TextCruncher Xtra Help

For up-to-date information please visit our web site:
xtras.tabuleiro.com

TEXTCRUNCHER HELP: INSTALLATION

The installation procedure is slightly different depending on the version of Director and platform used. Make sure you have administrative rights to create files in the directory where Director is installed on your system.

WINDOWS	MACINTOSH
Director 11	Director 11
Director MX 2004	Director MX 2004
Director MX	Director MX
Director 8.5	Director 8.5

TEXTCRUNCHER XTRA HELP: INSTALLATION: MACINTOSH - DIRECTOR 11

INSTALLING THE XTRA ON MAC OSX - Director 11

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

The first step is to copy the Universal binary version of the Xtra, which will be used in the authoring environment and also when creating Mac OSX projectors, for both Intel and PPC machines. This file is located in the install disk image, at:

TextCruncher/Mac Universal/TextCruncher.xtra

This file needs to be copied to the Director 11 Xtras folder. The final pathname for the Xtra in a default installation of Director 11 will be:

OSX Volume Name/Applications/Adobe Director 11/Configuration/Xtras/TextCruncher.xtra

Windows projectors can also be created directly on Director 11 running on Mac OSX after installation of the Windows version of the Xtra. It is located on the install disk, at:

TextCruncher/Windows/TxtCrnch.x32

Copy this file to the Cross Platform resources directory in Director 11, so that it will be available at:

OSX Volume Name/Applications/Adobe Director 11/Configuration/Cross Platform Resources/Windows/Xtras/TxtCrnch.x32

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave and cross-platform publishing features in Director 11, to locate the files needed when assembling the Windows version of your projector. The xtrainfo.txt file is located by default at:

```
OSX Volume Name/Applications/Adobe Director 11/Configuration/xtrainfo.txt
```

Double click the file to open it in TextEdit, or alternatively edit with another text editor. Make sure to save the file in plain text format, though. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

TEXTCRUNCHER XTRA HELP: INSTALLATION: MACINTOSH - DIRECTOR MX 2004

INSTALLING THE XTRA ON MAC OSX - Director MX 2004

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

The first step is to copy the OSX version of the Xtra, which will be used in the authoring environment and also when creating OSX projectors. This file is located in the install disk image, at:

TextCruncher/Mac Carbon/TextCruncher

This file needs to be copied to the Director MX 2004 Xtras folder. The final pathname for the OSX Xtra in a default installation of Director MX will be:

OSX Volume Name/Applications/Macromedia Director MX 2004/Configuration/Xtras/TextCruncher

Director MX 2004 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 TextCruncher to the correct location in your Director MX installation. First locate the Classic version of TextCruncher in the install disk:

TextCruncher/Mac Classic/TextCruncher

This file needs to be copied to the following location in the Director MX 2004 folder, to be used for cross-platform publishing. Copy it to:

OSX Volume Name/Applications/Macromedia Director MX 2004/Configuration/Cross Platform Resources/Classic MacOS/Xtras/TextCruncher

Windows projectors can also be created directly on Director MX 2004 running on Mac OSX after installation of the Windows version of the Xtra. It is located on the install disk, at:

TextCruncher/Windows/TxtCrnch.x32

Copy this file to the Cross Platform resources directory in Director MX 2004, so that it will be available at:

OSX Volume Name/Applications/Macromedia Director MX 2004/Configuration/Cross Platform Resources/Windows/Xtras/TxtCrnch.x32

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave and cross-platform publishing features in Director MX 2004, to locate the files needed when assembling the Classic MacOS and Windows versions of your projector. The xtrainfo.txt file is located by default at:

OSX Volume Name/Applications/Macromedia Director MX 2004/Configuration/xtrainfo.txt

Double click the file to open it in TextEdit, or alternatively edit with another text editor. Make sure to save the file in plain text format, though. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

TEXTCRUNCHER 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 "TextCruncher" on your desktop.

The first step is to copy the OSX version of the Xtra, which will be used in the authoring environment and also when creating OSX projectors. This file is located in the install disk image, at:

TextCruncher/Mac Carbon/TextCruncher

This file needs to be copied to the Director MX Xtras folder. The final pathname for the OSX Xtra in a default installation of Director MX will be:

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

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 TextCruncher to the correct location in your Director MX installation. First locate the Classic version of TextCruncher in the install disk:

TextCruncher Folder/Mac Classic/TextCruncher

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

OSX Volume Name/Applications/Macromedia Director MX/Classic MacOS/Xtras/TextCruncher

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave and cross-platform publishing features in Director MX 2004, to locate the files needed when assembling the Classic MacOS version of your projector. The xtrainfo.txt file is located by default at:

OSX Volume Name/Applications/Macromedia Director MX/xtrainfo.txt

Double click the file to open it in TextEdit, or alternatively edit with another text editor. Make sure to save the file in plain text format, though. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

TEXTCRUNCHER 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 "TextCruncher" on your desktop. To install the Xtra just copy the file "TextCruncher" from the Mac Classic folder to the Xtras folder of your Director 8.5 installation. The final pathname for the Xtra will be for example:

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

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. 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 capable of saving plain text files. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

TEXTCRUNCHER XTRA HELP: INSTALLATION: WINDOWS - DIRECTOR 11

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 "TextCruncher" on your machine. To install the Xtra, just copy the file Windows\TxtCrnch.x32 to the Director 11 XTRAS folder. If your copy of Director 11 is installed at the default location, the Windows Xtra file will be located at:

C:\Program Files\Adobe\Adobe Director 11\Configuration\Xtras\TxtCrnch.x32

Now you need to install the files necessary for creation of cross-platform projector for Mac OSX. Go back to the "TextCruncher" directory where the Xtra files were unpacked. Open the **Mac Universal** directory. Now copy the file "TextCruncher.cpio" to the "Configuration\Cross Platform Resources\Macintosh\Xtras" directory used by Director 11. In a default installation of Director this file will end up at the following location:

C:\Program Files\Adobe\Adobe Director 11\Configuration\Cross Platform Resources\Macintosh\Xtras\TextCruncher.cpio

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave and cross-platform publishing features in Director 11, to locate the files needed when assembling the Mac OSX version of your projector. The xtrainfo.txt file is located by default at:

C:\Program Files\Adobe\Adobe Director 11\Configuration\xtrainfo.txt

Double click the file to open it in notepad, or alternatively edit with any other text editor. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

TEXTCRUNCHER 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.)

Decompress the installation .zip file. This will unpack the Xtra, documentation and sample files to a folder named "TextCruncher" on your machine. To install the Xtra, just copy the file Windows\TxtCrnch.x32 to the Director MX 2004 XTRAS folder. If your copy of Director MX 2004 is installed at the default location, the Windows Xtra file will be located at:

C:\Program Files\Macromedia\Director MX 2004\Configuration\Xtras\TxtCrnch.x32

Now you need to install the files necessary for creation of cross-platform projector for Mac OSX. Go back to the "TextCruncher" directory where the Xtra files were unpacked. Open the **Mac Carbon** directory. Now copy the files "TextCruncher.data" and "TextCruncher.rsrc" files to the "Configuration\Cross Platform Resources\Macintosh\Xtras" directory used by Director MX 2004. In a default installation of Director these files will end up at the following locations:

C:\Program Files\Macromedia\Director MX 2004\Configuration\Cross Platform Resources\Macintosh\Xtras\TextCruncher.data

C:\Program Files\Macromedia\Director MX 2004\Configuration\Cross Platform Resources\Macintosh\Xtras\TextCruncher.rsrc

If you are running Director MX 2004 10.1, you can also install the files necessary for creation of cross-platform projector for Mac Classic. Again, go back to the "TextCruncher" directory where the Xtra files were unpacked. Open the **Mac Classic** directory. Now copy the files "TextCruncher.data" and "TextCruncher.rsrc" files to the "Configuration\Cross Platform Resources\Classic\Xtras" directory used by Director MX 2004. In a default installation of Director these files will end up at the following locations:

C:\Program Files\Macromedia\Director MX 2004\Configuration\Cross Platform Resources\Classic\Xtras\TextCruncher.data

C:\Program Files\Macromedia\Director MX 2004\Configuration\Cross Platform Resources\Classic\Xtras\TextCruncher.rsrc

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave and cross-platform publishing features in Director MX 2004, to locate the files needed when assembling the OSX and Classic versions of your projector. The xtrainfo.txt file is located by default at:

C:\Program Files\Macromedia\Director MX 2004\Configuration\xtrainfo.txt

Double click the file to open it in notepad, or alternatively edit with any other text editor. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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.

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

INSTALLING THE XTRA ON WINDOWS - Director MX and Director 8.5

Decompress the installation .zip file. This will unpack the Xtra, documentation and sample files to a folder named "TextCruncher" on your machine. To install the Xtra, just copy the file **Windows\TxtCrnch.x32** to the Director 8.5 or Director MX XTRAS folder. If you have previously installed an older copy of the Xtra make sure to remove or replace it.

These are the default locations of the Xtras folder for each application:

Director 8.5- C:\Program Files\Macromedia\Director 8.5\Xtras

Director MX- C:\Program Files\Macromedia\Director MX\Xtras

Finally, you need to edit the xtrainfo.txt file to include information about TextCruncher. This information is used by the Shockwave publishing features in Director. The xtrainfo.txt file is located by default at:

Director 8.5 - C:\Program Files\Macromedia\Director 8.5\xtrainfo.txt

Director MX - C:\Program Files\Macromedia\Director MX\xtrainfo.txt

Double click the file to open it in notepad, or alternatively edit with any other text editor. You need to add the following line to the end of the file:

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://download.tabuleiro.com/packages/TextCruncher/5/TextCruncher"]
```

You may want to customize this line in the future, to instruct Director to download TextCruncher 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 (Director MX).

TEXTCRUNCHER XTRA HELP: WHAT TEXTCRUNCHER XTRA DOES

TextCruncher does lightning-fast search and replace of text. You can use it in a Director projector or in a Shockwave movie to:

- Perform a real-time fulltext search

- Create indices for an indexed search of very large amounts of text

- Search HTML pages retrieved with getNetText

- Search and modify field or text members

- Change the formatting of rich text members by replacing tags in "the RTF of member" property

- Change HTML tagging or links of rich text member by replacing HTML tags in "the HTML of member" property

- Search and change code in script members by modifying "the scriptText of member" property

TEXTCRUNCHER XTRA HELP: GETTING STARTED

TextCruncher is a Scripting Xtra. Scripting Xtras are used to extend the Lingo language with new functions and datatypes. Unlike Asset Xtras there is no visual representation of a scripting Xtra in the Director interface, and you can not create castmembers or sprites.

The first step is to download and install the TextCruncher Xtra, following the instructions in the installation page. Now that TextCruncher is installed, let's verify that the installation was successful. If you are using DirectorMX you should see the TextCruncher entry in the Scripting Xtras context menu, appearing at the top of the message window. Selecting the TEXTCRUNCHER submenu and the "put interface" entry will output a list of all commands understood by TextCruncher in the message window. You can also use the following command

Lingo:

```
put the xtralist
```

JavaScript syntax:

```
trace(_player.xtraList)
```

to verify which Xtras are installed, including the version number for each one.

We will now try a simple scripting session using TextCruncher and the message window. All of TextCruncher functions are available as global Lingo keywords, so there is no need to create an instance of the Xtra in order to use them. For example:

Lingo:

```
mystring = ReplaceAll("dog cat", "mouse", "dog")

put mystring

--"mouse cat"
```

JavaScript syntax:

```
var mystring = ReplaceAll("dog cat", "mouse", "dog")

trace(myresult)
```

That's it. You just confirmed that TextCruncher is installed correctly, and its functions are available as new global keywords in Lingo and Javascript, ready to be used.

REGISTRATION

The registered version does not display a warning. If you have purchased the TextCruncher Xtra you received a registration number. Make the following registration call in the first movie that uses TextCruncher commands. Make sure you do it before using any other TextCruncher commands. The code can be in any type of script. The startMovie handler is the most convenient place to put it because it will execute before any other code that might try to use TextCruncher functions.

```
on startMovie TC_Register("DUF98989")

-- your registration number is the string inside the parentheses

end
```

TEXTCRUNCHER XTRA HELP: METHODS AT A GLANCE

Method and arguments	Purpose
TC_Register (registrationCodeString)	Prevents the demo dialog from coming up after purchase.

Search

FindFirst (sourceString, findString)	Returns the integer character position in sourceString of the first occurence of findString.
FindNext (sourceString, findString)	Returns the integer character position in sourceString of the occurence of findString after the current search position.
FindPrevious (sourceString, findString)	Returns the integer character position in sourceString of the occurence of findString before the current search position.
FindAll (sourceString, findString)	Returns a list of the starting character positions of all occurrences of findString in sourceString.

Character Position

GetWordOfCharPosition (sourceString, characterNumber)	Returns the integer word number of sourceString containing the specified character position.
GetLineOfCharPosition (sourceString, characterNumber)	Returns the integer line number of sourceString containing the specified character position.
GetItemOfCharPosition (sourceString, characterNumber, itemDelimiterASCIICod)	Returns the integer item number of sourceString containing the specified character position.

Replace

ReplaceFirst (sourceString, replaceString, findString)	Replaces the first occurrence of findString found in sourceString with replaceString. Returns the modified sourceString.
ReplaceNext (sourceString, replaceString, findString)	Replaces the occurrence of findString found in sourceString on or after the current search position with replaceString. Returns the modified sourceString.
ReplaceAll (sourceString, replaceString, findString)	Replaces all occurrences of findString found in sourceString with replaceString. Returns the modified sourceString.

Search/Replace Properties

SetPosition (characterNumber)	Sets the integer character position in the string from where the next find or replace will start. Affects FindNext, FindPrevious and ReplaceNext.
GetPosition ( )	Returns the integer character position from which the next find or replace will start. Affects FindNext, FindPrevious and ReplaceNext.
SetCaseSensitivity (boolean)	Determines whether the found string must match the case of the search string. (deprecated in D11)

Indexing

GetListOfWords (sourceString)	Returns a list of character chunks delimited by white space.
GetListOfLines (sourceString)	Returns a list of lines delimited by either CR only (Mac) or CR/LF (PC)
GetListOfItems (sourceString, itemDelimiterASCIICode)	Returns a list of items delimited by the itemDelimiter specified.

Case

ToUpperCase (sourceString)	Converts characters with upper case counterparts in sourceString to upper case. Leaves other characters unchanged. Returns the modified string. (deprecated in D11)
ToLowerCase (sourceString)	Converts characters with lower case counterparts in sourceString to lower case. Leaves other characters unchanged. Returns the modified string. (deprecated in D11)

Formatting

HardWrapText (sourceString, charsPerLine)	Limits the length of each line to the specified characters per line. Returns the modified string.
HardCenterText (sourceString, charsPerLine)	Limits the length of each line to the specified characters per line. Centers the text of each line by padding it with spaces on either end. Returns the modified string.
HardAlignTextRight (sourceString, charsPerLine)	Limits the length of each line to the specified characters per line. Right-justifies the text of each line by padding it with spaces on the left. Returns the modified string.

URL Encoding

TC_URLEncode (sourceString)	Hex-encodes some punctuation and characters above ASCII 127 to the web standard Latin-1 character set. Returns the modified string. (deprecated in D11)
TC_URLDecode (sourceString)	Decodes URL-encoded text. Returns the modified string. (deprecated in D11)

Error Reporting

TC_GetLastError ( )	Returns the error status code for the last TextCruncher operation
TC_ErrorCodeToString (errorCodeNumber)	Returns a text description of the error number.

TEXTCRUNCHER XTRA HELP: METHODS DOCUMENTATION

TC_Register	REPLACE	CASE	ERROR REPORTING
	ReplaceFirst	ToUpperCase	TC_GetLastError
SEARCH COMMANDS	ReplaceNext	ToLowerCase	TC_ErrorCodeToString
FindFirst	ReplaceAll		
FindNext		FORMATTING	
FindPrevious	SEARCH/REPLACE PROPERTIES	HardWrapText	
FindAll	SetPosition	HardCenterText	
	GetPosition	HardAlignTextRight	
CHARACTER POSITION			
GetWordOfCharPosition	INDEXING	URL ENCODING	
GetLineOfCharPosition	GetListOfWords	TC_URLEncode	
GetItemOfCharPosition	GetListOfLines	TC_URLDecode	
	GetListOfItems		

TC_Register(registrationCodeString) - No return. The demo version of TextCruncher Xtra will display a trial-version alert the first time you use a method other than TC_Register. If you have purchased the TextCruncher Xtra you received a registration number. Call the register method like so:

TC_Register("48dkd2929")

-- your registration number is the string inside the parentheses parameter

Example:

TC_Register("4812345")

SEARCH COMMANDS

FindFirst(sourceString,findString) - where sourceString is the string to search in and findString is the string to look for. Returns an integer character position in sourceString of the first occurence of findString or 0 if the string was not found or if there was an error

Finds the first occurrence of findString in sourceString and returns the character position of its first character in sourceString. Not affected by SetPosition. Always searches from character 1 of sourceString. After FindFirst the current position is the return value of FindFirst.

Example:

```
put findFirst("abcdefg","c")

-- 3
```

FindNext(sourceString,findString) - where sourceString is the string to search in and findString is the string to look for. Returns and integer character position in sourceString of the occurence of findString on or after the current search position or 0 if the string was not found or if there was an error.

Finds the first occurrence of findString in sourceString after the current position and returns the character position of its first character in sourceString. Current position may have been set by a previous Find or Replace or by the SetPosition command. After FindNext the current position is the return value of FindNext. If there is no occurence after the current position, FindNext returns 0, which sets the current positon to 0, therefore causing any subsequent FindNext to start at the beginning again.

Example:

```
set source = "When you select a category, the filters in that category appear in a list."

put FindFirst(source,"category")

-- 19

put FindNext(source,"category")

-- 49

setPosition(40)
```

put FindNext(source,"category")

-- 49

FindPrevious(sourceString,findString) - where sourceString is the string to search in and findString is the string to look for. Returns an integer character position in sourceString of the occurence of findString before the current search position or 0 if the string was not found or if there was an error.

Finds the first occurrence of findString in sourceString before the current position and returns the character position of its first character in sourceString. Current position may have been set by a previous Find or Replace or by the SetPosition command. After FindPrevious the current position is the return value of FindPrevious. If there is no occurence before the current position, FindPrevious returns 0, which sets the current positon to 0.

Example:

set source = "When you select a category, the filters in that category appear in a list."

setPosition(40)

put FindPrevious(source,"category")

-- 19

FindAll(sourceString,findString) - where sourceString is the string to search in and findString is the string to look for. Returns a list containing the starting character positions of all occurrences of findString in sourceString or empty list if the string was not found or if there was an error.

Finds all occurrences of findString in sourceString and returns a list of the character positions of each found string in sourceString. Not affected by SetPosition. Always searches from character 1 of sourceString. Does not affect current position.

Example:

set source = "When you select a category, the filters in that category appear in a list."

put FindAll(source,"category")

-- [19, 49]

CHARACTER POSITION

The return values of the Find commands would not be very useful unless you had some way of fetching the found word back out of the Director text chunk you were searching. These commands, in tandem with the Find commands can be used to do things like highlight the found word in a Director field or display the found word in context with the other words on the same line.

GetWordOfCharPosition(sourceString,characterNumber) - where sourceString is the string to search in and characterNumber is the integer character position in the string. Returns an integer word number in sourceString of word containing the specified character position or 0 if no word contains the character position or if there was an error. Converts the character position returned by one of the Finds to a word number in the text chunk.

Example:

```
on showWord fieldName,theWord

-- Finds a word in a field and highlights it

--

set charPos = FindNext(field fieldName,theWord)

if TC_GetLastError() = 0 then

set wordNum = GetWordOfCharPosition(field fieldName,charPos)

if wordNum > 0 then

hilite word wordNum of field fieldName

end if

end if

end
```

GetLineOfCharPosition(sourceString,characterNumber) - where sourceString is the string to search in and characterNumber is the integer character position in the string. Returns an integer line number in sourceString of line containing the specified character position or 0 if no line contains the character position or if there was an error. Returns 0 if the character in the specified position is a line ending character - RETURN (13), LINEFEED (10). Converts the character position returned by one of the Finds to a line number in the text chunk.

Example:

```
on showContext fieldName,theWord,contextFieldName

-- Finds the line in a field a word appears in

-- and displays the whole line of text in another

-- field

--

set charPos = FindNext(field fieldName,theWord)

if TC_GetLastError() = 0 then

set lineNum = GetLineOfCharPosition(field fieldName,charPos)

if lineNum > 0 then

put line lineNum of field fieldName into field contextFieldName

end if

end if

end
```

GetItemOfCharPosition(sourceString,characterNumber,itemDelimiterASCIICode) - where sourceString is the string to search in, characterNumber is the integer character position in the string and itemDelimiterASCIICode is the ASCII value (charToNum) of the character to use for the item delimiter. Returns an integer item number in sourceString of item containing the specified character position or 0 if no item contains the character position or if there was an error. Converts the character position returned by one of the Finds to an item number in the text chunk.

Example:

```
on getDataFieldContainingWord dataRecord,theWord
```

```
-- Find word in tab-delimited data record and

-- return data field containing word

--

set charPos = FindFirst(dataRecord,theWord)

if TC_GetLastError() = 0 then

set dataFieldNum = GetItemOfCharPosition(dataRecord,charPos,9)

if dataFieldNum > 0 then

set the itemDelimiter = numToChar(9)

return item dataFieldNum of dataRecord

else

return ""

end if

else

return TC_ErrorCodeToString(TC_GetLastError())

end if

end
```

REPLACE

The following commands find a specified character or string in text and replace it with another. The original string remains unchanged. A copy of the string with the changes is returned. You can perform multiple replaces on the same text by passing the return from one replace to the next like so:

```
set source = "The SRP for MS Word is $299.00"

set source = ReplaceAll(source, "suggested retail price","SRP")

set source = ReplaceAll(source, "Microsoft","MS")
```

```
put source

-- "The suggested retail price for Microsoft Word is $299.00"
```

ReplaceFirst(sourceString,replaceString,findString) - where sourceString is the string to search in, replaceString is the string to replace the found string with and findString is the string to look for. Returns a copy of the source string modified by the replace operation or "" if there was an error.

Finds the first occurrence of findString in the source string and replaces it with replaceString. Not affected by SetPosition. Always searches from character 1 of sourceString. After ReplaceFirst the current position is the characterposition after the end of the replaced string.

This command ignores the current case-sensitivity setting set by SetCaseSensitivity for high-ASCII (above numtochar(125)) characters. For high-ASCII characters it is always case-sensitive. See (deprecated in D11) for more information on case sensitivity.

Example 1:

```
set source = "aaabbb"

put replacefirst(source,"xxx","aaa")

-- "xxxbbb"

put getposition()

-- 4
```

Example 2:

```
on tabsToSpaces memName

-- Replace TAB indents at the beginnings of lines

-- with 5 spaces instead, in a text member

--

set temp = the text of member memName

set li = the number of lines of temp

repeat with x = 1 to li
```

```
set thisLine = line x of temp

if char 1 of thisLine = TAB then

set thisLine = ReplaceFirst(thisLine,"      ",TAB)

put thisLine into line x of temp

end if

end repeat

set the text of member memName = temp

end
```

ReplaceNext(sourceString,replaceString,findString) - where sourceString is the string to search in, replaceString is the string to replace the found string with and findString is the string to look for. Returns a copy of the source string modified by the replace operation or "" if there was an error.

Finds the first occurrence of findString in sourceString on or after the current position and returns the character position of its first character in sourceString. Current position may have been set by a previous Find or Replace or by the SetPosition command. After ReplaceNext the current position is the character position after the end of the replaced string. If ReplaceNext goes past the end of the source string it will return 0 and set position back to 0, which causes a subsequent ReplaceNext to wrap around and start at the beginning of the string.

This command ignores the current case-sensitivity setting set by SetCaseSensitivity for high-ASCII (above numtochar(125)) characters. For high-ASCII characters it is always case-sensitive. See (deprecated in D11) for more information on case sensitivity.

Example:

```
set source = "aaa cat bbb cat ccc cat "

set source = ReplaceFirst(source,"dog","cat")

put source

-- "aaa dog bbb cat ccc cat "

set source = ReplaceNext(source,"dog","cat")

put source
```

```
-- "aaa dog bbb dog ccc cat "

put getposition()

-- 16
```

ReplaceAll(sourceString,replaceString,findString) - where sourceString is the string to search in, replaceString is the string to replace the found string with and findString is the string to look for. Returns a copy of the source string modified by the replace operation or "" if there was an error.

Finds all occurrences of findString in sourceString and replaces each occurrence with replaceString. Not affected by SetPosition. Always searches from character 1 of sourceString. Does not affect current position.

Example 1:

```
set source = "aaa cat bbb cat ccc cat "

set source = ReplaceAll(source,"dog","cat")

put source

-- "aaa dog bbb dog ccc dog "
```

Example 2:

```
on fixLineEndings convertToWhichType

-- Converts Mac format text file to PC or

-- PC to Mac. Utility handler. No error

-- checking built in.

--

-- EX: fixLineEndings("PC")

-- Converts Mac file to PC format. Prompts

-- for file to convert.
```

```
--

set y = new(xtra "fileio")

set theFilePath = displayOpen(y)

if theFilePath <> "" then

openFile(y,theFilePath,1)

set temp = readFile(y)

delete(y)

closeFile(y)

set PCending = RETURN & numToChar(10)

set MacEnding = RETURN

if convertToWhichType = "PC" then

set temp = ReplaceAll(temp,PCending,MacEnding)

else

set temp = ReplaceAll(temp,MacEnding,PCending)

end if

createFile(y,theFilePath)

openFile(y,theFilePath,2)

writeString(y,temp)

closeFile(y)

end if

end
```

SEARCH/REPLACE PROPERTIES

The following properties affect the operation of the Find and Replace commands.

SetPosition(characterNumber) - where characterNumber is the integer character number to reposition search start to. No return. JavaScript Note: This command requires a small Lingo script to work in JavaScript.

Position is the current character position that TextCruncher has been set to. You use SetPosition to set the current character position manually. The following TextCruncher commands also affect the current position:

FindFirst

FindNext

FindPrevious

ReplaceFirst

ReplaceNext

Some TextCruncher commands use the current position as their starting point:

FindNext starts searching from the character after the current position.

FindPrevious starts searching from the character before the current position

ReplaceNext starts searching from and including the current position

TextCruncher does not associate position with any particular string you are searching. You should either use FindFirst or ReplaceFirst to find the first occurrence in a new string or reset the position to 0 manually between operations on different strings to avoid starting a search in the middle of the string instead of at the beginning.

Example 1:

```
-- Position affects commands differently

set source = "aaabbbccc"

setPosition(4)

put FindNext(source,"bbb")

-- 0
```

```
setPosition(4)

put FindPrevious(source,"bbb")

-- 0

setPosition(4)

put ReplaceNext(source,"xxx","bbb")

-- "aaaxxxccc"
```

Example 2

```
-- One operation can set position

-- so that another operation will

-- not start at the beginning of

-- the next string

--

set source = "aaabbbccc"

put FindFirst(source,"bbb")

-- 4

-- Position is now set to 4

set source = "Do not pass go."

put FindNext(source,"Do")

-- 0

-- The operation started at position 4,

-- so it missed the word at pos 1.

-- Should have used FindFirst on the

-- new string.
```

GetPosition( ) - Returns the integer character number where the next search will start. Returns the current character position. JavaScript note: This command requires a small Lingo script to work in JavaScript.

Example:

```
set source = "Do not pass go."

put FindFirst(source,"not")

-- 4

put GetPosition()

-- 4
```

SetCaseSensitivity(onOrOff) - where onOrOff is the boolean value, either 1 (TRUE) to consider case or 0 (FALSE) to ignore case. No return. Determines whether or not the Find and Replace commands will consider case. The default is FALSE - ignore case.

Example:

```
SetCaseSensitivity(FALSE)

set source = "Cart the cart over here."

put FindFirst(source,"cart")

-- 1

SetCaseSensitivity(TRUE)

put FindFirst(source,"cart")

-- 10
```

INDEXING

Use the list commands to index Director text chunks for operations where you otherwise would have to refer back to the text itself. For instance you can use GetListOfWords to speed up a proximity search (word 1 within so many words of word 2). List commands like:

```
if getAt(wordList,5) = secondword
```

are faster than text chunk commands like:

```
if word 5 of field "whatever" = secondword
```

GetListOfWords(sourceString) - where sourceString is the string to operate on. Returns a list of character chunks delimited by white space or empty list if there was an error. Returns a list of character chunks in the string delimited by white space. White space includes TAB(9), Linefeed(10) and Return(13).

Example:

```
on makeIndex thestring

-- Create an alphabetical index of the words

-- in a text chunk

--

set index = [:]

set thestring = ToLowerCase(thestring)

set wordList = GetListOfWords(thestring)

if TC_GetLastError() = 0 then

set numWords = count(wordList)

repeat with w = 1 to numWords

set thisword = getAt(wordList,w)

if voidP(getaprop(index,thisword)) then

addprop index,thisword,list(w)
```

```
else

append getprop(index,thisword),w

end if

end repeat

sort index

end if

return index

end
```

GetListOfLines(sourceString) - where sourceString is the string to operate on. Returns a list of lines delimited by either CR only (Mac) or CR/LF (PC) or empty list if there was an error. Creates a list of the lines contained in sourceString.

Example:

```
on readPCFile filePath

-- Automatically strips off linefeeds and puts lines into

-- a list

--

set f = new(xtra "fileio")

openFile(f,filePath,1)

set filetext = readFile(f)

closeFile(f)

set lineList = GetListOfLines(filetext)

return lineList

end
```

GetListOfItems(sourceString,itemDelimiterASCIICode) - where sourceString is the string to operate on, itemDelimiterASCIICode is the ASCII value (charToNum) of character to use for the item delimiter. Returns a list of items delimited by the itemDelimiterASCIICode specified or empty list if there was an error.

Divides a string into a list of pieces delimited by any character. Get the ASCII code needed by using chartonum() on any typeable character in the message window or refer to the ASCII chart.

Note: A string that begins with the specified item delimiter will create an empty item at the beginning of the list. The old TextCruncher did not create an empty item. This change was made to keep TC consistent with Director's item handling and also to preserve information that would otherwise be lost. For instance, if the string was a data record, the empty item would signify an empty data field and should be retained.

Example 1:

```
set phonenum = "891-752-3344"

put chartonum("-")

-- 45

set phonepieces = GetListOfItems(phonenum,45)

put phonepieces

-- ["891", "752", "3344"]

set areacode = getAt(phonepieces,1)

put areacode

-- "891"
```

Example 2:

```
on readInDataFile theFilePath, keyFieldNumber

-- Reads in tab-delimited data file and converts it

-- to a list indexed on the specified key field

-- Code assumes that record fields are delimited by
-- tabs, records are delimited by either CR or CR/LF,
-- and that the data fields themselves do not contain
-- CR or LF characters.
--
- Created list looks like this:
--
-- ["IN000789":["Vargas","Paul","PART987","Ballpeen hammer","5.00"],
-- "IN000790":["Jones","Marty","PART002","Switchplate","1.97"] ]
--
set f = new(xtra "fileio")
openFile(f,theFilePath,1)
set source = readFile(f)
closeFile(f)
set indexedList = [:]
set lineList = GetListOfLines(source)
if TC_GetLastError() = 0 then
repeat with oneLine in lineList
if oneLine = "" then next repeat
set fieldList = GetListOfItems(oneLine,9)
set key = getAt(fieldList,keyFieldNumber)
if not voidP(getaprop(indexedList,key)) then
alert "Duplicate key: " & key
else
deleteAt fieldList,keyFieldNumber
addProp indexedList,key,duplicate(fieldList)
```

```
end if

end repeat

end if

return indexedList

end
```

CASE

These operations use the Mac Standard Roman character set on Mac and PC ANSI on PC. Some decorative or shareware fonts put non-standard characters in empty or little-used character positions, which can cause unexpected results when characters above 127 are cased. Check the character set crossmap against the font before reporting a casing bug. These methods return a modified copy of the source string, leaving the original string unchanged. See (deprecated in D11) for more information on character encoding changes.

ToUpperCase(sourceString) - where sourceString is the string to operate on. Returns a copy of the source string uppercased or "" if there was an error. Replaces any character in the string with its uppercase counterpart if there is one. Leaves the character unchanged if there is no uppercase character for it. Uses Mac Standard Roman character set on Mac and PC ANSI on PC. This method is deprecated in D11.

Example:

```
set source = "uppercase this!"

put ToUpperCase(source)

-- "UPPERCASE THIS!"
```

ToLowerCase(sourceString) - where sourceString is the string to operate on. Returns a copy of the source string lowercased or "" if there was an error. Replaces any character in the string with its lowercase counterpart if there is one. Leaves the character unchanged if there is no lowercase character for it. This method is deprecated in D11.

Example:

set source = "pUT $50,000 iN UnMARked BIlLs IN tHe BRieFcaSE."

set source = ToLowerCase(source)

put ToUpperCase(char 1 of source) into char 1 of source

put source

-- "Put $50,000 in unmarked bills in the briefcase."

FORMATTING

The following commands break text up into lines of a specified length and align the text on each line to the left, center or right margin. They are most useful for formatting plain text that will display in a monospaced font such as Courier. Lines created by wrapping are delimited with the Mac line endings (CR only). A single word in the text that is longer than the wrap length will not be broken, and will therefore cause that line to be longer than the wrap length. These methods return a modified copy of the source string, leaving the original string unchanged.

HardWrapText(sourceString,charsPerLine) - where sourceString is the string to operate on and charsPerLine is the maximum number of characters per line. Returns a copy of the source string reformatted or "" if there was an error. Aligns text to left margin and breaks any line longer than the specified length limit with RETURN.

Example:

set ruler = RETURN & "123456789012345678901234567890" & RETURN

set source = "The quick brown fox jumped over the lazy dog."

set wrapped = HardWrapText(source,20)

put ruler & wrapped

-- "

123456789012345678901234567890

The quick brown fox

jumped over the lazy

dog."

HardCenterText(sourceString,charsPerLine) - where sourceString is the string to operate on and charsPerLine is the maximum number of characters per line. Returns a copy of the source string reformatted or "" if there was an error. Breaks any line longer than the specified length limit with RETURN. Centers text shorter than the specified line length by padding it on the left with spaces.

Example:

set ruler = RETURN & "123456789012345678901234567890" & RETURN

set source = "The quick brown fox jumped over the lazy dog."

set wrapped = HardCenterText(source,30)

put ruler & wrapped

-- "

123456789012345678901234567890

The quick brown fox jumped

over the lazy dog."

HardAlignTextRight(sourceString,charsPerLine) - where sourceString is the string to operate on and charsPerLine is the maximum number of characters per line. Returns a copy of the source string reformatted or "" if there was an error. Breaks any line longer than the specified length limit with RETURN. Aligns text shorter than the specified line length to the right margin by padding it on the left with spaces. Not currently supported under Mac OSX.

Example:

set ruler = RETURN & "123456789012345678901234567890" & RETURN

set source = "The quick brown fox jumped over the lazy dog."

```
set wrapped = HardAlignTextRight(source,30)

put ruler & wrapped

-- "

123456789012345678901234567890
        The quick brown fox jumped
                 over the lazy dog."
```

URL ENCODING

The following commands encode/decode punctuation characters in the lower ASCII range (< 128) and all characters in the high ASCII range, to make text that can be transmitted over the internet. These methods return a modified copy of the source string, leaving the original string unchanged.

The encoding conforms to RFC1738 for low ASCII characters. For characters 128 and above, which are not covered by RFC1738, the hex value of the corresponding character in the ISO 8859-1 Latin-1 character set is used.

Huh?

The content of a URL cannot contain characters that have special meaning within the URL like ":" or "/" or the browser will interpret the URL incorrectly. The convention for including such characters in a URL is to represent them with a percent sign followed by a two character hex value. The same format of encoding is used in e-mail messages to transmit 8-bit characters (> 128). TextCruncher's URL encoding converts text to a format that can be used in a URL, sent to a CGI script, or transmitted in e-mail successfully.

ASCII characters (Decimal 0 - 127)

These characters are the same on the Mac and the PC. TextCruncher's URLencoding encodes all punctuation in this character range and all non-printable characters. Spaces are converted to %20 rather than +, although + is used by many browsers. Encoding with %20 rather than + enables the converted text to be used by CGI scripts and e-mail as well as browsers. Browsers that recognize + for space will recognize %20 as well.

ASCII Chart

Decimal	Character	Decimal	Character	Decimal	Character	Decimal	Character
0	NUL	32		64	@	96	`
1	SOH	33	!	65	A	97	a
2	STX	34	"	66	B	98	b
3	ETX	35	#	67	C	99	c
4	EOT	36	$	68	D	100	d
5	ENQ	37	%	69	E	101	e
6	ACK	38	&	70	F	102	f
7	BEL	39	'	71	G	103	g
8	BS	40	(	72	H	104	h
9	TAB	41	)	73	I	105	i
10	LF	42	*	74	J	106	j
11	VT	43	+	75	K	107	k
12	FF	44	,	76	L	108	l
13	CR	45	-	77	M	109	m
14	SO	46	.	78	N	110	n
15	SI	47	/	79	O	111	o
16	DLE	48	0	80	P	112	p
17	DC1	49	1	81	Q	113	q
18	DC2	50	2	82	R	114	r
19	DC3	51	3	83	S	115	s
20	DC4	52	4	84	T	116	t
21	NAK	53	5	85	U	117	u
22	SYC	54	6	86	V	118	v

23	ETB	55	7	87	W	119	w
24	CAN	56	8	88	X	120	x
25	EM	57	9	89	Y	121	y
26	SUB	58	:	90	Z	122	z
27	ESC	59	;	91	[	123	{
28	FS	60	<	92	\	124	|
29	GS	61	=	93	]	125	}
30	RS	62	>	94	^	126	~
31	US	63	?	95	_	127	DEL

8-bit characters "high ASCII" (Decimal 128 - 255)

ISO 8859-1 Latin1 is the current web standard character set but it does not agree exactly with either Mac Standard Roman or ANSI, the western standards on Mac and PC. TextCruncher maps both Mac and PC characters to the corresponding ISO 8859-1 Latin1 character, if there is one, which is the same convention used for escaping 8-bit characters in HTML.

Both Mac and PC character sets contain characters that do not have any ISO counterpart. Those are the tinted characters in the 8-bit character crossmap chart below. The ANSI characters tinted green in the chart, have no corresponding characters in Latin1, but are still encoded and decoded by many browsers using their ANSI values. These ANSI characters and their Mac counterparts are encoded by TextCruncher using the ANSI value. Mac characters with no ISO counterpart and no counterpart in the "green" ANSI range, like the Mac apple (240) are encoded as %3F (?). ANSI positions, like 128, that contain no character are also encoded as ?. Since the Mac-only characters (pink with no matching character in green) will not be preserved, it's a good idea to stay away from them in text that is destined to be URL-encoded.

Keep in mind that because the character values after 127 represent different characters on the Mac and on the PC, most character positions above 127 will URLencode differently depending on whether TextCruncher is running on a Mac or PC. For example, numToChar(168) is the registered trademark symbol on the Mac, so it encodes to %AE (decimal 174), which is its character code in Latin1.On the PC, numToChar(174) is the registered trademark so that character position is what encodes to %AE.

Crossmap for 8-bit Characters

🟦🟪 = not in ISO 8859-1 Latin1

Decimal	Mac Standard Roman	ANSI (PC)	ISO 8859-1 Latin1	Decimal	Mac Standard Roman	ANSI (PC)	ISO 8859-1 Latin1
128	Ä	▨	▨	192	¿	À	À
129	Å	▨	▨	193	¡	Á	Á
130	Ç	‚	▨	194	¬	Â	Â
131	É	ƒ	▨	195	√	Ã	Ã
132	Ñ	„	▨	196	ƒ	Ä	Ä
133	Ö	…	▨	197	≈	Å	Å
134	Ü	†	▨	198	∆	Æ	Æ
135	á	‡	▨	199	«	Ç	Ç
136	à	ˆ	▨	200	»	È	È
137	â	‰	▨	201	…	É	É
138	ä	Š	▨	202		Ê	Ê
139	ã	‹	▨	203	À	Ë	Ë
140	å	Œ	▨	204	Ã	Ì	Ì
141	ç	▨	▨	205	Õ	Í	Í
142	é	▨	▨	206	Œ	Î	Î
143	è	▨	▨	207	œ	Ï	Ï
144	ê	▨	▨	208	–	Ð	Ð
145	ë	‘	▨	209	—	Ñ	Ñ
146	í	’	▨	210	“	Ò	Ò
147	ì	“	▨	211	”	Ó	Ó
148	î	”	▨	212	‘	Ô	Ô
149	ï	•	▨	213	’	Õ	Õ
150	ñ	–	▨	214	÷	Ö	Ö
151	ó	—	▨	215	◊	×	×
152	ò	˜	▨	216	ÿ	Ø	Ø
153	ô	™	▨	217	Ÿ	Ù	Ù
154	ö	š	▨	218	⁄	Ú	Ú
155	õ	›	▨	219	¤	Û	Û
156	ú	œ	▨	220	‹	Ü	Ü
157	ù	▨	▨	221	›	Ý	Ý
158	û	▨	▨	222	ﬁ	Þ	Þ
159	ü	Ÿ	▨	223	ﬂ	ß	ß
160	†			224	‡	à	à
161	°	¡	¡	225	·	á	á
162	¢	¢	¢	226	‚	â	â
163	£	£	£	227	„	ã	ã
164	§	¤	¤	228	‰	ä	ä
165	•	¥	¥	229	Â	å	å
166	¶	¦	¦	230	Ê	æ	æ
167	ß	§	§	231	Á	ç	ç
168	®	¨	¨	232	Ë	è	è
169	©	©	©	233	È	é	é
170	™	ª	ª	234	Í	ê	ê
171	´	«	«	235	Î	ë	ë
172	¨	¬	¬	236	Ï	ì	ì
173	≠	-	-	237	Ì	í	í
174	Æ	®	®	238	Ó	î	î
175	Ø	¯	¯	239	Ô	ï	ï
176	∞	°	°	240		ð	ð

TC_URLEncode(sourceString) - where sourceString is the string to operate on. Returns a copy of the source string encoded or "" if there was an error. Hex-encodes the punctuation and 8-bit characters in a string. This method is deprecated in D11.

Example:

```
set source = "This is an encoded string."

put TC_URLEncode(source)

-- "This%20is%20an%20encoded%20string%2e"
```

TC_URLDecode(sourceString) - where sourceString is the string to operate on. Returns a copy of the source string decoded or "" if there was an error. Converts hex-encoding sequences in a string back to single-characters. This method is deprecated in D11.

Example:

```
set source = "This%20is%20an%20decoded%20string%2e"

put TC_URLDecode(source)

-- "This is an decoded string."
```

ERROR REPORTING

The previous version of TextCruncher could return error values or data from the same call. The current version returns only data from calls with return values. If there is an error that prevents the call from returning data, the closest thing to nothing, but in the same data type, is returned. For instance a call that normally returns a Lingo list will return an empty list if there is an error. A call that normally returns a string will return an empty string if there is an error. This method of error handling insures that the same data type will always be returned from a call, so it enables the Lingo programmer to write more generic error handlers.

Instead of examining the return value to determine if an error occurred, use TC_GetLastError() directly after making any other call you want to monitor status for. The following chart lists the possible return values from TC_GetLastError()

Error number	Description
0	No Error
-10	Not enough memory to complete operation
-11	Findstring parameter is empty string
-12	Source string parameter is empty string
-13	Serial Number is Invalid
-14	Attempt to set position to a negative value
-16	Attempt to set case sensitivity to value other than 0 or 1
-17	The number of characters to wrap the lines is not positive
-18	The character position is white space
-19	The character position specified is greater than the length of the source string

TC_GetLastError( ) - Returns 0 for no error or a negative number if an error occurred. Returns the status of the last TextCruncher call made. Returns status for all calls other than TC_GetLastError itself or TC_ErrorCodeToString.

Example:

```
set foundList = FindAll(source,"madras")

set err = TC_GetLastError()

if err = 0 then

--- continue normal operation

else if err = -10 then

-- Inform user about error user can
-- do something about

alert "Not enough memory to continue"

end if
```

TC_ErrorCodeToString(errorCodeNumber) - where errorCodeNumber is the negative integer returned from TC_GetLastError. Returns the string description of the error code. Returns the text description of the specified error code.

Example:

```
on showError which

alert(TC_ErrorCodeToString(which))

end
```

TEXTCRUNCHER XTRA HELP: JAVASCRIPT

XTRAS AND JAVASCRIPT

Director MX 2004 added JavaScript as an alternative scripting language, and it is also available in Director 11. The syntax in the methods doc for TextCruncher Xtra works for both Lingo and JavaScript. For more info, see the tech note on JavaScript and Xtras.

TEXTCRUNCHER XTRA HELP: SHOCKWAVE

TextCruncher Xtra can be used in Shockwave: the distribution package provides packaged Xtras that are downloaded automatically to the user's machine, and installed on demand. Please consult Adobe's web site for a complete overview of the Xtras automated download mechanism: read the Shockwave Xtras downloading overview technote. The basic steps required to make TextCruncher Xtra available for download are outlined below.

To create a Shockwave movie that will auto-download the Xtra to the user's hard drive you must do the following, in this order:

1. Upload the packaged Xtra files to your web server

2. Modify the entry for TextCruncher Xtra in file xtrainfo.txt to point to the packaged Xtra files on your server

3. Do Modify -> Movie -> Xtras, select TextCruncher Xtra, and check the "Download if Needed" option

Once you have completed steps 1 and 2, you can create other Shockwave movies by doing only step 3.

PACKAGED FILES

Your TextCruncher Xtra archive contains a subfolder called "Shockwave". There are four files inside it:

TextCruncher.w32 - Win 32 package

TextCruncher.ppc - Mac Classic package

TextCruncher.carb - Mac Carbon package

TextCruncher.xpku - Mac Universal Binary package

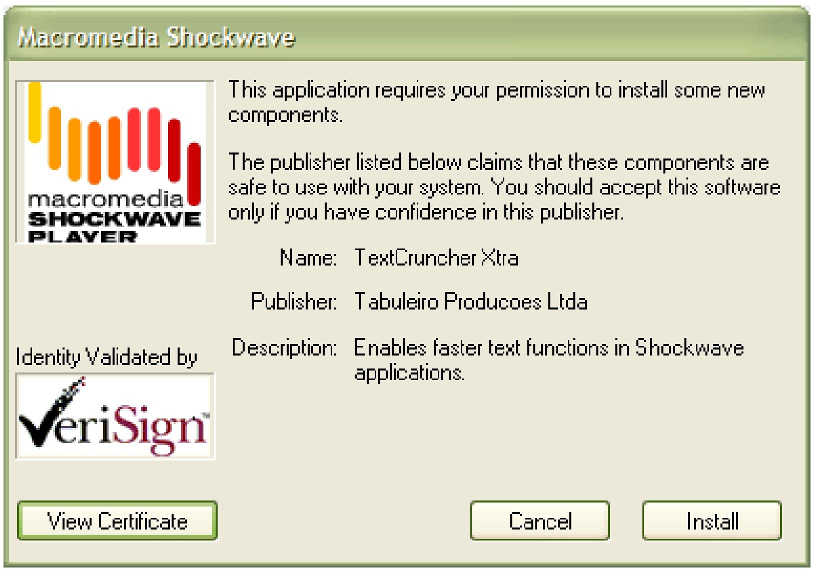
All packages contain the TextCruncher Xtra for that platform. Depending on the user's platform, a package autodownloaded to the user's hard drive will install the correct TextCruncher Xtra for the user's platform into their Shockwave support folder.

If, for some reason, you choose not to make your Shockwave movies autodownload the package files, you can have the user install the right Xtra for their platform into the Shockwave support folder manually.

Upload all package files to the same directory on your web server. Use a "binary" or "raw", not "text" transfer. If the packages are uploaded to two different directories, autodownloading will not work. Do not rename the package files.

If you are going to distribute TextCruncher Xtra with Shockwave movies, we recommend that you use your own web server to do so. Packages are available at Tabuleiro's download services, but we reserve the right to refuse access, without notice, to any referring URL that generates excessive traffic.

The TextCruncher Xtra packages included with the download have been signed and packaged by Tabuleiro, and will present the following security message to users of your Shockwave movies when they are installed for the first time:



This application requires your permission to install some new components.

The publisher listed below claims that these components are safe to use with your system. You should accept this software only if you have confidence in this publisher.

Name:	TextCruncher Xtra
Publisher:	Tabuleiro Producoes Ltda
Description:	Enables faster text functions in Shockwave applications.

You may choose to repackage TextCruncher Xtra and sign it with your own Verisign certificate. You might want to do this if you want your own company name to appear in the auto-download dialog box the user sees when an auto-download is initiated. Adobe is the best source of 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 TextCruncher Xtra in your xtrainfo.txt file that specifies the URL on your server for the package files. The last part of the path will always be "TextCruncher". That specifies the filename of the packages within the directory, without the file extension. Do not include a file extension at the end of the path.

```
[#namePPC:"TextCruncher", #nameW32:"TxtCrnch.x32",
#package:"http://www.domain.com/folder/TextCruncher"]
```

Make sure that the line above does not contain any return character after you paste it into your xtrainfo.txt file. Open your text editor wide and make sure the line does not wrap. If the opening and closing brackets are not on the same line, Director will not be able to create a valid list from the entry and the "Download if needed" button will be dimmed for TextCruncher Xtra in Director.

If you edit xtrainfo.txt while Director is open you should quit and restart Director to read in the changed information in xtrainfo.

EDITING THE MOVIE'S XTRAS LIST

Open the Director movie that you want to save as Shockwave. Choose Modify -> Movie -> Xtras and add TextCruncher. Select TextCruncher from the list and check the "Download if needed" option. Director will initiate an internet connection and look for the packages at the URL you specified in xtrainfo.txt.

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 autodownloading is necessary. The Director movie needs information about both platforms because it may find itself running on either platform once it is on the web.

Once "downloading" of the packages has finished, save the movie, then publish as Shockwave. The finished Shockwave movie can reside at any URL. It does not have to be in the same directory or even on the same server as the packaged Xtras.

If a connection cannot be opened, or the packages cannot be found at the specified location, Director will uncheck the "Download as needed" option automatically. You must have a successful connection for the box to remain checked. A Shockwave made out of a Director movie with the "Download as needed" button unchecked will not autodownload TextCruncher Xtra.

TEXTCRUNCHER XTRA HELP: LINGO TIPS FOR INCREASING SPEED

Operations on text chunks are much faster if you put the contents of the field or text member into a string variable and operate on that rather than on the field or text member itself. For example:

```
on oneMethod

SetPosition(0)

repeat while firstChar > 0

set firstChar = FindNext(field "someField","cat")

if char(firstChar + 3) of field "someField" = " " then

ReplaceNext(field "someField","cat","dog")

end if

end repeat

end


on fasterMethod

set temp = field "someField"

SetPosition(0)

repeat while firstChar > 0

set firstChar = FindNext(temp,"cat")

if char(firstChar + 3) of temp = " " then

ReplaceNext(temp,"cat","dog")

end if

end repeat

put temp into field "someField"
```

```
end
```

When you pass a large string variable as the argument to a handler, Lingo has to make a copy of the string variable which can really slow things down. If it is a very large string and you don't have enough memory to make the copy, Director might even crash. In such a case it is faster to store the large string in a global and let the handler operate directly on the global. If you are familiar with behaviors, you can accomplish the same thing by storing the string once as a property variable that you reference from the behavior's handlers.

```
on oneMethod findString,hugeSearchString

if FindFirst(hugeSearchString,findString) > 0 then

return TRUE

else

return FALSE

end if

end
```

```
on fasterSaferMethod findString

global hugeSearchString

if FindFirst(hugeSearchString,findString) > 0 then

return TRUE

else

return FALSE

end if

end
```

TEXTCRUNCHER XTRA HELP: LIMITATIONS

- **IMPORTANT**: TextCruncher is available for Director 11, in order to allow distribution and upgrade of legacy content. However, Director 11 uses a different string encoding (Unicode, specifically UTF8) in all internal string representations. As a result of this change, several functions of TextCruncher are no longer relevant, and have been deprecated or superseded by built-in commands. These limitations and the recommended workarounds are documented at this page.

- Under JavaScript only, TextCruncher's SetPosition and GetPosition commands conflict with commands of the same name inside Macromedia's FileIO Xtra. In order to use the commands, create a Lingo movie type script with the following code:

```
on tcSetPos pos

SetPosition (pos)

end



on tcGetPos

return GetPosition

end
```

In your JavaScript code, substitute "tcSetPos" and "tcGetPos" for the SetPosition and GetPosition commands, which will call the Lingo wrapper script to execute the commands.

- The function HardAlignTextRight does not currently work under Mac OSX.

- ReplaceFirst and ReplaceNext ignore the current case-sensitivity setting set by SetCaseSensitivity for high-ASCII (above numtochar(125)) characters. For high-ASCII characters those two methods are always case-sensitive.

- Lingo treats the character numToChar(0) as end of string. If you read a string in from an outside source containing numToChar(0) Lingo will truncate the string at that point. TextCruncher will not be able to operate on such a string correctly, nor will Lingo. Normal text files do not contain character 0.

```
set astring = "cat" & " " & numtochar(0) & "dog"

put astring

-- "cat "

put length(astring)

-- 8
```

- TextCruncher does not work with double-byte character sets on Director MX2004 and earlier. Also, some TextCruncher commands such as ToUpperCase, ToLowerCase,URLEncode and URLDecode were designed to work with the Latin-1 character set and will not work with other 8-bit character sets.

- Certain versions of Director cannot display more than 32K of text in the Message Window at a time. During development, put results from TextCruncher functions into rich text members rather than the Message Window to examine them, or put them into global variables and use chunk commands like "put line 5 of myGlobal" to examine smaller sections.

- If EasyBase Xtra is present in the Xtras folder and TextCruncher Xtra is added to the folder, EasyBase Xtra will prevent TextCruncher Xtra from loading and Director will display a "Duplicate Xtra" dialog. Workaround is to temporarily remove EasyBase Xtra and restart Director. Then add EasyBase Xtra and start Director again. At this point both xtras can load but the bogus "Duplicate Xtra" dialog will still appear.

TEXTCRUNCHER XTRA HELP: UPDATING PROJECTS FROM V1.0

The return values of some commands were changed to make them consistent. If a command that finds text does not find the target, it returns the closest thing to "nothing" in the same data type it would return if the text was found. For instance the FindAll command returns a list of character positions for the found string or an empty list if the string was not found. In v1.0 it returned void if the string was not found. This change in behavior may require changes in error handling for Lingo written for v1.0. Check the doc for the current return value for TextCruncher commands used in old code. You may also want to take advantage of additional error information provided by the new commands TC_GetLastError and TC_ErrorCodeToString.

TEXTCRUNCHER HELP: HOW TO ORDER & REGISTER

The unregistered version of TextCruncher Xtra is fully-functional and may be used for evaluation, nonprofit and educational purposes only: commercial distribution is strictly prohibited. A registered version of the Xtra can be used in commercial products, and may be purchased online at xtras.tabuleiro.com, using a secure server. At our web site you can also consult our purchase policy, purchase instructions, payment, delivery and security methods.

If you decide to buy the Xtra you don't need to download a new copy of the software. After your order is processed you will receive an e-mail with a serial number to register the software you've already installed on your machine.

To register the Xtra you should use the TC_Register() function, usually called at the startup of your movie, or before a TextCruncher Xtra function is used. More information about specific syntax can be found at the Methods Documentation page. Please keep your serial number archived for future reference.

TEXTCRUNCHER HELP: LICENSING & AVAILABILITY

TextCruncher Xtra is a commercial product. Current price and updated information can be found at xtras.tabuleiro.com. If your product provides printed documentation and package we ask you to kindly include the following copyright information:

No royalty-fees are required for a distribution of the Xtra with your product.

TEXTCRUNCHER HELP: TECHNICAL SUPPORT

Please use the Your Account section available at our web site xtras.tabuleiro.com to submit your questions. The site also contains Technotes and other resources that can help you identify and solve the most common problems quickly.