

## QB64 Languages &amp; keyboards

**Index**

QB64 Languages & keyboards.....	2
Solution.....	3
InkeyHit project.....	3
KEYHIT operation.....	4
Reverse mapping of the CP437 page.....	6
Keyboard mapping.....	6
InkeyHit\$, an emulation of INKEY\$ for page CP1252.....	7
Tests.....	11
Project status and expectations.....	12
Layout of some keyboards that use page CP1252.....	13
Code Pages with their symbol and unicode in the center.....	15

Version	Date	Author	Comments
A01.00	05/06/20	moises1953	Alpha version. Spanish keyboard only
B01.01	12/06/20	moises1953	First results after help of RhoSigma
B02.01	19/06/20	moises1953	Added description of KEYHIT operation, Added accents table, more keyboards, restructured flowcontrol of accents. More tests. Candidate first release.

## QB64 Languages & keyboards

QB64 works with an 8-bit character set, which allows 256 different characters, while Windows processes texts in unicode (UTF-16) since 2018, thus allowing all the necessary characters for all the languages of the world, however it maintains a 8-bit local page system for different groups of nations, which is what comes to QB64, unless Windows is set to unicode. The most popular on the Internet is the UTF-8.

For occidental countries the page that Windows configure is CP1252 (USA, Canada, Latin America, Iceland, Ireland, United Kingdom, Portugal, Spain, France, Netherlands, Germany, Switzerland, Italy, Austria, Nordic countries, South Africa, Australia, New Zealand and Indonesia), while for Central and Eastern Europe the CP1250 is used (Polish, Czech, Slovak, Hungarian, Slovenian, Bosnian, Croatian, Serbian (Latin script), Romanian and Albanian).

QB64 maps the characters it will display, as if they had been encoded with the original MSDOS page, which is the extended ASCII code page CP437, looking for a character on the windows page (presumably CP1252) that looks the same or similar, for example, Ñ has the unicode code 209, which is the same as that used on page CP1252, while on CP437 it was 165, therefore it is necessary to map 209 to 165 in order for Ñ to appear, however this mapping has only been done at the presentation level, but not at the keyboard level, therefore when the Ñ is typed, in a spanish keyboard, it is not captured according to the presentation mapping, but receives 209 and therefore shows the image of 209 on the CP437, which is ±. That means, it is possible to use the mapping table in reverse to map the keyboard captures, what should have been done.

On the other hand, to access certain symbols, it is necessary to use the [AltGr] key to obtain the alternative character, which should capture the same code, but this is not the case in INKEY\$, which returns a non-unique compound code, which therefore does not allows you to map it, which is another flaw of INKEY\$. In general the INKEY\$ function does not behave like in QuickBasic:

- 1) Suffer the problems of not key mapping
  1. you cannot type local characters: Ñ ñ ç Ç ° ª ¡ ¿
  2. characters accessible with AltGr are not captured: € \ | @ # { } []
- 2) Accents do not work, therefore it is not possible to type accented letters: á é í ó ú
- 3) Alt <number> does not work, so Greek letters and other symbols are not available

However, the KEYHIT function delivers the characters according to the configured page code, whether they are a direct key, accented or with [AltGr], so in this case the corresponding code is obtained from page CP1252 and therefore does not depend on the key pressed, so it works for all keyboards that use the same Windows CP1252 page. For example, typing @ provides the code 64, which is also 64 in the CP437, and it does not matter if the keyboard has it or it was necessary to access it using [AltGr], while if you type € (euro), which is 128 in unicode and on page CP1252, but does not exist in CP437, however it can be mapped to the lowercase epsilon € of the Greek alphabet, which is similar to it and is code 238 (its position in CP437), therefore when mapping 128 generated by the € key in Windows with page CP1252, you can redirect to 238, which is where the € is on CP437. This € is the unicode 949 (U03B5), which was initially mapped to ASCII code 238, with an instruction: MAPUNICODE 949 TO 238, but which is intended to be captured from the keyboard by pressing [AlrGr] [€], which generates the code 128.

Unfortunately the KEYHIT function returns the changes of a key, so it does not solve the problem of modifiers [Alt], [Control], [Shift], nor does it allow pressing the [Alt] key to type from 0 to 255 to generate a ASCII code, therefore additional code is required to emulate the INKEY\$ and to be able to generate all the two-character codes beginning with 0, which correspond to the pressing of the modifiers [Alt], [Control], [Shift]. As you can see this manual text input in QB64 is only for keyboard, but windows can capture text from voice.

## Solution

If the CP1252 language is configured in the IDE, it works quite well except that when in text mode, the border and corner stripes of the frame are other characters, since this page does not have the frame drawing characters, but it still maps to the page CP437, so it is necessary to change the mapping using `_MAPUNICODE` instructions in the program itself, which is inconsistent.

For this solution to be complete it might be necessary:

- To be able to choose the mapping to the page between the CP437, for compatibility with the migration of QuickBasic applications, or another page for access to more regions and languages.
  - For the IDE
  - Within the application itself, at run time, making it start by default on the page that has the IDE configured
- Adapt the IDE to a graphic display mode
  - Create a graphic cursor or use Windows
  - Draw the box
- ....

However, the correct thing would be to read the code page configured in Windows and use it without any configuration. (for example `GetKeyboardLayout`).

The definitive solution will be to redo `INKEY$` capturing the queue of messages that reach the execution window, obtaining the independent translation of its origin (including virtual keyboard or voice), according to the configured code page, so that there is no need to map anything, leaving `KEYHIT` as is. In any case, it would be a matter of analyzing the `WM_CHAR` messages and performing a mapping if necessary for compatibility, but not for consistency, since what is received is what is typed, in any keyboard configuration.

It would be necessary to check how the messages of the special keys arrive, such as the function keys, and if necessary, map them to the DOS configuration to maintain compatibility.

Hopefully, eventually everything will end up in unicode, so at some point it will be necessary to address its use in QB64.

The workaround is to create a function that emulates `INKEY$` that does the reverse mapping of the key codes to the codes on the CP437 so that the corresponding key is displayed on any keyboard on the same page CP1252, and looks at the modifier keystrokes.

## ***InkeyHit project***

The initial purpose of this project is to have an `INKEY$` emulation that works in Windows for the USA and Western Europe, so that the usual local keyboards of the area covered by the Windows-1252 page can be used, while the QB64 team decides how to resolve `INKEY$` compatibility with QuickBasic.

This will require capturing the keystrokes using `KEYHIT` and mapping from each local keyboard to the codes on the page CP437 currently used by QB64 for compatibility with QuickBasic in the initial MSDOS.

Unfortunately the way that `KEYHIT` receives the dead keys (Alt, Control, Shift, accents...), and some other character, is not independent of the keyboard layout, which requires discriminating the mapping based on the connected keyboard, complicating the project, since it is not a simple mapping.

## KEYHIT operation

KEYHIT returns the one and two byte ASCII codes, the OpenGL virtual key, and the unicode code when Windows is set to unicode.

Typically the KEYHIT character codes are produced when a key is pressed: a positive one, which indicates a press or repeat, and a negative one, which indicates that the key has been released, but there are keys that only produce a negative code.

This table, which corresponds to the ASCII table, represents the key codes delivered by KEYHIT when you press the corresponding key, even with capital letters, when Windows is configured in a code page CP1250, CP1251, CP1252 ... and not in unicode:

Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key
8	BkSp	9	Tab			13	Intro			27	Esc				
32	space	33	!	34	“	35	#	36	\$	37	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
49	1	50	2	51	3	52	4	53	5	54	6	55	7	56	8
48	0	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	Y	122	z	123	{	124		125	}	126	~	127	␣

The extended table, from 128 to 255 are those of the configured code page, for example in the USA and Western Europe the CP1252, so that on a keyboard that has the €, which is the code 128 in the CP1252, KEYHIT will receive 128 when pressed and -128 when released.

### Code page extension CP1252:

Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key	Cod	Key
128	€	129		130	,	131	f	132	„	133	...	134	†	135	‡
136	ˆ	137	%	138	š	139	<	140	œ	141		142	ž	143	
144		145	‘	146	’	147	“	148	”	149	•	150	–	151	—
152	˜	153	™	154	š	155	>	156	œ	157		158	ž	159	ÿ
160		161	ı	162	¢	163	£	164	¤	165	¥	166	ı	167	§
168	¨	169	©	170	ª	171	«	172	¬	173		174	®	175	¯
176	°	177	±	178	²	179	³	180	´	181	µ	182	¶	183	·
184	¸	185	¹	186	º	187	»	188	¼	189	½	190	¾	191	¿
192	À	193	Á	194	Â	195	Ã	196	Ä	197	Å	198	Æ	199	Ç
200	È	201	É	202	Ê	203	Ë	204	Ì	205	Í	206	Î	207	Ï
208	Ð	209	Ñ	210	Ò	211	Ó	212	Ô	213	Õ	214	Ö	215	×
216	Ø	217	Ù	218	Ú	219	Û	220	Ü	221	Ý	222	Þ	223	ß
224	à	225	á	226	â	227	ã	228	ä	229	å	230	æ	231	ç
232	è	233	é	234	ê	235	ë	236	ì	237	í	238	î	239	ï
240	ð	241	ñ	242	ò	243	ó	244	ô	245	õ	246	ö	247	÷
248	ø	249	ù	250	ú	251	û	252	ü	253	ý	254	þ	255	ÿ

These codes are generated when pressing a live key (that represents a character), being able to repeat if it is kept pressed, but not when releasing it, on the contrary the dead keys such as caps, control, alt and accents, only generate codes when releasing it and therefore isolated negatives.

Dead keys also generate virtual key codes, with an absolute value greater than 100,000, both when pressing and releasing. Are 2 types of dead keys:

1. Modifiers: Shift, Control, Alt: -16, -17, -18
2. Accents: acute, serious, circumflex, umlaut ... (they vary according to the keyboard layout)

An important difference between INKEY\$ and KEYHIT is how they work when using the CTRL, ALT, or SHIFT modifier keys. INKEY\$ returns a different code if you hold down the CTRL, ALT, or SHIFT key before pressing a key, while KEYHIT will only deliver different codes with the capitals of the basic ASCII code, for the rest it will return the same code regardless of which modifiers are used, therefore it was necessary to use KEYDOWN to see which modifier keys were pressed when receiving a key code. When the [AltGr] key is pressed, KEYHIT returns the codes Alt (100307) and Ctrl (100306), and the same negatives when released.

The keys that generate 2-byte codes where the first is CHR\$ (0) are (second byte):

1. Function keys: F1 to F10
  1. Alone: 59 to 68.; <=>? @ ABCD
  2. Modified (must be added): [Alt]: + 45 [Ctrl]: + 35 [Shift]: + 25
2. Function keys: F11, F12
  1. Alone: 133, 134
  2. Modified (must be added): [Alt]: + 6 [Ctrl]: + 4 [Shift]: + 2
3. Scroll keys: Start, ↑, RePag, ←, →, End, ↓, AvPag, Insert, Delete
  1. Alone: 71,72,73,75,77,79,80,81,82,83 GHIKMOPQRS
  2. Ctrl (add): Start, RePag, ←, →, End, AvPag: 71,73,75,77,79,81 wästuv

Since KEYHIT does not combine the modifier keys [Alt] [Ctrl] and some [Shift], it will be necessary to emulate Quickbasic's INKEY\$, and in the same way combine the accents keys that are applied depending on the keyboard.

Codes for some dead key accents under your keyboards. Each column represents the negative code delivered by KEYHIT when using the keyboard indicated in the row.

Teclado	KBID	-186	-187	-191	-192	-219	-220	-221	-222		
Danish (da-DK)	1030	^"				´					
German (de-DE)	1031				¨		^	´			
Swiss (de-CH)	2055				¨	´		´^			
Spanish (es-ES)	1034	^`							´´		
French (fr-FR)	1036				´			¨^			
Belgian (fr-BE)	2060				´		´	¨^			
Swiss (fr-CH)	4108										
Portugues (pt-PT)	1040	´´	¨	^							
Neerland (nl-NL)	1043				´´			^¨			
Swedish (sv-SE)	1053	^"				´´					

It can be seen that the same key does not generate the same code, therefore it will be necessary to make a large control block to map accented letters taking into account the keyboard.





```

hit = KEYHIT
IF hit THEN
  car = 0
  keyshift = KEYDOWN(100303) OR KEYDOWN(100304)
  keyctrl = KEYDOWN(100305) OR KEYDOWN(100306)
  keyAltGr = KEYDOWN(100307) AND KEYDOWN(100306)
  keyalt = (KEYDOWN(100307) OR KEYDOWN(100308)) AND NOT keyAltGr

IF hit > 0 THEN
  IF hit < 256 THEN lastKey = hit
  IF hit >=32 AND hit < 123 THEN
    SELECT CASE prekey
      CASE PkAcute '
        p = INSTR(AcuteLook, CHR$(hit))
        IF p THEN car = ASC(AcuteMapi, p)
      CASE PkGrave '`
        p = INSTR(GraveLook, CHR$(hit))
        IF p THEN car = ASC(GraveMapi, p)
      CASE PkUmlau
        p = INSTR(UmlauLook, CHR$(hit))
        IF p THEN car = ASC(UmlauMapi, p)
      CASE PkCircu
        p = INSTR(CircuLook, CHR$(hit))
        IF p THEN car = ASC(CircuMapi, p)
    END SELECT
  END IF

IF car THEN
  prekey = 0
ELSE
  '--- control sequences and special behavior ---
  SELECT CASE hit
    CASE 9 'tab
      IF keyshift THEN dblcar = CHR$(0) + CHR$(15) ELSE car = hit
    CASE 48 TO 57 'numeric heys 0-9
      IF keyalt = 0 THEN car = hit
    CASE 65 TO 90 'CTRL CAPS A-Z: 1-26
      IF keyctrl THEN car = hit - 64 ELSE car = hit
    CASE 97 TO 122 'CTRL a-z: 1-26
      IF keyctrl THEN car = hit - 96 ELSE car = hit
    CASE 0 TO 127 'ASCII
      car = hit
    CASE 128 TO 255
      '--- bring the system codepage mapped inputs back to Cp437, if available ---
      p = INSTR(KeyLook, CHR$(hit))
      IF p THEN car = ASC(KeyMapi, p) ELSE car = hit
    CASE 256 TO 65535 'double byte chr$(0)+
      dblcar = MKI$(hit)
      IF ASC(dblcar) = 0 THEN
        car = ASC(dblcar, 2)
        SELECT CASE car 'priority ordering (Alt -> Ctrl -> Shift)
          CASE 59 TO 68 'F1-F10
            IF keyalt THEN
              MID$(dblcar, 2) = CHR$(car + 45)
            ELSEIF keyctrl THEN
              MID$(dblcar, 2) = CHR$(car + 35)
            ELSEIF keyshift THEN
              MID$(dblcar, 2) = CHR$(car + 25)
            END IF
          CASE 133, 134 'F11-F12
            IF keyalt THEN
              MID$(dblcar, 2) = CHR$(car + 6)
            ELSEIF keyctrl THEN
              MID$(dblcar, 2) = CHR$(car + 4)
            ELSEIF keyshift THEN
              MID$(dblcar, 2) = CHR$(car + 2)
            END IF
          CASE 71 'Home
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(119) 'w
          CASE 73 'RePag
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(132) '.,
          CASE 75 'Left
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(115) 's
          CASE 77 'Right
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(116) 't
          CASE 79 'End
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(117) 'u
          CASE 81 'AvPag
            IF keyctrl THEN MID$(dblcar, 2) = CHR$(118) 'v
        END SELECT
      END SELECT
  END IF

```

```

        IF CVI(dblcar) THEN
            Inkeyhit$ = dblcar
            prekey = 0
            lastKey = 0
        END IF
        car = 0
        CASE IS >= &H40000000 'unicode (someday)
            hitu = hit - &H40000000 '4 bytes
        END SELECT
    END IF 'car
ELSE 'hit<0
    kb1 = kbdLayout
    SELECT CASE hit
        CASE -57 TO -48 'character code: numeric keys, also numeric keypad with numlock
            IF keyAlt THEN
                IF LEN(number$) > 2 THEN number$ = RIGHT$(number$, 2)
                number$ = number$ + CHR$(ABS(hit))
            ELSEIF hit = -50 AND keyAltGr THEN 'fr-FR ~
                car = 126
            END IF
        CASE -100308 'Alt up: capture character code
            IF LEN(number$) THEN
                car = VAL(number$)
                number$ = ""
            END IF
        CASE -lastKey 'cancel lastkey
            lastKey = 0
            'special keys that only release
            'including accents. prekey: 1-acute, 2-grave, 3-umlaut, 4-circumflex
        CASE -186 'es-ES pt-PT da-DK & sv-SE
            SELECT CASE kb1
                CASE kbPtPt 'pt-PT: grave & acute accent
                    IF keyshift THEN
                        prekey = PkGrave
                    ELSE
                        prekey = PkAcute
                    END IF
                CASE kbEsEs 'es-ES: circumflex & grave accent
                    IF keyshift THEN
                        prekey = PkCircu
                    ELSE
                        prekey = PkGrave
                    END IF
                CASE ELSE 'da-DK & sv-SE: circumflex & umlaut
                    IF keyshift THEN
                        prekey = PkCircu
                    ELSE
                        prekey = PkUmlau
                    END IF
            END SELECT
        CASE -187 'pt-PT: umlaut
            prekey = PkUmlau
        CASE -191 'pt-PT: circumflex
            prekey = PkCircu
        CASE -192 'nl-NL de-CH & fr-FR fr-BE
            SELECT CASE kb1
                CASE kbNlNl 'nl-NL: grave & acute accent
                    IF keyshift THEN
                        prekey = PkGrave
                    ELSE
                        prekey = PkAcute
                    END IF
                CASE kbDeCh 'de-CH: umlaut
                    prekey = PkUmlau
                CASE ELSE 'fr-FR fr-BE: acute
                    prekey = PkAcute
            END SELECT
        CASE -219 'de-CH da-DK & sv-SE
            IF kb1 = kbDeCh THEN 'de-CH: acute
                prekey = PkAcute
            ELSE 'da-DK & sv-SE: grave & acute accent
                IF keyshift THEN
                    prekey = PkGrave
                ELSE
                    prekey = PkAcute
                END IF
            END IF
        CASE -220 'fr-BE & de-DU
            IF kb1 = kbFrBe THEN 'fr-BE: grave
                prekey = PkGrave
            ELSE 'de-DE: circumflex

```

```

    prekey = PkCircu
  END IF
CASE -221 'fr-FR fr-BE n1-NL & de-DE
  SELECT CASE kb1 'keyboard layout
    CASE kbFrFr, kbFrBe 'fr-FR: umlaut & circumflex
      IF keyshift THEN
        prekey = PkUmlau
      ELSE
        prekey = PkCircu
      END IF
    CASE kbN1N1 'n1-NL: circumflex & umlaut
      IF keyshift THEN
        prekey = PkCircu
      ELSE
        prekey = PkUmlau
      END IF
    CASE kbDeCh 'de-CH grave & circumflex
      IF keyshift THEN
        prekey = PkGrave
      ELSE
        prekey = PkCircu
      END IF
    CASE ELSE 'de-DE : grave & acute accent
      IF keyshift THEN
        prekey = PkGrave
      ELSE
        prekey = PkAcute
      END IF
  END SELECT
CASE -222 'es-ES: umlaut & acute accent; Spanish (di, resis)
  IF keyshift THEN
    prekey = PKUmlau
  ELSE
    prekey = PKAcute
  END IF
CASE -226 'pt-PT: \ ?
  car = 92
END SELECT 'hit
END IF 'hit>0
IF car THEN
  Inkeyhit$ = CHR$(car)
  prekey = 0
END IF
END IF 'hit
END FUNCTION

```

**Tests**

The tests of the InkeyHit\$ function have been carried out with the simulation of some keyboards that use the code page CP1252, in a Windows 10 in Spanish language with a Spanish keyboard, configuring several keyboards and preferred languages, making the keyboard changes in the task bar, paying attention to the following groups of keys:

1. Alphabet keys, upper and lower case.
2. Numbers of the main keyboard and the numeric
3. Punctuation marks: ! ' , . - : ; \_
4. National letters in keyboard: ñÑçÇÜ
5. Symbols [AltGr]: € \ | @ # { } [
6. Accented letters using accent: áéíóúÿää
7. Function keys, except [Alt] [F4]
8. Scroll keys
9. ASCII code: [Alt] number
10. Control characters: [Control] a-z

The summary is recorded in this table, where

OK means it works, - not aplicable because there are no such keys, and 'no' means it doesn't work:

Teclado	Code	1.Alfa	2.Num	3.Punt	4.Nat.	5.Altgr	6.Ace	7.Fxx*	8.Scol	9.Alt-n	10.Ctrl
Dansk (da-DK)	1030	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
German (de-DE)	1031	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
Swiss (de-CH)	2055	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
English (en-US)	1033	OK	OK	OK	-	-	-	OK	OK	OK	OK
English (en-GB)	2057	OK	OK	OK	OK	OK <sup>1</sup>	-	OK	OK	OK	OK
Español (es-ES)	1034	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
Français (fr-FR)	1036	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
Belge (fr-BE)	2060	OK	OK	OK	OK	OK	OK <sup>2</sup>	OK	OK	OK	OK
Swiss (fr-CH)	4108	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
Italliano (it-IT)	1040	OK	OK	OK	OK	OK	-	OK	OK	OK	OK
Neerland (nl-NL)	1043	OK	OK	OK	OK	OK	OK <sup>2</sup>	OK	OK	OK	OK
Swedish (sv-SE)	1053	OK	OK	OK	OK	OK	OK <sup>2</sup>	OK	OK	OK	OK

\* [Alt] F4 closes the application, cannot be used.

<sup>1</sup> Except for [AltGr] AIOU → ´, in CP437 there is no ÁÍÓÚ, perhaps can be replaced by lowercase.

<sup>2</sup> Incomplete: missing: ~ (accent, there are no vowels with that accent in CP437)

'Keys with only negative codes (not preceded by a positive one) depend on the keyboard layout. At the moment it will be identified by GetKeyboardLayout, which does not detect the changes by the taskbar until the program is restarted, while the mapping does.

**Idioma**

Idiomas preferidos

Las aplicaciones y los sitios web aparecerán en el primer idioma de la lista que admitan. Selecciona un idioma y, a continuación, selecciona Opciones para configurar los teclados y otras características.

The screenshot shows the Windows 'Idiomas preferidos' (Preferred Languages) settings. At the top, there is a button to 'Agregar un idioma preferido'. Below, a list of installed languages is shown, each with a flag icon and a 'Paquete de idioma instalado' label. The languages listed are: Español (España), English (United States), English (United Kingdom), Français (France), Deutsch (Deutschland), and Italiano (Italia). To the right of each language name are icons for adding, removing, and setting as default.

## Project status and expectations

The project is in beta candidate phase. The author will use it in his own project.

The tests have been carried out by the author with a Windows 10 in Spanish (es-ES) and also configuring da-DK, de-DE, de-CH, en-US, en-GB, fr-FR, fr-BE, fr-CH, It-IT, nl-NL, sv-SE, configuring preferred languages and switching between them through the taskbar, and RhoSigma from the QB64 forum, in German (de-DE), but more tests are needed with more original keyboards with the purpose of both discover defects or validate, such as proposing new keyboards of the target area, such as Irish or Norwegian, for which it is necessary to discover the codes of the silent keys that determine accents, other than ~, and complete the table to facilitate the programing.

We want to receive help and requests in the following areas:

- 1) Test results: Specify operating system, language and keyboard, as well as results, at least in the categories of the table used in the test chapter, with the pertinent notes.
- 2) Proposals for the inclusion of languages or keyboards, on condition that they provide the codes of the silent keys that generate accented letters, and cooperate in the tests.
- 3) Proposals for methodological improvements, such as a method to find more mappable characters than the initial 54, and the 4 added ones.
- 4) Proposals for substituting characters in the mapping, either because they are missing, or because they are more similar, more common, etc ...
- 5) Proposals to improve the operation of the code, or its maintainability
- 6) Optimization proposals.
- 7) Others.

The expectation is to cover all the occidental countries that use the CP1252.

## Tools

Keyhit-Test.bas shows the input codes of KEYHIT function, and the mapping applied. Allows to see codes of your keyboard.

CP437 Reverse table-v0100.bas shows the reverse mapping extracted of the actual page using the Mapunicode function.

**Layout of some keyboards that use page CP1252.**

Qwerty España:  
es-ES: 1034

~	!	"	#	\$	%	&	/	(	)	=	?	^	←
1	2	3	4	5	6	7	8	9	0	'	i	Backspace	
Tab	Q	W	E	R	T	Y	U	I	O	P	{	}	Enter
↔	Q	W	E	R	T	Y	U	I	O	P	{	}	↔
↕	A	S	D	F	G	H	J	K	L	Ñ	Ç	↕	
↕	↕	A	S	D	F	G	H	J	K	L	Ñ	Ç	↕
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕	
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕	
Ctrl	Win Key	Alt							Alt Gr	Win Key	Menu	Ctrl	

Qwerty US:  
en-US: 1033

~	!	@	#	\$	%	^	&	*	(	)	-	+	←
1	2	3	4	5	6	7	8	9	0	-	=	Backspace	
Tab	Q	W	E	R	T	Y	U	I	O	P	{	}	Enter
↔	Q	W	E	R	T	Y	U	I	O	P	{	}	↔
↕	A	S	D	F	G	H	J	K	L	:	"	↕	
↕	↕	A	S	D	F	G	H	J	K	L	:	"	↕
↕	↕	Z	X	C	V	B	N	M	<	>	?	↕	
↕	↕	Z	X	C	V	B	N	M	<	>	?	↕	
Ctrl	Win Key	Alt							Alt	Win Key	Menu	Ctrl	

Qwerty UK:  
en-EN: 2057

~	!	"	£	\$	%	^	&	*	(	)	-	+	←
1	2	3	4	5	6	7	8	9	0	-	=	Backspace	
Tab	Q	W	E	R	T	Y	U	I	O	P	{	}	Enter
↔	Q	W	E	R	T	Y	U	I	O	P	{	}	↔
↕	A	S	D	F	G	H	J	K	L	:	@	↕	
↕	↕	A	S	D	F	G	H	J	K	L	:	@	↕
↕	↕	Z	X	C	V	B	N	M	<	>	?	↕	
↕	↕	Z	X	C	V	B	N	M	<	>	?	↕	
Ctrl	Win Key	Alt							Alt Gr	Win Key	Menu	Ctrl	

Qwertz Deutch:  
& Austrian  
de-DE: 1031

°	!	"	§	\$	%	&	/	(	)	=	?	^	←
1	2	3	4	5	6	7	8	9	0	'	ü	↕	
↔	Q	W	E	R	T	Z	U	I	O	P	Ü	*	↔
↔	Q	W	E	R	T	Z	U	I	O	P	Ü	*	↔
↕	A	S	D	F	G	H	J	K	L	Ö	Ä	↕	
↕	↕	A	S	D	F	G	H	J	K	L	Ö	Ä	↕
↕	↕	Y	X	C	V	B	N	M	;	:	-	↕	
↕	↕	Y	X	C	V	B	N	M	;	:	-	↕	
Strg	(Wn)	Alt							AltGr	(Wn)	(Menu)	Strg	

Qwerty Italiano:  
it-IT: 1040

!	!	"	£	\$	%	&	/	(	)	=	?	^	←	
1	2	3	4	5	6	7	8	9	0	'	i	↕		
Tab	Q	W	E	R	T	Y	U	I	O	P	é	*	↔	
↔	Q	W	E	R	T	Y	U	I	O	P	é	*	↔	
↕	A	S	D	F	G	H	J	K	L	Ç	°	§	↕	
↕	↕	A	S	D	F	G	H	J	K	L	Ç	°	§	↕
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕		
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕		
Ctrl	Tasto Win	Alt							Alt Gr	Tasto Win	Menu	Ctrl		

Qwerty Portuguese:  
pt-PT: 2070

!	!	"	#	\$	%	&	/	(	)	=	?	»	←
1	2	3	4	5	6	7	8	9	0	'	«	Backspace	
Tab	Q	W	E	R	T	Y	U	I	O	P	*	»	↔
↔	Q	W	E	R	T	Y	U	I	O	P	*	»	↔
↕	A	S	D	F	G	H	J	K	L	Ç	»	↕	
↕	↕	A	S	D	F	G	H	J	K	L	Ç	»	↕
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕	
↕	↕	Z	X	C	V	B	N	M	;	:	-	↕	
Ctrl	Win Key	Alt							Alt Gr	Win Key	Menu	Ctrl	



### Code Pages with their symbol and unicode in the center

CP437

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-	FSP 2007 0	© 263A 1	⊙ 263B 2	▼ 2665 3	♠ 2666 4	♣ 2663 5	♠ 2660 6	• 2022 7	▪ 25D8 8	◦ 25CB 9	▪ 25D9 10	♂ 2642 11	♀ 2640 12	♫ 266A 13	♫ 266B 14	☼ 263C 15
1-	▶ 25BA 16	◀ 25C4 17	↑ 2195 18	!! 203C 19	¶ 00B6 20	§ 00A7 21	— 25AC 22	‡ 21A8 23	↑ 2191 24	↓ 2193 25	→ 2192 26	← 2190 27	↔ 221F 28	↔ 2194 29	▲ 25B2 30	▼ 25BC 31
2-	SP 0020 32	! 0021 33	" 0022 34	# 0023 35	\$ 0024 36	% 0025 37	& 0026 38	' 0027 39	( 0028 40	) 0029 41	* 002A 42	+ 002B 43	, 002C 44	- 002D 45	. 002E 46	/ 002F 47
3-	0 0030 48	1 0031 49	2 0032 50	3 0033 51	4 0034 52	5 0035 53	6 0036 54	7 0037 55	8 0038 56	9 0039 57	: 003A 58	; 003B 59	< 003C 60	= 003D 61	> 003E 62	? 003F 63
4-	@ 0040 64	A 0041 65	B 0042 66	C 0043 67	D 0044 68	E 0045 69	F 0046 70	G 0047 71	H 0048 72	I 0049 73	J 004A 74	K 004B 75	L 004C 76	M 004D 77	N 004E 78	O 004F 79
5-	P 0050 80	Q 0051 81	R 0052 82	S 0053 83	T 0054 84	U 0055 85	V 0056 86	W 0057 87	X 0058 88	Y 0059 89	Z 005A 90	[ 005B 91	\ 005C 92	] 005D 93	^ 005E 94	_ 005F 95
6-	` 0060 96	a 0061 97	b 0062 98	c 0063 99	d 0064 100	e 0065 101	f 0066 102	g 0067 103	h 0068 104	i 0069 105	j 006A 106	k 006B 107	l 006C 108	m 006D 109	n 006E 110	o 006F 111
7-	p 0070 112	q 0071 113	r 0072 114	s 0073 115	t 0074 116	u 0075 117	v 0076 118	w 0077 119	x 0078 120	y 0079 121	z 007A 122	{ 007B 123	 007C 124	}	~ 007E 126	△ 2302 127
8-	Ç 00C7 128	ü 00FC 129	é 00E9 130	â 00E2 131	ä 00E4 132	à 00E0 133	â 00E5 134	ç 00E7 135	ê 00EA 136	ë 00EB 137	è 00E8 138	ï 00EF 139	î 00EE 140	ï 00EC 141	Ä 00C4 142	Å 00C5 143
9-	É 00C9 144	æ 00E6 145	Æ 00C6 146	ô 00F4 147	ó 00F6 148	ò 00F2 149	û 00FB 150	ù 00F9 151	ÿ 00FF 152	Ö 00D6 153	Ü 00DC 154	¢ 00A2 155	£ 00A3 156	¥ 00A5 157	₣ 20A7 158	ƒ 0192 159
A-	á 00E1 160	í 00ED 161	ó 00F3 162	ú 00FA 163	ñ 00F1 164	Ñ 00D1 165	ª 00AA 166	º 00BA 167	¿ 00BF 168	ƒ 2310 169	¬ 00AC 170	½ 00BD 171	¼ 00BC 172	¡ 00A1 173	« 00AB 174	» 00BB 175
B-	☼ 2591 176	☼ 2592 177	☼ 2593 178	 2502 179	 2524 180	 2561 181	 2562 182	¶ 2556 183	¶ 2555 184	¶ 2563 185	¶ 2551 186	¶ 2557 187	¶ 255D 188	¶ 255C 189	¶ 255B 190	¶ 2510 191
C-	L 2514 192	L 2534 193	T 252C 194	 251C 195	— 2500 196	† 253C 197	† 255E 198	† 255F 199	ℒ 255A 200	ℒ 2554 201	ℒ 2569 202	ℒ 2566 203	ℒ 2560 204	= 2550 205	† 256C 206	± 2567 207
D-	ℒ 2568 208	ℒ 2564 209	ℒ 2565 210	ℒ 2559 211	ℒ 2558 212	ℒ 2552 213	ℒ 2553 214	ℒ 256B 215	ℒ 256A 216	ℒ 2518 217	ℒ 250C 218	■ 2588 219	■ 2584 220	■ 258C 221	■ 2590 222	■ 2580 223
E-	α 03B1 224	β 03B2 225	Γ 0393 226	π 03C0 227	Σ 03A3 228	σ 03C3 229	μ 00B5 230	τ 03C4 231	Φ 03A6 232	Θ 0398 233	Ω 03A9 234	δ 03B4 235	∞ 221E 236	∅ 2205 237	€ 2208 238	∩ 2229 239
F-	≡ 2261 240	± 00B1 241	≥ 2265 242	≤ 2264 243	∫ 2320 244	∫ 2321 245	÷ 00F7 246	≈ 2248 247	° 00B0 248	· 2219 249	· 00B7 250	√ 221A 251	ⁿ 207F 252	² 00B2 253	■ 25A0 254	NBSP 00A0 255

CP1252

Windows-1252 (CP1252)

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_0	NUL 0000 0	SOH 0001 1	STX 0002 2	ETX 0003 3	EOT 0004 4	ENQ 0005 5	ACK 0006 6	BEL 0007 7	BS 0008 8	HT 0009 9	LF 000A 10	VT 000B 11	FF 000C 12	CR 000D 13	SO 000E 14	SI 000F 15
1_16	DLE 0010 16	DC1 0011 17	DC2 0012 18	DC3 0013 19	DC4 0014 20	NAK 0015 21	SYN 0016 22	ETB 0017 23	CAN 0018 24	EM 0019 25	SUB 001A 26	ESC 001B 27	FS 001C 28	GS 001D 29	RS 001E 30	US 001F 31
2_32	SP 0020 32	! 0021 33	" 0022 34	# 0023 35	\$ 0024 36	% 0025 37	& 0026 38	' 0027 39	( 0028 40	) 0029 41	* 002A 42	+ 002B 43	, 002C 44	- 002D 45	. 002E 46	/ 002F 47
3_48	0 0030 48	1 0031 49	2 0032 50	3 0033 51	4 0034 52	5 0035 53	6 0036 54	7 0037 55	8 0038 56	9 0039 57	: 003A 58	; 003B 59	< 003C 60	= 003D 61	> 003E 62	? 003F 63
4_64	@ 0040 64	A 0041 65	B 0042 66	C 0043 67	D 0044 68	E 0045 69	F 0046 70	G 0047 71	H 0048 72	I 0049 73	J 004A 74	K 004B 75	L 004C 76	M 004D 77	N 004E 78	O 004F 79
5_80	P 0050 80	Q 0051 81	R 0052 82	S 0053 83	T 0054 84	U 0055 85	V 0056 86	W 0057 87	X 0058 88	Y 0059 89	Z 005A 90	[ 005B 91	\ 005C 92	] 005D 93	^ 005E 94	_ 005F 95
6_96	` 0060 96	a 0061 97	b 0062 98	c 0063 99	d 0064 100	e 0065 101	f 0066 102	g 0067 103	h 0068 104	i 0069 105	j 006A 106	k 006B 107	l 006C 108	m 006D 109	n 006E 110	o 006F 111
7_112	p 0070 112	q 0071 113	r 0072 114	s 0073 115	t 0074 116	u 0075 117	v 0076 118	w 0077 119	x 0078 120	y 0079 121	z 007A 122	{ 007B 123	 007C 124	}	~ 007D 125	DEL 007E 126
8_128	€ 20AC 128		¸ 201A 130	ƒ 0192 131	„ 201E 132	… 2026 133	† 2020 134	‡ 2021 135	^ 02C6 136	% 2030 137	Š 0160 138	< 2039 139	Œ 0152 140		Ž 017D 142	
9_144		ˆ 2018 145	˜ 2019 146	“ 201C 147	” 201D 148	• 2022 149	– 2013 150	— 2014 151	~ 02DC 152	™ 2122 153	š 0161 154	> 203A 155	œ 0153 156		ž 017E 158	ÿ 0178 159
A_160	NBSP 00A0 160	ı 00A1 161	¢ 00A2 162	£ 00A3 163	¤ 00A4 164	¥ 00A5 165	¦ 00A6 166	§ 00A7 167	¨ 00A8 168	© 00A9 169	ª 00AA 170	« 00AB 171	¬ 00AC 172	SHY 00AD 173	® 00AE 174	¯ 00AF 175
B_176	° 00B0 176	± 00B1 177	² 00B2 178	³ 00B3 179	´ 00B4 180	µ 00B5 181	¶ 00B6 182	· 00B7 183	¸ 00B8 184	¹ 00B9 185	º 00BA 186	» 00BB 187	¼ 00BC 188	½ 00BD 189	¾ 00BE 190	¿ 00BF 191
C_192	À 00C0 192	Á 00C1 193	Â 00C2 194	Ã 00C3 195	Ä 00C4 196	Å 00C5 197	Æ 00C6 198	Ç 00C7 199	È 00C8 200	É 00C9 201	Ê 00CA 202	Ë 00CB 203	Ì 00CC 204	Í 00CD 205	Î 00CE 206	Ï 00CF 207
D_208	Ð 00D0 208	Ñ 00D1 209	Ò 00D2 210	Ó 00D3 211	Ô 00D4 212	Õ 00D5 213	Ö 00D6 214	× 00D7 215	Ø 00D8 216	Ù 00D9 217	Ú 00DA 218	Û 00DB 219	Ü 00DC 220	Ý 00DD 221	Þ 00DE 222	ß 00DF 223
E_224	à 00E0 224	á 00E1 225	â 00E2 226	ã 00E3 227	ä 00E4 228	å 00E5 229	æ 00E6 230	ç 00E7 231	è 00E8 232	é 00E9 233	ê 00EA 234	ë 00EB 235	ì 00EC 236	í 00ED 237	î 00EE 238	ï 00EF 239
F_240	ð 00F0 240	ñ 00F1 241	ò 00F2 242	ó 00F3 243	ô 00F4 244	õ 00F5 245	ö 00F6 246	÷ 00F7 247	ø 00F8 248	ù 00F9 249	ú 00FA 250	û 00FB 251	ü 00FC 252	ý 00FD 253	þ 00FE 254	ÿ 00FF 255

Letter  Number  Punctuation  Symbol  Other  Undefined  Differences from ISO-8859-1