[*Machine Translation Today: Translating and the Computer 15*. Papers at a conference…18-19 November 1993 (London: Aslib)]

FOREIGN LANGUAGES IN WORDPERFECT

Peter Kahrel
Marsstraat 44, 2024 GE Haarlem, The Netherlands

WordPerfect offers several facilities to handle foreign languages and multi-lingual documents. This paper discusses two aspects of language handling in WP: the language code, which is a WP formatting code that gives access to language modules and the keyboard editor, which facilitates entering foreign characters.
The paper discusses the possibilities offered in the 5.1 version of the program. The last section discusses improvements in WP 6.0.

INTRODUCTION

WordPerfect (WP) offers a number of facilities to handle foreign languages and foreign characters. In this paper, I will concentrate on two of the major aspects of foreign language handling, namely the language code and its implications for hyphenation and spell checking, and the keyboard editor, which is essential for those who frequently need characters not represented on the keyboard. I will also show how to create a keyboard layout that is sensitive to the language code. Such a keyboard layout is convenient for typing multi-lingual documents.

The WP facilities discussed here and the techniques outlined to facilitate typing accented characters hold for WP version 5.1. Recently, however, a new version of the WP program has been released, WP 6.0. Though the linguistic facilities of WP 6.0 are essentially the same as in WP 5.1, they are more sophisticated in some respects. But since the majority of the users will still be using WP 5.1, and since there were no language modules available at the time of writing, I will concentrate on version 5.1 Any relevant changes and improvements in WP 6.0 will be discussed in the last section.

To designate keystrokes required in WP, I will use the following conventions. Two keystrokes separated with a hyphen mean that you press the first, hold it down and then press the second. For example, Shift-F8 means that you press the Shift key, hold it down, and then press the F8 key. Keys separated by a comma mean that you press the keys one after the other. For example, Home, Enter means that you should press the Home key, release it, then press the Enter key.

THE LANGUAGE CODE

The basis for WP's ability to handle foreign languages is the language module. A language module consists of a word list which is used by the spell checker and a hyphenation file which is consulted when hyphenation is enabled. In many cases, a language module also includes a thesaurus (a dictionary that you can use to look up synonyms and antonyms), a keyboard driver to facilitate typing some special characters and a screen font to display characters not contained in the standard IBM character set. For example, the Hungarian language module includes a screen driver to display the ő and the ű. Each version of WP includes the language module of the package language. For example, the English version comes with the English language module and the German version with the German language module. Language modules can be bought separately.

Within WP, a language module is accessed by using the **language code**. This code determines which language module WP will use after the point where it is inserted. For example, if you enter the French language code in a document, WP will use the French dictionary for spell checking the document and the French hyphenation module to hyphenate words from that point onwards. Also, when you spell check a document and you add words to the additional word list, these words will be added to the French word list

Like other WP codes, the language code is a formatting code. You enter it as follows: press Shift-F8, 4, 5 (Format, Language, Other) and type the language abbreviation (for example, FR for French, UK for British English, US for American English). Then press Enter until you are back at the edit screen. Since the language code is a formatting code like any other code, it can be inserted into a document in any place and as frequently as is necessary. And you can enter as many different language codes as you have language modules. Thus, it can be used in an alternating French-English document, but also in 'EC-documents' which contain all EC languages.

The spell checker and the hyphenation module are well documented (see for example Kahrel (1)). I will therefore not discuss them here, but rather move on to keyboards and typing accented characters.

THE KEYBOARD

Computer keyboards contain only a limited number of keys. Keyboards for use in the US and in Britain contain only the twenty-six base letters (ie, unaccented letters). In some countries, keyboards contain some accented characters; but no keyboard contains all accented letters. WP, in contrast, knows about 1900 characters distributed over 11 character sets, and by using the special Compose feature, all these characters can be typed with relative ease using the limited number of keys on the keyboard. Even characters not included in any character set can be typed using the Overstrike feature. Below I will discuss the notions character set, Compose key and Overstrike in detail.

Character sets

All the WP characters are contained in character sets. For example, character set 0 contains the 'standard' characters, ie the characters you also find on the keyboard. Character set 1 contains a number of floating accents (accents without a letter, such as ¯ ` ´ ¸ ˛) and a large number of accented characters. Other character sets contain Greek, Hebrew, Arabic, Cyrillic, Japanese, typographical symbols and mathematical symbols. Overviews of the other character sets can be found in the WP manual and most books on WP. Kahrel (1) also discusses some inconsistencies in set 1 and how they can be solved. For the purposes of this paper I will concentrate on character sets 1 and 2.

Each character is identified by the character set number and the position within that character set. For example, the ą is character 95 in character set 1. By convention, characters are designated as set, number. Thus, the ą is defined as character 1,95. In the remainder of this paper I will use this convention.

Typing accented characters; the Compose key

Using the Compose key, you can type any character by entering its character set number and its position. To activate the Compose key, press Ctrl-V. At the bottom of the screen you see the prompt **Key =**, at which you enter the character's number. For example, to type the ą, press Ctrl-V and type 1,95 followed by Enter. In this way, any WP character can be entered.

Entering characters in the above way is of course awkward, since nobody will be able to memorize each character's number. WP therefore allows you to use characters in the Compose key rather than numbers. Normal letters you type as such, while a number of accents are represented by a convention. For example, at the Compose key, WP interprets the comma as the cedilla and the ^ as the circumflex accent. Thus, to type the ç, press Ctrl-V and type ,c followed by Enter. And to type the ř, press Ctrl-V and type vr followed by Enter (the v designates the ˇ, the hacek accent). The order in which you type the accent and the character is immaterial.

Table 1 lists which keys are recognized as accents at the Compose key and which characters can be typed. The first column gives the keys representing the accents in the second column. So you see that since the semicolon represents the ogonek (the Polish hook), you type ;a at the Compose key to enter the ą. The table lists only lower case letters, but the corresponding upper case letters can be entered as well: Ctrl-V ´A enters the Á. The last four lines in table 1 show some other characters that can be entered using the Compose key. Thus, to enter the ¿, press Ctrl-V and type ?? followed by Enter.

Table 1. Accent designations in the Compose key.

Key	Accent	WP characters
'	acute	áéíóúýćǵĺńśź
'	grave	àèìùòỳr̀
^	circumflex	âêîˆôûŷĉĝĥŝŵ
~	tilde	ãĩõũñ
;	ogonek	ąęįų
/	slash	łø
.	overdot	ėiİċġż
:	centred dot	ŀ
,	cedilla	çķļņŗşţĢ
@	corona	åů
"	umlaut	äëïöüÿ
v	hacek	ěčďǧňřšťž
_	macron	āēīōūd̄n̄t̄
-		đ ŧ
>>		«
>>		»
!!		¡
??		¿

A few comments. As you can see, you can enter the i by typing .i in the Compose key. However, this is not the 'normal' i, but the ı with a dot (a dotted dotless i, so to speak). This is the Turkish i, and if you type Turkish you are advised to use it rather than the normal i. Firstly, to ensure that the ı and the i are sorted correctly. In Turkish, the i follows the ı, but if you type the normal i, it will be sorted before the ı. Secondly, if you enable kerning, WP automatically creates ligatures like ﬁ. It will do this for any sequence of fi, fl and, if you have expert fonts, ff, ffi and ffl. But naturally, to distinguish the fı and the fi combinations in Turkish, the fi combination should not be turned into a ligature.

Another thing is that some accents are not available in the Compose key. Notable examples are the Hungarian umlaut (˝) and the breve accent (˘) used, for example in Turkish. To type letters with these accents, you need to enter the numerical code in the Compose key, such as Ctrl-V 1,117 to enter the ğ. However, this can be remedied with a key macro, which I will discuss in the next section. (In WP 6 the u can be used to enter the breve accent in the Compose key.)

Apart from the characters mentioned above, some other characters can be entered using mnemonic keys in the Compose key. For completeness' sake these characters are listed in table 2.

Table 2. Miscellaneous characters

Keys	Result	Keys	Result
a=	ª	>=	≥
o=	º	<=	≤
Y=	¥	--	—
L-	£	**	•
f-	ƒ	*.	·
Pt	₧	*O	○
/c	¢	*o	∘
/2	½	co	©
/4	¼	ro	®
/=	≠	sm	℠
+-	±	tm	™
ox	¤	Rx	℞
P|	¶		

Overstrike.

Although the WP character sets contain almost all known accented characters, some are not included. For example, the Welsh ẅ and some Slovene accents are not in character set 1. You can however create any character yourself using the Overstrike feature. As its name suggests, the Overstrike feature prints two characters in the same position.

Let us say you want to create the ẅ. To do so, go to the Overstrike feature: press Shift-F8, 4, 5, 1 (Format, Other, Overstrike, Create). At the bottom of the screen you will now see the **[Ovrstk]** prompt and now you can enter the two characters. To create the ẅ, type "w and press Enter until you are back at the edit screen. In the edit screen you will see only the second character that you typed (in this case the w). But if you activate the Reveal Codes screen, you see the Overstrike character displayed as **[Ovrstk:"w]**.

However, you must be careful with accents that you type at the Overstrike prompt. The ẅ is an interesting example, because if you enter it using the " key, it will be printed as ẅ! So at the Overstrike prompt, you cannot use the conventional characters listed in table 1. Rather, you must use a floating accent from set 1. Now, the ¨ is character 1,7. So to create the w correctly, do as follows: go to the Overstrike prompt (Shift-F8,4,5,1). Now press Ctrl-V to activate the Compose key and type 1,7 followed by Enter. Finally, type the w and press Enter until you are back at the Edit screen. If you now look in the Reveal Codes screen, the character you just created is displayed as **[Ovrstk:■w]** . To see some more information, place the cursor on the Overstrike character and now you will see it displayed as **[Ovrstk:[■:1,7]w]** .

The order in which you enter characters at the Overstrike prompt is not important. But since you will see only the second character of an overstrike pair, it is convenient to first enter the accent and then the letter. In the print preview (Shift-F7,6) you can see how the characters will print.

The Overstrike feature is quite powerful, but it has some disadvantages. For example, words containing an Overstrike character are not sorted correctly; they are not added correctly to the supplement word list during spell checking; they are lost when you save the document as a DOS text file; and although you can search for the Overstrike code, you cannot search a particular Overstrike, nor can you do a find-and-replace in Overstrike characters. (The last point has been remedied in WP 6.)

We may conclude that WP has some convenient features to enter accented characters and special symbols. Nevertheless, if you need to enter a limited number of characters very frequently, even the Compose key becomes awkward. But WP offers another facility to conveniently enter special characters, namely the customizable keyboard. This will be taken up in the next section.

THE KEYBOARD EDITOR

Most languages use only a few accented characters very frequently. It is then not very handy to enter them using the Compose key, since flexible as it may be, it does need a handful of keystrokes. To handle this inconvenience, you can assign any character to virtually any key or key combination. In this section I will show a number of ways that can be used to reconfigure the keyboard.

Key assignments are in fact small macros that are assigned to particular keys. Indeed, the keyboard editor is identical to the macro editor. It is beyond the scope of this paper to explain the full operation of the keyboard editor; rather, I will assume knowledge of it. Most books on WP have a section on this subject; for example, my book on WP characters and languages contains all necessary background information (Kahrel (1)). Below I will make some suggestions for key assignments and discuss key macros that may make life a simpler.

The most obvious thing to do (and this is done very frequently) is to assign particular characters to particular keystrokes. This is useful if you need certain accented characters often. For example, in Dutch only three accented characters are used frequently: the é, ë and ï. It would therefore be convenient to be able to enter these characters by pressing one key, let us say Ctrl-I to enter the ï. This is easily done in the keyboard editor. (Note that in Ctrl-letter combinations, you can use only lower case letters. Thus, it is not possible to define Ctrl-i and Ctrl-I as two distinct keystrokes.)

I mentioned that you can assign a macro of any complexity to a key for special purposes. Let me give a few examples. I will begin with some relatively simple examples, and finish with a rather more complex one.

If you type words separated by a slash (such as *man/woman),* it would be convenient to insert a so-called invisible hyphen after the slash, so that this 'word' is hyphenated correctly after the hyphen. The easiest way to accomplish this is to define the / key such that when you press it, the invisible hyphen is inserted automatically. To do so, assign the following macro to the slash key:

```
/{Home}{Enter}
```

With this key assignment, you don't have to think about inserting the invisible hyphen anymore.

The next example is useful if you type Portuguese or Polish. These two languages share the rule that if a word that contains a hyphen is hyphenated at the end of a line, the hyphen is doubled. For example, ***Polski-Fiat*** looks like this at the end of a line: ***Polski-***
-Fiat. In WP, you can make the hyphens behave correctly for Polish and Portuguese if you enter them as the combination of the soft hyphen and the hard hyphen. To have these two distinct hyphens inserted by pressing just the - key, assign the following macro to the - key in the keyboard editor:

```
{Shy}{Home}-
```

{SHy}- stands for soft hyphen, which is the hyphen inserted by the hyphenation module; **{Home}-** are the keystrokes required to enter the hard hyphen, which is the hyphen that is always visible.

With the next example I come back to my promise to show how omissions in the Compose characters can be remedied. I mentioned that, contrary to what you would expect, the u does not enter the breve accent in the Compose key (in WP 6 it does). But it is not difficult to create your own Compose character. The following macro takes care of that:

```
u
{IF}{SYSTEM}13¯=32790¯
{ELSE}
  {RETURN}
{END IF}
{CHAR}ch¯¯
{Enter}
{CASE}{VARIABLE}ch˜˜
  A¯u1¯a¯u2¯G¯u3¯g¯u4¯U¯u5¯u¯u6¯Y¯u7¯y¯u8¯
  {RETURN}

{LABEL}u1¯{NTOK}1,98¯{RETURN}
{LABEL}u2¯{NTOK}1,91¯{RETURN}
{LABEL}u3¯{NTOK}1,116¯{RETURN}
{LABEL}u4¯{NTOK}1,117¯{RETURN}
{LABEL}u5¯{NTOK}1,188¯{RETURN}
{LABEL}u6¯{NTOK}1,189¯{RETURN}
{LABEL}u7¯{NTOK}1,224¯{RETURN}
{LABEL}u8¯{NTOK}1,225¯{RETURN}
```

With this macro assigned to the u key, the u behaves as the breve accent in the Compose key. Thus, you can type ug in the Compose key to enter the ğ.

To conclude this section, and to link smart keyboards to the language code, I will give an example of a way to handle multilingual documents. Suppose that you use two languages: English and Russian. What you need is a keyboard that enables you to type English and Russian (in the Cyrillic alphabet) and a key that inserts the correct language code. We'll start with the key that inserts the language code. Take the following macro:

```
{DISPLAY OFF}
{IF}"{SYSTEM}32~"="UK"~
  {Format}44RU{Enter>{Exit>
{ELSE}
  {Format}44UK{Enter}{Exit}
{END IF}
```

What this macro does is the following. When activated, it checks which language code is active (system variable 32 holds the current language code). If English is active, the Russian language code is inserted (RU), and if it is not, the English code is inserted. It is convenient to assign this macro to a key that has no meaning in WP, such as the Alt-Enter combination. Now for the keys. It is possible to create a keyboard in which each letter produces either a Latin or a Cyrillic character. We can do this by making each key sensitive to the language code. So, if the English language code is active, the d key should produce the d, and if the Russian language code is active, the д. Basically, this is a variant of the previous macro. Take the following macro:

```
{IF}"{SYSTEM}32~"="UK"~
  d
{ELSE}
  д
{END IF}
```

Assign this macro to the d key in the keyboard driver. Thus defined, the d key behaves as follows: when pressed, first the current language code is determined, if it is English, the d is inserted into the document, otherwise the д. Note that this key macro does not check for Russian. It assumes that if the UK code is not active you want Russian. This macro can therefore be used for other languages as well; just change the д to another character.

Although the macro discussed here works fine, it has one shortcoming. If you are accustomed to using the mnemonic letters rather than the numbers while cruising the WP menus, you cannot use these mnemonics if the Russian language code is active. For example, you can go to the line margin menu by pressing Shift-F8,l,m. But if Russian is active, you would have to use Shift-F8,l,7, since the l and the m then produce Cyrillic, which WP does not understand. Further, if Russian is active, you cannot type a file name when saving or retrieving a document, answer *y* or *n* to a WP question, and so on. So apart from making the keys sensitive to the language code, we also want to make them sensitive to whether or not we are in the edit screen. The general format of such keys is as follows:

```
{IF}{STATE}&4~
  (do something)
{ELSE}
  (alternative)
{END IF}
```

This general format is WP macro language for 'if at the edit screen, do something, else do something else'. Edit screen here also includes headers, footers, endnotes and footnotes.

The thing to do now is to embed the macro for the d key in this general format. The macro to be assigned to the d key will then look as follows:

```
{IF}{STATE}&4~
  {IF}"{SYSTEM}32¯"="UK"¯
    d
  {ELSE}
    д
  {END IF}
{ELSE}
  d
{END IF}
```

To complete the keyboard, you should assign similar macros for each key. Fortunately, you can copy a macro from one key to another, which is convenient in our case. Do as follows: create the macro for the d key as described below, then go back to the edit screen to activate the keyboard layout with just this one d in it. Then go to the keyboard editor again. To copy the macro from the d to the i, type 1 (Create) in the keyboard editor and press Enter to enter the macro window. The i is in this window; delete it. Now press Ctrl-V and type d to copy the macro assigned to the d in the current window. Just replace the d with the i and the д with the и and press F7 to save the changes. In this way it is not difficult to create a well-working bilingual keyboard.

WORDPERFECT 6

Recently, WordPerfect released WP version 6. In this new version various linguistic characteristics are more refined. On the whole, I think that for anyone using foreign languages, WP 6.0 is a great improvement. A rather drastic change is the ability to edit in a graphic screen, in which every character is displayed correctly: screen font editors are a thing of the past. Secondly, WP includes a large number of printer fonts in the package that enable you to print all characters. Fonts are included in Type 1 and Speedo format, and WP also supports TrueType and Agfa Intellifont. With the included font installer fonts of these formats can be installed in the WP printer driver. Another big change is the macro language, which is completely new.

Character sets. Most character sets have changed in some way. Linguistically, the following changes have been made.

Character set 1 has been modified to correct some errors and to add some characters. WP 5.1 documents are updated automatically when you retrieve them in version 6. For example, the dotless i (ı) was 1,24 in WP 5.1, but is 1,239 in WP 6; when you retrieve a 5.1 document in 6, the 1,24 code is changed to 1,239 automatically.

Character set 2 is new and linguists will love it: it contains 144 phonetic characters.

Character set 8 (Greek): some minor changes.

Character set 9 (Hebrew): this character set has been reorganized considerably.

Character set 10 (Cyrillic) is now called Cyrillic/Georgian. The Cyrillic has been slightly modified. It now includes Georgian as well.

Character set 11 (Japanese). Drastically changed. In WP 5.1, this set contained the full Hiragana and Katakana sets; in WP 6, by contrast, it contains only 62 Katakana characters.

Character sets 13 and 14 are new: they contain Arabic and script Arabic characters.

Compose key. Although mapped to another key (it is Ctrl-A, in WP 5.1 it was Ctrl-V), the Compose key works the same. The breve accent has been added as an accent that can be typed in the Compose key. You can use the letter u as a mnemonic.

Character window. This is new in WP 6, but you may know it from WP 5.2 for Windows: press Ctrl-W to get an on-screen overview of all character sets. You can use this window to insert characters in the document.

Overstrike. Using WP's Search function, you can now search a particular Overstrike character. You can also do a find-and-replace in Overstrike: search one Overstrike character and replace it by another. In the text mode, only the last character is displayed, as in WP 5.1; in the graphic mode, all characters are displayed.

Word lists. The supplement word list, which is created and updated when you add words to it during spell checking, is no longer a standard WP document. You must now use a special menu to edit it. An interesting feature is that you can define an automatic replacement in the word list. Suppose you want to change English to American spelling. You can include in the supplement list statements to the effect that ***center*** should be changed to ***centre, harbor*** to ***harbour,*** etcetera. Once these replacements are defined in the word list, they are automatically implemented during spell checking.

Spell checker. The spell checker itself is essentially the same as the one in 5.1. But it is now possible to include codes to exclude parts of a document from spell checking.

Grammar checking. WP 6 includes the Grammatik grammar checker, which is also included WP and Word for Windows.

Macros. Although WP contains the basics of a good linguistic word processor, it basically lives on the macro language to drive the keyboard. This was true in WP 5.1, and is still true in WP 6.0. It is therefore a relief that the macro language in WP 6.0 is much more powerful than the one in WP 5.1. The new macro language is basically Turbo Pascal with a bit of C notation. Anyone who knows Pascal can write WP macros without any effort; you only need to get used to a few notational variants. For example, converting a Quicksort routine and a binary search function from Turbo Pascal to WP was a matter of minutes!

WordPerfect includes a program to convert 5.1 macros to 6.0 format. Contrary to the 4.2 to 5.0/5.1 converter, this program works very well. Even complex macros were converted successfully.

References.

1. Kahrel, P., 1992, "Working with foreign languages and characters in WordPerfect", John Benjamins, Amsterdam.