Typing diacritics in JuniusX

JuniusX has better support for diacritics than any other free medieval font. You can place a diacritic (or "combining mark") on just about any character where it makes any sense at all to place it. Just type the base character and the diacritic (you'll need to know the hexadecimal Unicode number, e.g. "1DD1," and how to enter it). But there is a significant complication that has to do with the distinction between Unicode diacritics and those defined by the Medieval Unicode Font Initiative (MUFI).

Software tends to "know" and correctly handle diacritics that are part of the Unicode standard, but the same is not true of diacritics defined by MUFI and assigned codepoints in a font's Private Use Area. Fortunately, Junius X provides a method that will work with most software, especially browsers and LibreOffice. (But if you're working in Microsoft Word, which has the worst font support of any major text-editing or processing software, there is nothing you can do. Sorry.)

First the good news. Unicode includes the diacritics most commonly used by medievalists: chances are you'll never need any others, so all you need is a convenient reference, which the following table will provide. Here are the Unicode diacritics identified by MUFI as being of interest to medievalists (the first three digits of the Unicode number are along the left side, the fourth across the top):

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
030	ò	ó	ô	õ	ō	o̅	ŏ	ȯ	ö	o̡	o̊	ő		o̍	o̎	
031						o᾽										
032				ọ				o̓	o̔							
033			o̲	o̳		ɵ	ɵ								o̽	o̿
035										o͕		o͝	o͡o			
036				o͔	o͕	o͖	o͗	o͘	o͙	o͚	o͛	o͜	o͝	o͞	o͟	o͠
1DC														o͡o	o͢	o͢
1DD	o͢	oͣ	oͤ	oͥ	oͦ	oͧ	oͨ	oͩ	oͪ	oͫ	oͬ	oͭ	oͮ	oͯ	oͯ	oͯ
1DE	o͞	o͞	o͞	o͞	o͞	o͞	o͞									

That's a lot of diacritics! And as you can see, they are (mostly) correctly positioned over their base characters. Any one of them would be correctly positioned over a wider character, e.g. m̐, or over a taller one, e.g. Ǎ t̐.

But MUFI contains many more diacritics, and some software will not position them correctly. For example, a combining ar ligature (U+F038) looks like this over m: m᷎. It looks like this over an A: A᷎. That's because the layout engine (Harfbuzz) employed by the application in which I'm writing this document (LibreOffice) does not recognize or treat U+F038 as a combining mark.

But I can fool LibreOffice by typing an "a" instead of U+F038 and applying a "Character Variant" tag "cv64=3" to convert that "a" into the combining "ar" ligature: m᷎. (This won't work in MS Word, which does not support Character Variants.) You apply a Character Variant by adding a suffix to the font name in the font box at the top of the screen: change "JuniusX" to "JuniusX:cv64=3" ("JuniusX:cv64" would be the same as "JuniusX:cv64=1").

In browsers, you should define CSS classes to do this work. For example, assuming you have correctly set up the @font-face definitions, e.g.

```
@font-face {
  font-family: JuniusX;
  src: url('../fonts/JuniusX-Regular.woff2');
  font-weight: 400;
  font-stretch: 100%;
  font-style: normal;
}
```

next you define a class to handle, e.g. the combining form of ē:

```
.ecomb {
  font-feature-settings: “cv70” on;
}
```

Finally, to produce the correct base+mark combination, type

```
<span class=”ecomb”>mē</span>.
```

The result will be to convert the ē into a combining mark that is positioned correctly: m̄. Note that the `<span>` must include both the m and the ē, that is, both the base character and the mark. This won't work: `m<span class=”ecomb”>ē</span>`. Similarly, in LibreOffice you must select both the base character and the mark before applying the Character Variant.

Because most of these Character Variants convert ordinary letters into combining marks, you need to apply them only to the two characters, or perhaps the word. Otherwise, the result may be shocking, e.g. “The quick brwn fx jumps ver the lazy dg.” Here “cv72” has converted every o into an exotic combining mark.

A few Character Variants (cv61–63) produce different forms of a semantically identical combining mark, e.g. o̓ instead of o̔. You can and should apply these to the whole text once you’ve decided which form you want to use.

Finally you need a reference for the Character Variations and what they do. The following table shows all of the ones that create or vary combining marks:

code		source		1	2	3	4
cv61	turns	o̔	into	o̓	o̓		
cv62		ő		o̿			
cv63		o̕		o̽			
cv64		a		oᵃ	oᴺ	oᵃ	oᴿ
cv65		b		oᵇ	oᴮ		
cv66		d		oᴰ			
cv67		f		oᶠ			
cv68		ı		oᴵ			
cv69		j		oᴶ			
		ȷ		oᴶ			
cv70		ę		oᶒ			
		ē		oᵉ̄			
cv71		k		oᴷ			
cv72		o		o[1]	oᵒ		
		ō		oᵒ̄			
		ǫ		oᵫ			

1 A combining mark produced by an o can’t go over an o with this system for obvious reasons. For that and similar cases you must use the original MUFI code point: see https://skaldic.abdn.ac.uk/m.php?p=mufi.

		ø		ø a			
cv73		p		p o			
cv74		q		q o			
cv75		t		t o			
		ʈ		ʈ o			
cv76		w		w o			
cv77		y		y o			
cv78		þ		þ o			