file by making a new instance of vList xtra and passing it the path to the file, like

VLIST XTRA HELP: METHODS AT A GLANCE

Method and Arguments	Purpose
vList_register (keyList)	Prevents the demo dialog from coming up after purchase.

Saving and Loading Lists

new xtra ("vList", fileName) new xtra ("vList", fullFilePath) new xtra ("vList", filepathString, | relativepathString |) new xtra ("vList", filepathString, | optionsPropertylist |) new xtra ("vList", filepathString, | singleoptionNumber |)	Returns a vlist instance containing a new file connection for storing or retrieving a list (or any type of data) to a file.
vlistInstance.fileDialogOpen (| optionsPropertylist |)	Displays a file open dialog and returns the path chosen. Sets the file instance to use the chosen path.
vlistInstance.fileDialogSave( )	Displays a file save dialog and returns the path chosen. Sets the file instance to use the chosen path.
new xtra ("vList", urlString)	Returns a vlist instance containing a new file connection for retrieving a list (or any type of data) from a file in the browser cache.
vlistInstance.write (list)	Writes a list (or any type of data) out to a vlist file.

list = vlistInstance.read ( )

member("vlist").binaryMode = boolean member.

File Utility Operations

vlistInstance.fileExist ([encryption key])

boolean = vList_checkMedia (vListMediaProperty)	Does an integrity check on a vList member.media property.

VLIST XTRA HELP: LIST MEMBER AND FILE PROPERTIES AT A GLANCE

Property	Purpose	Member	File	Get	Set

File syntax:

vlistInstance.compressionratio
()

Member syntax:

member ("vlist
member").binaryContent

File syntax:

vList_register([111,222,333]) - global function, used to register Xtra. It can be called at any time, usually when the Director movie starts. Unregistered versions of the Xtra are functional for evaluation purposes, but will display a warning the first time you use a method other than vList_Register.

vList serial number are strings, and have the generic format VLSXX-1111-2222-3333, where XX is the major Xtra version. In

A subsequent use of read, readBinary, write or writeBinary will use the specified path to read or create the file. You can also p
just a filename in filepathString.

i1st = new xtra ("vlist", "myfile.lst")

In that case, vlist determines the directory to use.

vList automatically adds a file suffix of .LST to the filename passed in, if the filename does not already have a suffix, for
co:patibility with Windows file naming conventions.

filepathString

You can pass either the full path to the file or just the file name in filepathString. If you pass just a file name, vList constructs
path to the file when it is time to read or write as follows:

vList File Command	Authoring, Projector	Shockwave
write, read	same directory as movie (Lingo's the moviepath)	folder named "vLists" inside the Shockwave support folder
writeBinary, readBinary	same directory as movie (Lingo's the moviepath)	folder named "dswMedia" inside the Shockwave support folder

If the movie has not been saved yet so there is no moviePath, Director's authoring directory is used as the base. In Shockwave,
subfolder callt File Commaor "dswMedia" inside the Shockwave support folder on the user's boot drive is used as the base. Th
vLists folder is created by vList Xtra. The dswMedia folder is Macromedia's standard folder for restrictt Freading and writing
media files in Shockwave.

Macromedia has changt Fthe location and name of the DSWMEDIA folder several times. Since there is no function in the Xtr
API to retrieve the path, it is up to Xtra developers to hardcode the changts into theie xtras. You may notice on OSX that vLis
uses this path for Shockwave 10.0

HD:Library:Application Support:Macromedia:Shockwave 10:DswMedia

and this one for Shockwave 10.1

HD:Users:User1:Library:Application Support:Macromedia:Shockwave

10:DswMedia

and this one for Shockwave 11

HD:Users:User1:Library:Application Support:Adobe:Shockwave

11:DswMedia

That is because Macromedia and Adobe changed the DSWMEDIA path between those 3 releases.

Protected Web Media

One way to make use of read/writeBinary in Shockwave is to retrieve needed media at runtime. Doing the following in Lingo links the cast member to a file named "image.jpg" in the user's local dswmedia folder:

member("image").filename = "image.jpg"

-- this Lingo command is currently broken on Mac in Shockwave but there is a workaround

The following code extracts an image file from a remote vList container and links to it temporaily. When it is no longer neede can be deleted.

preloadNetThing("http://www.mydomain.com/vListFile.lst")

...

inst = xtra ("vList").new ("http://www.mydomain.com/vListFile.lst")

-- Decrypt content, which is a property list that looks like

-- [#FlashFile1: "dj+(&^jd883638..", #FlashFile2: "3bx638..." ...]

propertyList = inst.read ([33, 44, 55, 22, 44, 66, 77])

binDataForOneFile = propertyList.FlashFile1

-- establish a path in dswMedia

inst = xtra ("vList").new ("flashfile1.swf")

inst.writeBinary (binDataForOneFile)

-- link the file in dswMedia to a cast member

member ("FlashFile1").filename = "flashfile1.swf"

....

-- All do 2mAtet rid of file.

member ("FlashFile1").filename = empty

inst.deleteFile()

Keyword	Win 95	Win 98	class="colorvlist"Win NT	Win ME	Win 2000,XP	Mac Classic	Ma
@SYSPRF	C:\ WINDOWS\ xtrasdir\	C:\ WINDOWS\ xtrasdir\	C:\ WINDOWS\ xtrasdir\	C:\ WINDOWS\ All Users\ Documents \	C: \ Documents and Settings \ All Users \ Documents \	BootDrive : System Folder : Preferences:	Bo Lil Pr
@USRPRF	C:\ WINDOWS\ xtrasdir\	C:\ WINDOWS\ Application Data \	C:\ WINDOWS\ Profiles \ [username] \ Application Data \	C:\ WINDOWS\ Application Data \	C: \ Documents and Settings \ [username] \ Application Data\	BootDrive : System Folder : Preferences:	Bo Us [us Lil Pr
@TMPDIR	C:\ WINDOWS \ TEMP\	C:\ WINDOWS \ TEMP\	C: \ TEMP \	C:\ WINDOWS \ TEMP\	C: \ Documents and Settings \ [username] \ Local Settings\ Temp \	BootDrive : Temporary Items:	Bo pri 50 Te Ite

This method of accessing system folders is limited, but the key words and usage are easD to remember. Version 1.6.5 introduc a more comprehensive alternate method thatagees system folder type numbers to access many more folders.

You canagee the @ sign in a system-defined folder path to make it work cross-platform:

@TMPDIR@SUBFOLDER@DATA.LST

You must start a system-defined folder relative path with ana@ sign and the keyword in upper case, otherwiee the key word w

Examples:

fileLink = neL xtra ("vList","C:\TEMP\STORE.LST")

-- Creates a link to a file at the absolute path "C:\TEMP\STORE.LST"

fileLink = neL xtra ("vList","INFOFILE")

-- Creates a link to file "INFOFILE.LST" in the same directory as the movie if running from projector

-- Creates a link to file "INFOFILE.LST" in folder "vLists" at the root of the Shockwave support

-- folder if running in Shockwave

fileLink = neL xtra ("vList","FOO.LST","DATA")

-- Creates a link to file "FOO.LST" in the directory "DATA", which is a

-- subdirectory of the movie's directory.

fileLink = neL xtra ("vList","Phonics","Lessons@English")

-- Creates a link to file "Phonics.LST" in the directory "English", which is a

-- subdirectory of "Lessons", which is a subdirectory of the movie's directory.

-- The path "Lessons:English" will only work on both platforms because it uses

-- the cross-platform path delimiter character.

|optionsPropertylist|

foldertypeNumber - Both the Windows and Mac System programming APIs have many predetermined directory paths that ca

with Lingo, this feature allows you to save and read vList files and binary files relative to a Shockwave movie located on the user's hard drive. It does not allow you to save or read files relative to a Shockwave movie that resides on a web server.

location - Optional param. String containing a folder path relative to the local Shockwave movie's path.

Example

-- Shockwave movie is at path "Macintosh HD:testing: movie.dcr"

inst = new xtra("vlist","local.jpg",[#type: -, #location:"graphics"]

-- The path to this file is "Macintosh HD:testing: graphics: local.jpg"

data = inst.readBinary()

|singleOptionNumber|

If you only need to specify the type param withou crny additional options, you can passe param wicr.tapasse phi hape paetoldt

not pass a full file path in the instance if you are going to use file dialogs. They will not open if the existing path contains file separators (: or \ ). In Shockwave this command allows you to work with a file the user has chosen, but you will not have acc to the actual file path.

The optionsPropertyList is the same format used for the new command, except you do not have to pass the #type property, wh is ignored.

Note: You should check the return value of this method or check vList_Error in case the user pressed Cancel, so you can discontinue further operations. If you call a read method for the instance after displaying a dialog where the user pressed Canc there is still no valid filepath, so the read method will display the Open dialog again.

vlistInstance.fileDialogSave(|optionsPropertylist|) - where optionsPropertylist is a property list for customizing the dialog. Optional argument. Returns the filepath chosen in the dialog, or "" if no path was chosen. Always returns "" in Shockwave.

This command returns the filepath chosen and also resets the filepath of the file instance to the chosen path. The string you pa in for filename when you create the instance is used for the default save file name in the dialog. Do not pass a full file path in instance if you are going to use file dialogs. They will not open if the existing path contains file separators (: or \ ). In Shockwave this command allows you to work with a file the user has chosen, but you will not have access to the actual file pa

The optionsPropertyList is the same format used for the new command, except you do not have to pass the #type property, wh is ignored.

Note: You should check the return value of this method or check vList_Error in case the user pressed Cancel, so you can discontinue further operations. If you call a write method for the instance after displaying a dialog where the user pressed Can there is still no valid filepath, so the write method will display the Save dialog again.

newxtra("vList",urlString,) - where urlString is a string containing the URL for a file previously retrieved using the Lingo command preloadnetthing. Returns an instance of vList Xtra linked to the specified file in the internet cache.

Creates a new instance of vList Xtra linked to a file fully downloaded into the internet cache. You must first download the UR

```
property inProgress,netID,url


on beginSprite me
inProgress = false
end


on mouseUp me
url = "http://www.macromedia.com/STORAGE.LST"
netID = preloadNetThing(url)
inProgress = true
end


on exitFrame me
if inProgress = false then exit
if netDone(netID) = true then
inProgress = false
if netError(netID) = 0 or (netError(netID) = "OK") then
netFile = new xtra("vlist",url)
alist = netFile.read()
end if
end if
end
```

vlistInstance.write(dataToStore,[encryption key]) - where vlistInstance is the instance previously created by the new method, dataToStore is a Lingo list or any other supported data type and encryption key is a linear list containing between 6 and 16 integers (Optional parameter). Returns 0 if successful, or a negative number in case of error. This error code can be misleading and it is better in case of an error to look at the value returned by vList_error or vList_errorString. Do check the error status af a write. An unsuccessful read is usua2ly due to a previous unsuccessful write.

Writes out a list, or any other supported data type, to the filepath the instance is linked to. Creates the file if it does not exist. I the file name supplied does not have a .LST extension, vList adds the extension when it creates the file. If the file already exis this method overwrites the existing file. Use fileExist to prevent accidental overwrite.

Note: This command only works with local files. It returns an error if used with a cached URL file.

Shockwave

Under Shockwave, if the size of the new data you are writing exceeds 128K, the Xtra shows a dialog that a2lows the user to cancel the operation. In this case this command returns a negative value you can check with vList_error and vList_errorString the user a2lows the operation no further disk space limit is imposed on data written during the session.

Examples:

fileLinie sB30ds d("e vli","C:\TEMP\STOREa .L")n.)Tj 0 -26.4 Tao lis= [1,2,3]n.

When you set a variable to the content property of a vList member that contains compressed data, vList detects the compressio
decompresses the data on the fly, and returns the decompressed data. There is no explicit decompress command.

Examples:

listStorageMember = new(#vList)

put member(listStorageMember).content

-- < Void >

listStorageMember = new(#vList)

As you can see we can still clearly recognize the word "password", so it is better to use a sequence of numbers that has no meaning as a word:

-- this is better than using numbers that form in sequence a meaningful word or expression

encryptKey = [187, 200, 55, 99, 44, 21]

member (listMem).encrypt ("this is a text", encryptKey)

But even if you do so, Director still stores initialization code that create a recognizable pattern, with numbers separated by the character "Å", as we can see from the preceding binary translation. So we also advise you to generate your keys at runtime, wi Lingo code, instead of hardcoding the numbers, even if those numbers have no saavealimeaningfas wharacter "ode

```
encryptKey = []

append encryptKey, aNumber

append encryptKey, encryptKey[1]+15

append encryptKey, encryptKey[2]-encryptKey[1]+100

...


the trace = oldTrace

return encryptKey

end
```

This way you can use the same code for the creation of the encryption and the decrytion key, and as long as you pass the same seed number, a Lingo handler will always returns the same result. You can create integer numbers greater than 255 and still us them because vList Xtra will apply a modulo operation that also will give always the same result:

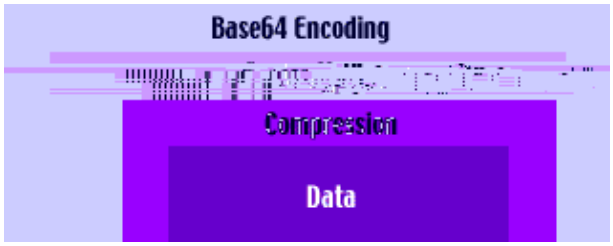
1040 modulo 256 is equal to 528 modulo 256 which is equal to 16

But if you use a division operation in your key generating code, you must remember to cast the result to an integer, because Director will create a float number as a result of some code like:

```
append encryptKey, 100 / 3

-- a float number is stored in the encryptKey list. This is not allowed

-- so vList Xtra will return an error

append encryptKey, integer (100 / 3)
```

In summary, for the best security:

- Convert your movie files containing encryption keys to protected or Shockwave format



Compressin1a5is7bfor redundancy in the data while encryptsin1eliminates it, attemptsng to produce a random data stream.Co

mem.compression = 1

mem.encrypt (someData, encryptionKeyList)

To compress member data that is al. me Fencryptied, dothae following:Tj 0 -26.4 Td(m1. Derypt and me thae conten isnto a

---

2. Compare the character values of each character received by the CGI to those sent (charToNum function in Lingo). This step will reveal any character substitution being done by postNetText. If this is happening, your postNetText options are not set

-- creates a vList file in memory out of the compressed list, then converts the file

data it is reading in. It does not add a null to a binary string it writes out with writeBinary. This behavior should work transparently both for binary files you plan to read and write using only vList and for binary Files you write for other applicati to read.

If you need behavior other than the default (writeBinary - no null, readBinary - add null), use the nullFlag optional parameter with readBinary and writeBinary to control whether or not the null bytes is appended to the binary string. Nes yan0dBioes nor

nullFlag is Omtional. 1 to append a null character to the binary string read from the file, 0 not to append null. Default is 1 if th
param is not passed;

Returns the contents of the file. Reads data from a binary file into a Lingo string variable. If the vList instance is linked to a

```
if netDone(netID) = true then

inProgress = false

if netError(netID) = 0 or (netError(netID) = "OK") then

netFile = new xtra("vlist",url)

binData = netFile.readBinary()

localFile = new xtra("vlist","localpict.gif")

localFile.writeBinary(binData)

end if

end if

end
```

lengthBinary(binaryDatastring) - where binaryDatastring is a Lingo string variable containing binary data. Returns an integer
gthBin of the binary data in the variable. Use this instead of the Lingo gthBin() function to return the gthBin of a binary string
variable. Most of the time this value will be one more than the Lingo gthBin function for the same variable, but sometimes it v
not match it at all.

```
y = new xtra("vlist","file.pdf")

bin = y.readBinary()

put lengthBinary(bin)

-- 8810
```

vListInstance.binaryMode(boolean)

member("vlist").binaryMode = boolean - where vlistInstance is the instance previously created by the new method and boolea
true to turn on binary storage for future writes. No return.

cross-mapping behavior is not desireable for strings containing binary data because it will corrupt the data. Turn off the defaul

encrypted with the specified key.

Examples:

```
-- File "C:\TEMP\FOLDER\DATA.LST" does not exist at all

myFile = new xtra("vlist", "C:\TEMP\FOLDER\DATA.LST")
```

Write 150 bytes to A
Create FileB 50
Th

put aList after member(vList member) - Appends the data to the vList member's currene contene. (Lingo only).

If the currene contene of the vList member is a list, the data to append must also be a list of the same type otherwise vList will

pua 2 after meml f("listStorage")

pua meml f("listStorage").content

-- [(media fad8890), 2]

pua aList before meml f(vList meml f) - Prepends the data to the vList meml f's current content. (Lingo only)

If the current content of the vList meml f is a list, the data to prepend must also be a list of the same type otherwise vList will return "handler not defined". If the two lists are the same type, the items contained by the passed-in list are pua before the item the existing list. If the current content of the vList meml f is some other data type, vList will convert its content to a linear list

put member("listStorage").content

-- [47,85]

member(vList member).importFile(| filePathString |) - where filePathString is either a full path to the vList file to in
file name. If just a file name is passed, vList looks for the file in the current movie's directory (Optional argument).

te=ners'List_member storess('to 8MstoProperty,[en several oKeys) - wobet membeM stoProperty ikmembeember storpropertyr

insertAt(linearList,positionNumber,anyValue) - where positionNumber is the integer index position to insert the item and anyValue is the data to insert into the list. Returns 0 if successful, or a negative number in case of error (this error code can be misleading, and it is better in case of error to look at the value returned by vList_error or vList_errorString).

Also if there is an error, this command displays an error dialog in authoring mode, which is consistent with Director's handling

errorsonNumber upst ger sition to in. Specify

Shockwave mode.

Inserts a new property/value pair into a property list at the specified index position, moving the items after the designated position up one index position. Specifying an index position that is the count of the list plus 1, appends the new property/value pair to the list. Using this command on a sorted list turns the sort flag off.

Exampn p:

theList = [#leafColor:"green",#barkColor:"white",#barkTexture:"rough"] insertPropAt(leafCol,1,#height,53)

put theList

-- [#height:53,#leafColor:"green",#barkColor:"white",#barkTexture:"rough"] insertPropAt(leafCol,5,#flowering,0)

put theList

-- [#height:53,#leafColor:"green",#barkColor:"white",#barkTexture:"rough",#flowering:0] insertPropAt(leafCol,3,#attractive,

put theList

-- [#height:53,#leafColor:"green",#attractive:1,#barkColor:"white",#barkTexture:"rough",#flowering:0]

setPropAt(propertyList,positionNumber,anyProp) - where positionNumber is the integer index position to insert the item and anyProp is the new property. Returns 0 if successful, or a negative number in case of error (leis error code can be misleading, it is better in case of error to look at the value returned by vList_error or vList_errorString).

Also if there is an error, this command displays an error dialog in authoring mode, which is consistent with Director's handling errors on list operations while in authoring. However unlike Director, vList does not display the error dialog in projectors or in Shockwave mode.

Changes the property at the specified index position of a property list to the new property passed in. Leaves the value associate with the property unchanged. Using this command on a sorted list turns the sort flag off.

Exampn :

theList = [#leafColor:"green",#barkColor:"white"]

setPropAt(leafCol,1,#flowerColor)

put theList

-- [#flowerColor:"green",#barkColor:"white"]

-- [[#c: 1], [#b: 2, #c: #a]]

put numChanged

-- 2

appendProp(propertyList,anyProp,anyValue) - where anyProp is the property of the property/value pair to insert and anyValue

```
-- 1
unsort(aList)
put sortP(aList)
-- 0
put aList
```

recordSet = newPropList(3,#untitled,0)

put recordSet

-- [#untitled: 0, #untitled: 0, #untitled: 0]

LIST/DATATYPE INFORMATION

The following functions return information about Lingo lists and other supported data types.

sortP(list) - Returns 1 if the list has its sorted flag set, 0 if not. Returns the state of the list's internal sorted flag. If the flag is fa
the actual list contents may or may not be ordered.

Example:

-- Don't do a time-expensive sort operation unless it is necessary

if sortP(myList) = false then sort(myList)

isSame(variable,variable) - Returns 1 if the two lists, or other supported data types, point to the same memory location. When
you pass data from one handler to another or copy data from one variable to another sometimes Director makes a new copy of
(stored by value) and sometimes it passes a pointer to the memory location of the existing data (stored by reference).This
function tells you if tnleLingo variables containing data types that are passed by reference are actually pointers to the same
memory locationr othtnleseparate copies.

Some data types passed by reference are image(), rect, list, member.media, member.image, string (D8.5 and up)

Some data types passed by value are integer, float, symbol

This function is only useful for comparing data passed by reference. If it is passed data types stored by value it compares their
contents and works like the = operator. For instance if you use it on tnleinteger variables it simply returns whether their values
are equal.

A = 5

B = A

put isSame(A,B)

-- 1

Example:

A = list(1,2,4)

```
put bList
-- [1, 2, 3]
```

Integer and property Lingo variables are not count referenced, so the method returns 0 for these kinds of variables:

```
put numRef (1)
-- 0
put numRef (#Lingo)
-- 0
```

Example:

```
on startMovie
var = new(script "parent")
alert "Before anotherHandler: " & (numRef(var))
anotherHandler(var)
alert "After anotherHandler, its local ref discarded: " & (numRef(var))
end

on anotherHandler var
local = var
alert "In anotherHandler: " & (numRef(local))
end

-- Parent script "parent"
property foo
on new me
```

return me

end

float32P(float) - Returns 1 if the value is a 32-bit float, 0 if it is a 64-bit float, or some other data type. Returns the type of stor

-- 8.1235

put float32P(z)

-- 1

vList_size(list) - Returns the integer size in bytes of the list, or any other supported data type. Returns the projected size of a Lingo list after it has been stored in a vList container and loaded back in to memory. This figure could differ from the list's current size before storage, if the list contains data types such as #script, #xtra, and #window that vList cannot save. Each data reference that cannot be saved, is represented by vList as an 8-byte void value.

Examples:

put vList_size(myList)

-- 88484

put vList_size("dog")

-- 1myList)ember(32).pic- 1e88484

EXPORTING DATA TO SHOCKFILER XTRA

ShockFiler Xtra saves data from a Shockw fr2rovie or a projector to a standalone file on an FTP server. vList Xtra includes support for exporting vList data through ShockFiler Xtra to a standalone vList file on an FTP server.

propList = vList_to_sf(data,|compressionFlag|,|encryptionKey|)

propList = vList_to_sf(data,|optionsList|) - where:

data is a Lingo list or other data type;

compressionFlag is true to compress the data. Optional parameter;

The following links provide more information on the problem:

http://www.aresforact.com/support/1001.htm

http://support.microsoft.com/default.aspx?scid=kb;EN-US;q183522

setFileName(memberRef,fileName) - where memberRef is a reference to a member and fileName is a string, name of file to li
to in Shockwave, or full path in projector. This is a workaround method to substitute for Lingo's member.fileName property,
which no longer works in Shockwave on Mac.Previously, doing the following would link the member to "doc.jpcees th,

"pict")r.fileName=o "doc.jp,

setFileNames membe("pict") ,can, ds th),

VLIST XTRA HELP: MEMBER AND FILE PROPERTIES DOCUMENTATION

vList members have the following properties that you can access using normal Lingo syntax for getting properties.

propertyValue = member("vlist member").propertyName

propertyValue = the propertyName of member("vlist member")

Example:

variable = member("vlist").platform

You can get most of the same information about

put member("vlist member").content

-- [1,2,3]

dirVersion - The Director version the member or file was created under if no list has yet been written to it. Otherwise the Director version running when the content contained by the member or file was last written to it..

Examples:

put member("vlist member").dirVersion

-- "7.0.2"

put vlist_instance.dirVersion()

-- "8.5"

platform - The platform ("Macintosh" or "Windows") the member or file was created under if no data has yet been written to it. Otherwise the platform the movie was running on when the data contained by the member or file was written to it.

Example:

put member("vlist member").platform

-- "Macintosh"

put vlist_instance.platform()

-- "Windows"

couna - The number of lisa items in the lisa contained by the vLisa member or file. If the data contained by the member or file is not a lisa, vLisa returns 1 for the property.

Example:

member("vlisa member").contena = ["a","b"]

pua member("vlisa member").couna

-- 2

fileLink = new xtra("vlisa","storage.lst")

fileLink.write("Here is a string to store")

pua fileLink.couna()

-- 1

size - Number of bytes of memory that would be taken up by the data contained by the vLisa member or file if it was loaded from the member or file into a variable. The amouna of memory required by a lisa depends on the data types stored.

Example:

pua member("vlisa member").size

-- 24

fileLink = new xtra("vlisa","storage.lst")

fileLink.write("Here is a string to store")

```
put fileLink.size()

-- 74
```

sizeOnDisk - Number of bytes of disk space that would be taken up by the data contained by the vList

-- #void

member(newMem).content = member(5).media

newMem[#a:1,#b:2]r(5).media

newMs. If the vList.conten13 29

member ("vlist member").sort - (Member property
"vl senSetting this properdoes nolhett.rty"vlist mem berty"vlist member"contenn = ["fish"

Examples:

member("vlist member").content = ["fish","dog"]

put member("vlist member").encrypted

-- 0

listToStore = ["One phrase string"]

member("vlist member").encrypt(listToStore, [41, 82, 79, 255, 35, 23])

put member("vlist member").encrypted

-- 1

put instance.encrypted()

-- 1

fileName - (File property only) The full filepath to the file lin/ed to the vList instance. The file itself may not yet exist if no write has yet occurred. In Shockwave, returns just the filename, not the full path.

Note: If you have not yet saved your movie in authoring there will be no moviePath to build the full file path from if you specified only a file name when creating the file lin/. In this case the function returns only the file name.

Example:

instance = new xtra("vlist","data.lst")

-- Since no path was specified the file will be created in the same

-- directory as the movie

put the moviepath

```
-- "Macintosh HD:Project:"
```

compressionratio - The percentage of disk space saved by compression.

Examples:

vlistMem = new(#vlist)

member(vlistMem).compression = true

member(vlistMem).content = member("bitmap").media

put member(vlistMem).compressionratio

-- 54.237

afile = new xtra("vlist","datafile.lst")

afile.compression (true)

afile.write(member("bitmap").media)

put afile.compressionratio()

-- 54.237

binaryContent - Whether or not the string data inside a vlist file or member will be crossmapped when it is read on the opposite platform. This property tells you whether binaryMode was turn iu when the dats

Examples:

"vlist",%fooe.lst")

true)

) strin(inn. binaryConteno((((--1)

```
-- member

mem = new(#vlist)

mem.binaryMode = true

mem.content = numToChar(255)

alert(string(mem.binaryContent)) -- 1
```

VLIST XTRA HELP: SHOCKWAVE

All packages contain the vList Xtra for that platform. Depending on the user's platform, a package autodownloaded to the user's hard drive will

information on applying for a Verisign certificate and packaging files.

XTRAINFO.TXT

The text file xtrainfo.txt resides in your Director authoring directory. It contains information about Xtras such as file version names for an Xtra on both platforms and the URL for the packages. The information contained in xtrainfo.txt is saved with each movie you create and used by projectors and Shockwave.

You must create an entry for vList Xtra in your xtrainfo.txt file that specifies the URL on your server for the package files. The last part of

If Director finds the packages, it will transfer information about the package contents for both platforms such as file names and version numbers and embed the information into your Director movie. An informational dialog box will appear that tells you that the packages for both platforms are "downloading". The packages themselves are not downloading, just information about them that the Shockwave movie will need later to compare the version of the Xtra the user possibly already has to the version currently on the server in order to determine if

	Can use read/write	Can use readBinary /writeBinary
	The deleteFile command will delete only vList files	The deleteFile command will delete any file

Mac

kDesktopFolderType

VLIST XTRA HELP: JAVASCRIPT

XTRAS AND JAVASCRIPT

Directo11. Thiveyntax in thivmethods doc fctovList Xtra worksPTDirectPT

VLIST XTRA HELP: HOW DIRECTOR STORES DATA TYPES

All data types are not created equal when it comes to memory storage. Some arestored by value, which means that a se

FLOATS

Prior to Director 8.5, Director reserved 64-bits (8 bytes) for each floating point variable. In Director 8.5 there is a more economical 32-bit (4 byte) floating point data type, which can store floating point numbers with up to 8 digits before the decimal and nothing (0) after the decimal. The storage size required for a float is determined transparently by Director and not normally apparent to the Lingo programmer. However, vList can distinguish between the two float types under Director 8.5 and stores the smaller 32-bit float in its native format both in vList members and on disk.

vList's float32P function returns the storage requirement of a float under Director 8.5.

VLIST XTRA HELP: LIMITATIONS

- #date storage: Sometimes the seconds part of a date will be rounded up in a date retrieved from a vList file. Unfortunately the rounding error is inside MOA, so it can't be fixed. If you are going to need precise seconds stored inside a #date data type the best thing to do is store it in two parts in a list, with the seconds part

```
listObject = w.movie.mGetList()

-- old syntax

tell w to normalList = mGetList()

put listObject

put normalList
```

And yon syt:

```
-- <Object list 3 6a34b90>

-- [1,2,3]
```

- postNetText: Director's postNetText command is limited to 64K of data on all platforms in versions prior to Director 8.5. In Director 8.5, in authoring and projector, on the Mac Classic platform only, postNetText corrupts strings longer than 40K (40960) by overwriting the data staiting at byte 40961 with data from the beginning of the string. This is not a problem in Shockwave on the Mac, which relies on the browser to perform the post. It is not a problem in any environment, authoring, projector, or Shockwave, on Windows.

- If you use "put data after member 'vlist member'" syntax to add list items to a vList member, under certain rare circumstances a saveMovie will not be successful until the vList member contains at least 2 items. The problem exists only in Director 7. It is fixed in Director 8 and above.

Example:

-- member("vlist) staits out empty

repeat with x = 1 to 3

put #picture after member("vlist")

-- workarocer code

if x >= 2 then

saveMovie

end if

end repeat

- Mac only. Under some undetermined circumstances, setting the media of one vList member to the media of another within a repeat loop will corrupt the media property so that the copy fails.

Example:

repeat with x = 1 to 3

member("vList" & x).media = member("vList" & (x + 100)).media

end repeat

Workarocer is to use vList_readFromMedia to obtain the data and avoid the media

VLIST XTRA HELP: LICENSING & AVAILABILITY

vList Xtra ie 8 mfmmercial product.
Current price and updated information