NeXTtoMacText

INHERITS FROM   TextConverter

DECLARED IN   NeXTtoMacText.h

CLASS DESCRIPTION

The NeXTtoMacText class serves to convert raw text from the standard NeXT character set to the standard Mac character set.   It accomplishes this by overriding the methods provided by TextConverter class, and introducing its own conversion algorithms instead.   See the TextConverter class document for details about the way one of these classes works.

This uses the standard NeXTstep encoding vector defined in the version 2 documentation.   Most of the conversion is straight forward.   Those characters that can not be mapped to a single Macintosh character are either not altered at all, and an error code is set, or a string representing them is substituted in their place.   In all cases, there is room for ambiguity when dealing with single quotes.   On the NeXT, the single quotes ` amd ' are both in the ASCII set (below 127) and are also proper 'curly' quotes.   The Mac has a neuter quote and a grave accent that is also used as an opening quote by some in those positions.   Some users would want to convert the NeXT quotes to the Macintosh typograhically pretty quotes, others will want to convert them to the standard ASCII positions.   Thus, this allows a flag (UseCurlyQuotes) to be set to YES if the user wishes to map them to the typographically pretty quotes.   Otherwise standard ASCII ones are generated.

A table listing the conversions done by this class is provided at the end.

INSTANCE VARIABLES

Declared in MacToNeXTText	Boolean	strictIM

METHOD TYPES

	Initalizing	- init
Converting characters	- ConvertCharacter:	
	- ConvertString:WithLength:	
Setting conversion type	- SetMode:	

CLASS METHODS

None

INSTANCE METHODS

ConvertCharacter:
- (Character) **ConvertCharacter:** (Character) *macChar*
  - 0   Result codes and test
  - 1   The returned character

This returns a single character in the Macintosh character encoding set that most closely matches the specified Macintosh character.   If the match can not be made exactly, this sets the error to errCANTMAPTOONE.

ConvertString:WithLength:
- (Pointer) **ConvertString:** (Pointer) *theData* **WithLength:** (Integer) *length*

0   Result codes and test
1   The returned Pointer
2   The lenght of the returning data

This behaves as documented in the TextConverter class: it converts data from the source string to a destination string it creates, and it then returns this string with its length.   See the description above, and the table below, for encoding details.

init
- (Instance) **init**
    0   Result codes and test

This initalizes the object, including setting its UseCurlyQuotes flag to NO.

UseCurlyQuotes:
- (Instance) **UseCurlyQuotes:** (Boolean) *useCurlyQuotes*
    0   Result codes and test

This allows the caller to specify whether future conversions should convert the NeXTstep characters ` and ' to equivalent *looking* Macintosh characters (above 128), or equivalent characters in the ASCII set (below 128).   If set to YES, it converts to the upper.   Otherwise, it converts to the lower.

BUGS AND PROBLEMS

I'm not at all sure these are ideal implementations for two reasons: One could argue that a general class that reads a conversion table in would be nicer.   Yet, then one must cart around a conversion table outside the class somewhere, and I don't like that very much, despite its general advantages.   The other is that If one has a string of things to be converted, and would like to do it faster than one character converted per method call, but doesn't want the strings offered in the ConvertString method when an unknown character is found, one is stuck!

ENHANCEMENT IDEAS

CONSTANT, DEFINED TYPES AND ERROR CODES

#define errCANTMAPTOONE   1001

CONVERSION TABLE

Characters from 0x00 to 0x79 are converted exactly as the superclass converts them, with the following exceptions (the second is to convert the line termination character)

Character name	NeXT code	Mac code
NUL	0x00	0x00
LF/CR	0x0A	0x0C
quoteright	0x27	0xD4 (only if UseCurlyQuote is YES)
quoteleft	0x60	0xD5 (only if UseCurlyQuote is YES)

Characters above 0x79 are converted as detailed in the following columns.   Note that an entry of `ÐÐ' means that the specified Mac character could not be converted.   When this occurrs, the object returns the Mac character code, and sets an error value.   If the strict Inside Mac V. 1 flag is set, then all characters above D8 are returned as themselves with no error (as shown in column 5).   If one uses the ConvertString method, the object returns the values detailed in column 4. Note that these never return a value of `ÐÐ', but instead return the name of the character it could not convert. (the values with `ÐÐ' in the final column will get the strings showin the ConvertString column if this flag is set when calling that method)

Character name	NeXT code	Mac code	ConvertString
nbspace	0x80	0xCA	0xCA
Agrave	0x81	0xCB	0xCB
Aacute	0x82	0xE7	0xE7
Acircumflex	0x83	0xE5	0xE5
Atilde	0x84	0xCC	0xCC
Adieresis	0x85	0x80	0x80

Aring	0x86	0x81	0x81
Ccedilla	0x87	0x82	0x82
Egrave	0x88	0xE9	0xE9
Eacute	0x89	0x83	0x83
Ecircumflex	0x8A	0xE6	0xE6
Edieresis	0x8B	0xE8	0xE8
Igrave	0x8C	0xED	0xED
Iacute	0x8D	0xEA	0xEA
Icircumflex	0x8E	0xEB	0xEB
Idieresis	0x8F	0xEC	0xEC
Eth	0x90	ÐÐ	[Eth]
Ntilde	0x91	0x84	0x84
Ograve	0x92	0xF1	0xF1
Oacute	0x93	0xEE	0xEE
Ocircumflex	0x94	0xEF	0xEF
Otilde	0x95	0xCD	0xCD
Odieresis	0x96	0x85	0x85
Ugrave	0x97	0xF4	0xF4
Uacute	0x98	0xF2	0xF2
Ucircumflex	0x99	0xF3	0xF3
Udieresis	0x9A	0x86	0x86
Yacute	0x9B	ÐÐ	[Yacute]
Thorn	0x9C	ÐÐ	[Thorn]
mu	0x9D	0xB5	0xB5
multiply	0x9E	ÐÐ	[multiply]
divide	0x9F	0xD6	0xD6
copyrightserif	0xA0	0xA9	0xA9
exclamdown	0xA1	0xC1	0xC1
cent	0xA2	0xA2	0xA2
sterling	0xA3	0xA3	0xA3
fraction	0xA4	0xDA	0xDA
yen	0xA5	0xB4	0xB4
florin	0xA6	0xC4	0xC4
section	0xA7	0xA4	0xA4
currency	0xA8	0xDB	0xDB
quotesingle	0xA9	0x27	0x27

quotedblleft	0xAA	0xD2	0xD2
guillemotleft	0xAB	0xC7	0xC7
guilsinglleft	0xAC	0xDC	0xDC
guilsinglright	0xAD	0xDD	0xDD
fi	0xAE	0xDE	0xDE
fl	0xAF	0xDF	0xDF
registerserif	0xB0	0xA8	0xA8
endash	0xB1	0xD0	0xD0
dagger	0xB2	0xA0	0xA0
daggerdbl	0xB3	0xE0	0xE0
periodcentered	0xB4	0xE1	0xE1
brokenbar	0xB5	ÐÐ	[brokenbar]
paragraph	0xB6	0xA6	0xA6
bullet	0xB7	0xA5	0xA5
quotesinglebase	0xB8	0xE2	0xE2
quotedblbase	0xB9	0xE3	0xE3
quotedblright	0xBA	0xD3	0xD3
guillemotright	0xBB	0xC8	0xC8
elipsis	0xBC	0xC9	0xC9
perthousand	0xBD	0xE4	0xE4
logicalnot	0xBE	0xC2	0xC2
questiondown	0xBF	0xC0	0xC0
onesuperior	0xC0	ÐÐ	[onesuperior]
grave	0xC1	0x60	0x60
acute	0xC2	0xAB	0xAB
circumflex	0xC3	0xF6	0xF6
tilde	0xC4	0xF7	0xF7
macron	0xC5	0xF8	0xF8
breve	0xC6	0xF9	0xF9
dotaccent	0xC7	0xFA	0xFA
dieresis	0xC8	0xAC	0xAC
twosuperior	0xC9	ÐÐ	[twosuperior]
ring	0xCA	0xFB	0xFB
cedilla	0xCB	0xFC	0xFC
threesuperior	0xCC	ÐÐ	[threesuperior]
hungarumlaut	0xCD	0xFD	0xFD

ogonek	0xCE	0xFE	0xFE
caron	0xCF	0xFF	0xFF
emdash	0xD0	0xD1	0xD1
plusminus	0xD1	0xB1	0xB1
onequarter	0xD2	ĐĐ	[onequarter]
onehalf	0xD3	ĐĐ	[onehalf]
threequarters	0xD4	ĐĐ	[threequarters]
agrave	0xD5	0x88	0x88
aacute	0xD6	0x87	0x87
acircumflex	0xD7	0x89	0x89
atilde	0xD8	0x8B	0x8B
adieresis	0xD9	0x8A	0x8A
aring	0xDA	0x8C	0x8C
ccedilla	0xDB	0x8D	0x8D
egrave	0xDC	0x8F	0x8F
eacute	0xDD	0x8E	0x8E
ecircumflex	0xDE	0x90	0x90
dieresis	0xDF	0x91	0x91
igrave	0xE0	0x93	0x93
AE	0xE1	0xAE	0xAE
iacute	0xE2	0x92	0x92
ordfeminine	0xE3	0xBB	0xBB
icircumflex	0xE4	0x94	0x94
idieresis	0xE5	0x95	0x95
eth	0xE6	ĐĐ	[eth]
ntilde	0xE7	0x96	0x96
Lslash	0xE8	ĐĐ	[Lslash]
Oslash	0xE9	0xAF	0xAF
OE	0xEA	0xCE	0xCE
ordmasculine	0xEB	0xBC	0xBC
ograve	0xEC	0x98	0x98
oacute	0xED	0x97	0x97
ocircumflex	0xEE	0x99	0x99
otilde	0xEF	0x9B	0x9B
odieresis	0xF0	0x9A	0x9A
ae	0xF1	0xBE	0xBE

ugrave	0xF2	0x9D	0x9D
uacute	0xF3	0x9C	0x9C
ucircumflex	0xF4	0x9E	0x9E
dotlessi	0xF5	0xF5	0xF5
udieresis	0xF6	0x9F	0x9F
yacute	0xF7	ÐÐ	[yacute]
lslash	0xF8	ÐÐ	[lslash]
oslash	0xF9	0xBF	0xBF
oe	0xFA	0xCF	0xCF
germandbls	0xFB	0xA7	0xA7
thorn	0xFC	ÐÐ	[thorn]
ydieresis	0xFD	0xD8	0xD8
not defined	0xFE	ÐÐ	[not defined]
ascii ctrl char	0xFF	ÐÐ	[ascii control char]

MODIFICATION HISTORY