

## Functions and Subroutines Available in Betalib.DLL

### LibVersion

*Declare Function LibVersion Lib "betalib.dll" As Integer*

Returns the version of Beta as an integer. E.g. version 2.12 would be returned at 212

### SwitchTo

*Declare Sub SwitchTo Lib "betalib.dll" (language As Integer)*

*Declare Function SwitchTo Lib "betalib.dll" (language As Integer) As Integer*

Switches the current language where:

0 = off

1 = English

2 = Greek

3 = Hebrew

SwitchTo also has the feature that it will load Beta if it is not already loaded. Beta's icon will change to indicate the current language. SwitchTo is usually used as a subroutine (i.e. the return value is ignored) but can also be used as a function in which case the previous language number is returned. Other possible values returned are: -3 invalid language, -2 failed to load Beta, -1 loaded Beta

### SetLast

*Declare Sub SetLast Lib "betalib.dll" (LastChar As Integer)*

*Declare Function SetLast Lib "betalib.dll" (LastChar As Integer) As String*

Beta always keeps a record of the last key to be pressed, so that when a new key is pressed, the pair can be checked to see if they are a valid accent combination. SetLast allows the value of the last key to be changed. This is typically set to the character before the cursor. SetLast is usually used as a subroutine (i.e. the return value is ignored) but can also be used as a function in which case a string is returned containing the version number and date.

### Combine

*Declare Function Combine Lib "betalib.dll"(Ch As Integer, Last As Integer, Lang As Integer) As int*

This and the following two functions use Beta in what is called *server mode*. The ability of Beta to intercept keystrokes is ignored, and it is used for combining and splitting accented characters. Combine returns the result of combining characters *ch* and *last*, in the language *Lang*. If the combination is invalid, then -1 is returned. The current language setting of Beta is unaffected.

### UnAccent

*Declare Function UnAccent Lib "betalib.dll" (Char As Integer, Lang As Integer) As Integer*

UnAccent attempts to remove the accent from *Char* and return the resulting vowel (using the rules for the language *Lang*). If the operation is not valid, then -1 is returned. If the vowel had two diacritics, then accents are removed before breathings.

### **UnVowel**

*Declare Function UnVowel Lib "betalib.dll" (Char As Integer, Lang As Integer) As Integer*

UnVowel attempts to remove the accent from *Char* and return it (using the rules for the language *Lang*) and return it. If the operation is not valid, then -1 is returned. If the vowel had two diacritics, then accents are returned rather than breathings.

### **Acc2Key and Key2Acc**

*Declare Function Acc2Key Lib "betalib.dll" (Acc As Integer, Lang As Integer) As Integer*

*Declare Function Key2Acc Lib "betalib.dll" (Key As Integer, Lang As Integer) As Integer*

These two functions convert between accent *keys* and their respective code points in the character set. These are not always the same, and in some cases several keys will produce the same accent. The keys on the numeric keypad: / \* - and + are represented by keys 130, 131, 132, 133. If the value is not an accent then -1 is returned.

### **Glob**

*Declare Function Glob Lib "betalib.dll" (Index As Integer, Value As Integer, Write as Integer) As Integer*

Provides global storage for wfw macros. An array of 11 integers is provided, accessed by setting index from 0 to 10. If *Write* is 0, the value of the element is returned. Otherwise "Value" is written into the element and the previous value is returned.

### **ChClass**

*Declare Function ChClass Lib "betalib.dll" (ch As Integer, Lang As Integer) As Integer*

Returns the class of character *ch* in Language *Lang* where:  
3 = combination, 2 = accent, 1 = vowel, 0 = other.  
2 is returned for combinations that are also accents.

## Word for Windows Macros

Macros are set to the following keys in Beta.dot (They may of course be changed)

F10 BetaEnglish (use the ALT key instead to get to the menu bar)  
F11 BetaGreek  
F12 BetaHebrew  
Pause BetaBackSpace  
Shift/Pause BetaMagic

=====  
BetaEnglish

*Switches back to the default font (using resetchar).*

*If the switch is from Hebrew, then an attempt is made to be intelligent about where to leave the cursor. If Hebrew was entered with the cursor at the end of the line, then it will now move back to the end of the line.*

'Switches back to the default font

```
Declare Sub SwitchTo Lib "betalib.dll"(Lang As Integer)
Declare Function SetLast Lib "betalib.dll" (LastChar As Integer) As String
Declare Function Glob Lib "betalib.dll"(Ind As Integer, Valu As Integer, Wr As Integer) As Integer
```

```
Sub MAIN
```

```
  If(LCase$(Font$()) = "hebrew") And(Glob(1, 0, 0) = 13) Then EndOfLine
```

```
  ResetChar
```

```
  SwitchTo(1)
```

```
End Sub
```

=====  
BetaGreek

*Switches to Greek and sets the size to 12 point.*

'Switches to Greek mode

```
Declare Sub SwitchTo Lib "betalib.dll" (lang As Integer)
```

```
Sub MAIN
```

```
FormatCharacter .Font = "Greek"
```

```
SwitchTo(2)
```

```
End Sub
```

=====  
BetaHebrew

*Switches to Hebrew and sets the size to 12 point.*

*In order to prevent the program switching back out of Hebrew due to right-to-left typing, the space character to the left of the insertion point is also converted to Hebrew. If there was no space character, then one is inserted.*

'Switches to Hebrew mode

```
Declare Sub SwitchTo Lib "betalib.dll" (Lang As Integer)
Declare Function Glob Lib "betalib.dll"(Ind As Integer, Valu As Integer, Wr As Integer) As Integer
```

```
Sub MAIN
  x = Glob(1, Asc(Selection$()), 1)
  Dim Form As FormatCharacter
  GetCurValues Form

  If LCase$(Form.Font) <> "hebrew" Then
    CharLeft()
    If Selection$() = " " Then
      EditClear 1
    Else
      CharRight()
    End If
    Form.Font = "Hebrew"
    FormatCharacter Form
    Insert " "
  End If
  SwitchTo(3)

End Sub
```

=====**BetaBackSpace**

*Intelligent Backspace.*

*First the macro reads the current font and sets Beta up accordingly.*

*If the language is Hebrew then a reverse backspace is performed.*

*If the language was English or Greek and the previous character was accented, then instead of backspacing off the whole character, only the accent is removed. A second backspace will of course remove the vowel as well.*

'Intelligent Backspace

```
Declare Sub SetLast Lib "betalib.dll" (LastChar As Integer)
Declare Sub SwitchTo Lib "betalib.dll" (lang As Integer)
Declare Function UnAccent Lib "betalib.dll" (Char As Integer, Lang As Integer)
As Integer
```

```
Sub MAIN
  OldFont$ = Font$()
  Select Case LCase$(Font$())
  Case "greek"
    Lang = 2
  Case "hebrew"
    Lang = 3
  Case Else
    Lang = 1
  End Select
```

```

SwitchTo(Lang)

If Lang = 3 Then
    EditClear 1
Else

CharLeft()
Last$ = Selection$()
CharRight()
SetLast(Asc(Last$))

Vowel = UnAccent(Asc(Last$), Lang)
EditClear - 1
If Vowel <> - 1 Then
    FormatCharacter .Font = OldFont$
    Insert Chr$(Vowel)
    SetLast(Vowel)
Else
    CharLeft()
    Last$ = Selection$()
    CharRight()
    SetLast(Asc(Last$))
End If
End If
End Sub

```

=====**BetaMagic**=====

*This is a very simple macro which reads the current font and previous character and sets Beta up accordingly. To use, place the cursor at the point where it is desired to insert text, and press the key associated with this macro*

'Set Beta to language at cursor

```

Declare Function SetLast Lib "betalib.dll" (LastChar As Integer) As Integer
Declare Sub SwitchTo Lib "betalib.dll" (lang As Integer)

```

```

Sub MAIN
    Select Case LCase$(Font$())
        Case "greek"
            SwitchTo(2)
        Case "hebrew"
            SwitchTo(3)
        Case Else
            SwitchTo(1)
    End Select

    CharLeft()
    Last$ = Selection$()
    CharRight()

```

```
Result = SetLast(Asc>Last$))
End Sub
```

```
=====BetaReverse
```

*Select some text and run this macro to reverse the characters.  
Does not work properly if the text spans more than one line.*

'Reverse selected text

```
Sub MAIN
  Old$ = Selection$()
  Length = Len(Old$)

  New$ = ""
  For i = 1 To Length
    New$ = Mid$(Old$, i, 1) + New$
  Next i

  EditClear
  Insert New$
End Sub
```

```
=====BetaToGreek
```

*Select some text and run this macro to convert from English text into Greek. Pairs of vowel-accent characters are converted into accented characters. This macro is useful if you need to enter Greek words but are not using Windows at the time. Later the document may be read into WfW and the Greek words converted using this macro.*

'Convert English characters to Greek, with accents

```
Declare Function Combine Lib "betalib.dll"(Ch As Integer, Last As Integer, Lang As Integer) As Integer
```

```
Sub MAIN
  Old$ = Selection$()
  Length = Len(Old$)

  New$ = ""
  Last$ = " "
  For i = 1 To Length
    Next$ = Mid$(Old$, i, 1)
    Result = Combine(Asc(Next$), Asc>Last$), 2)
    If Result < 0 Then
      New$ = New$ + Next$
      Last$ = Next$
    Else
      Last$ = Chr$(Result)
      New$ = Left$(New$, Len(New$) - 1) + Last$
    End If
  Next i
```

```
EditClear
FormatCharacter .Font = "Greek"
Insert New$
End Sub
```

```
=====BetaFromGreek
Select some text and run this macro to convert from Greek text to English. The accents are expanded out to the keys used originally to create the Greek text. This macro does the reverse of BetaToGreek. It is useful if a document needs to be converted to contain only ASCII characters, for example for email.
```

'Convert Greek characters to English, with accents

```
Declare Function UnAccent Lib "betalib.dll" (Char As Integer, Lang As Integer)
As Integer
Declare Function UnVowel Lib "betalib.dll" (Char As Integer, Lang As Integer) As
Integer
```

```
Sub MAIN
Old$ = Selection$()
Length = Len(Old$)

New$ = ""
For i = 1 To Length
Next$ = Mid$(Old$, i, 1)
Vowel = UnAccent(Asc(Next$), 2)
If Vowel < 0 Then
New$ = New$ + Next$
Else
Acc$ = Chr$(UnVowel(Asc(Next$), 2))
Vowel2 = UnAccent(Vowel, 2)
If Vowel2 < 0 Then
New$ = New$ + Chr$(Vowel) + Acc$
Else
Acc2$ = Chr$(UnVowel(Vowel, 2))
New$ = New$ + Chr$(Vowel2) + Acc2$ + Acc$
End If
End If
Next i
```

```
EditClear
ResetChar
Insert New$
End Sub
```

```
=====Overstrike
A macro for use in Hebrew. Type a consonant followed by one or more vowels / diacritics, and press Overstrike. The macro will replace character pairs with
```

*combinations where they are available, otherwise it will overstrike the characters.*

'Overstrike Hebrew consonant with diacritics

```
Declare Function Combine Lib "betalib.dll"(Ch As Integer, Last As Integer, Lang As Integer) As Integer
```

```
Declare Function ChClass Lib "betalib.dll"(ch As Integer, Lang As Integer) As Integer
```

```
Sub MAIN
```

```
MoveCount = 0
```

```
NewStr$ = ""
```

```
c$ = Selection$()
```

```
While ChClass(Asc(c$), 3) = 2
```

```
CharRight 1
```

```
MoveCount = MoveCount + 1
```

```
Result = Combine(Asc(c$), Asc(Selection$()), 3)
```

```
If Result > - 1 Then
```

```
    c$ = Chr$(Result)
```

```
Else
```

```
    NewStr$ = NewStr$ + c$
```

```
    c$ = Selection$()
```

```
End If
```

```
Wend
```

```
NewStr$ = NewStr$ + c$
```

```
If c$ = Chr$(21) Then
```

```
    MsgBox "Character already overstruck"
```

```
    Goto abort
```

```
End If
```

```
Print NewStr$
```

```
CharRight 1
```

```
CharLeft MoveCount + 1, 1
```

```
EditClear
```

```
Lenstr = Len(NewStr$)
```

```
If lenstr > 1 Then
```

```
    newchars$ = "eq \O("
```

```
    For i = 1 To Lenstr
```

```
        If(Mid$(NewStr$, i, 1) = "\") Or(Mid$(NewStr$, i, 1) = ",") Then newchars$ = newchars$ + "\"
```

```
        newchars$ = newchars$ + Mid$(NewStr$, i, 1)
```

```
        If i <> Lenstr Then newchars$ = newchars$ + ","
```

```
    Next i
```

```
    InsertField .Field = newchars$ + ")"
```

```
Else
```

```
    Insert NewStr$
```

```
End If
```



```
CharLeft 1
abort:
End Sub
```

```
=====BetaUnderdot
```

*A macro for use when typing transliterated Hebrew. The character to the left of the cursor will have a dot placed under it.*

```
'Place dot under character (used for transliterated Hebrew)
```

```
Sub MAIN
CharLeft 1, 1
c$ = Selection$()
If c$ < " " Then
  MsgBox "Not a Valid Character to Underdot"
Else
  EditClear
  InsertField .Field = "eq \O(" + c$ + ",\s\do4(.))"
End If
End Sub
```