Using Junicode 2 to reproduce the abbreviations in Martin, *The Record Interpreter*, *Statutes of the Realm*, and similar texts

A preliminary note on transcription

Here are a few observations, based on a long career as a scholarly editor of medieval and eighteenth-century texts.

Before embarking on the task of transcribing an old document, ask yourself what value your work is adding, because different kinds of transcription add different kinds of value. The kind of transcription that adds the least is that which aims at the exact *visual* reproduction of a document. A transcript is not a facsimile: it needs to do more than a photograph can do.

Converting a document from visual image to Unicode-encoded text adds a good bit of value all by itself, but only if done with due regard for the semantics of Unicode characters. Every Unicode character has a meaning, and that meaning is a help to readers. Using the wrong character is a hinderance to readers, even it if *looks* right.

For example, in transcribing a Middle English text, you may decide that the Unicode EZH (ʒ, U+0292) looks more like the yogh in your source than the Unicode YOGH (ȝ, U+021D) and therefore decide to use it for YOGH. But the ezh is not a yogh! It is a character in the International Phonetic Alphabet and a letter in the alphabets of several minor languages. If you use it where the yogh is called for, it will make your text less accessible and less searchable. Indexing, concordance and bibliographical programs will be misled by it; screen readers will misinterpret it. To solve one problem (that of exact visual representation), you may well have introduced a host of far more serious problems.

Fortunately, Junicode offers a solution for this particular problem. The OpenType feature **cv63** substitutes for the yogh a character that *looks* like the ezh but is semantically a yogh and therefore will be handled correctly by computer applications. But neither Junicode nor any other program can solve every problem of this kind. Sometimes you will have to call to mind the important principle stated above: *A transcript is not a facsimile*. It is much more important that it should have the same *meaning* as the original than that it should have the same *look*.

This document concerns the transcription of texts in Latin (and to some extent, other archaic languages, e.g. Old and Middle English, Old French). It is long-standing custom, when transcribing certain kinds of documents, to retains marks of abbreviation—for example, transcribing the **ꝓpͣ** you find in a manuscript or printed edition for the word (*propterea*) it represents. This is okay—and Junicode can help with the task! But when dealing with the abbreviations, punctuation, and diacritics of an old text it is more important than ever that you use semantically correct characters for your transcription, as this will help your readers deal with an already challenging text.

For example, the abbreviation **ꝓpͣ**, cited above, consists of an underlying sequence of Unicode characters, which, taken together, constitute one correct choice for rendering *propterea*: **ꝓ** (U+A753, the common abbreviation for *pro*) + **p** + ◌ͣ (U+0363, the combining small *a*). The OpenType feature **hlig** (Historical Ligatures) has been applied to this sequence, changing its appearance but not its underlying value. That underlying value will be meaningful to computer applications.

This doesn’t mean, though, that computer programs can correctly interpret **ꝓpͣ** as *propterea*. Many (probably *most*) Latin abbreviations are ambiguous: this one, for example, can mean *propterea* or *propria*. Some abbreviations (notoriously ◌͛ U+035B) can mean many things, depending on context.

So another way you can add value in your transcript is by supplying expansions of your abbreviations. Fortunately, systems for representing texts often offer ways to handle this task gracefully. For example, in a TEI (Text Encoding Initiative) text, you would use this construction:

```
<choice>
  <abbr rend=”hlig”>ꝓpͣ</abbr>
  <expan>propterea</expan>
</choice>
```

This kind of structure can be approximated in HTML, with supporting CSS and scripting to allow readers to choose between a “diplomatic” version, with unexpanded abbreviations, and a “reading” version, with expanded abbreviations and perhaps other amenities, such as modern punctuation and capitalization.

A transcript that shows readers both the raw original and its interpretation adds a great deal of value.

1. Common combining marks

A **combining mark** is a character that combines with another character (called the **base**) to form a character with accent (e.g. **é**) or an abbreviation (e.g. **p͛** for *prae*). Unicode and the Medieval Unicode Font Initiative (MUFI) offer code points for many precomposed combinations of base + combining mark, but it is also possible to place any mark over any base character by entering first the base and then the combining mark. It is also possible to place a combining mark over another combining mark. For example, to produce **qͣ̄**, enter this sequence: q (U+0071) + U+0363 + U+0304.

Junicode 2 contains many variants of combining marks: for example the curly zigzag ◌͛ is a variant of Unicode's angular zigzag ◌͛ (U+035B), produced by applying the OpenType feature **cv81[2]** to **both the base character and the combining mark**. Sometimes the combination of base + combining mark + OpenType feature will not produce the desired effect. When this happens, place U+034F (a special invisible combining mark, included in Unicode for exactly this purpose) between the base and the (visible) mark.

a. For a straight stroke over any letter, use the COMBINING MACRON (U+0304):

   ōnis *omnis*; om̄is *omnis*; dāpna *dampna*; damp̄a *dampna*.

   The combining macron can also be applied above superscripts and combining marks. Apply the OpenType feature **cv84[39]** for a narrower macron:

   antiqu$^{\bar{a}}$ *antiquam*; qͣ̄ *quam*.

   For the superscript *a*, use the OpenType feature **sups** (see r. below).

b. For a straight stroke through a tall letter, use the COMBINING SHORT STROKE OVERLAY (U+0335): **f̵ đ ƚ**. But Unicode also has precomposed versions of **d**, **l** and other characters with stroke, e.g. **đ** (U+0111), **ƚ** (U+019A).

c. For **~** above any character, use the COMBINING TILDE (U+0303):

   ã *ac*, *apud*; ã *alias*.
   dñs *dominus*; carĩna *carmina*; fc̃is *factis*.
   põita *posita*.

d. For **~** through a vertical stroke, use the TILDE OVERLAY (U+0334): **l̴ d̴** (U+0303 would be positioned above the letter, e.g. **l̃**, **d̃**).

e. For the tilde positioned above two letters, use COMBINING DOUBLE TILDE (U+0360) between the letters. It is automatically repositioned to clear tall characters: c͠o t͠o d͠o o͠l. The same is true of DOUBLE BREVE (U+035D) c͝o d͝o, DOUBLE MACRON (U+035E) c͞o d͞o, DOUBLE INVERTED BREVE (U+0361) c͡o d͡o, and DOUBLE CIRCUMFLEX (U+1DCD) c᷍o d᷍o.

f. The figure used to represent *er* (and other similar combinations) is a common medieval abbreviation which takes many forms. The semantically correct Unicode character is the COMBINING ZIGZAG (◌͛, U+035B), but the best match in Junicode 2 for the figure as it appears in the *Record Interpreter* and the *Statutes* is a gothic variant of this, which MUFI encodes as U+F1C8 (the curly form zigzag). However, because for technical reasons many applications will not position the MUFI character correctly over the base, that code point should be avoided. The best way to access this variant is to apply **cv81[2]** to U+035B, as here:

> de͛be debere; int͛ *inter*; f͛rū *ferrum*; gn͛o generatio; p͛; *prae*; ser͛ue *servire*.

The curly form of the combining zigzag may be attached to any letter, and it may change shape depending on the letter it is attached to (including caps, for which use the **case** feature, and small caps: A͛ B͛ ᴄ͛ ᴅ͛).

g. All letters a–z, and several others too, have combining forms. You must access these via their code points or Junicode's special entity references. For details, see the document Diacritics_guide.pdf.

> qͦ *quo*; qͥ *qui*; quattͦͬ *quattuor*.

2. Spacing characters

h. The symbol for *is*, *es* and a number of other abbreviations is the IS-SIGN (U+A76D):

> forꝭ *foris*; om̄ꝭ *omnes*; ꝯtꝭ *competentes*; infꝭ *infortunium*.

i. There are two characters for *-us* in Unicode: SPACING US U+A770 (do not confuse this with CON U+A76F) and COMBINING US U+1DD2. The *Record Interpreter* and *Statutes* appear to use only the spacing character:

> ip̃iꝰ *ipsius*; u͛sꝰ *uersus*; pꝰtea *postea*; pꝰ *post*.

j. The three-like sign is the ET SIGN (◌ꝫ, U+A76B, also used for *us*—do not confuse this with Middle English yogh: **ȝ**, U+021D):

quibꝫ *quibus*; licꝫ *licet*; sꝫ *sed*.

k. For *-rum* the Unicode RUM ROTUNDA (U+A75D) is like the one in MUFI/Junicode. The one in the *Record Interpreter* and *Statutes* is a late stylized version of this. Use U+A75D and apply OpenType feature **cv80** to obtain the correct shape:

aĩaꝝ *animarum*; coꝝpere *corrumpere*; beatoꝝ *beatorum*.

l. For *cum*, *con*, etc. use SMALL LETTER CON (U+A76F):

ꝯputus *computus*; ꝯa *contra*; ꝯnouit *cognouit*.

m. For *per* (or sometimes *par* and other similar sequences), use P WITH STROKE U+A751:

ꝑsōa *persona*; ꝯꝑet *comparet*.

n. For *pro,* use P WITH FLOURISH U+A753:

ꝓceres *proceres*.

o. For *prae*, *præ*, *pre*, there is no separate character; use a variant of the ZIGZAG (f. above) with **p**:

p͛sẽs *praesens*.

p. For **q** with stroke through the descender, there are two Unicode points: U+A757 for a straight stroke, and U+A759 for a diagonal stroke (the *Record Interpreter* appears to use only the former, and neither is listed among the *Statutes* abbreviations):

ꝗ *quod*; ꝗd *quid*; ꝗbꝫ *quibus*.

q. For *quae*, *que*, use **q** followed by ET (U+A76B) with or without **hlig**: **qꝫ qꝫ**. For the semi-colon-like ET sign (**q;**), use **cv83[1]**; for the subscripted version (which can also form a ligature via **hlig**), use **cv83[2]**: **qꝫ qꝫ**.

r. All of the letters a-z are available in superscript form. Access with the **sups** OpenType feature:

q^{o}s *quos*; c^{i}lo *circulo*; cap^{i} *capituli*.

The basic Latin letters a–z have anchors that allow you to position combining marks over them (see a. above)

s. Tironian ET sign ⁊ U+204A, cap ⹒ U+2E52. With **cv69[1]** ꝣꝢ; with **cv69[2]** ƶƵ.

t. For *est*, use ∻ U+223B HOMOTHETIC. Use of a mathematical sign for this purpose is not ideal, but Unicode offers no better solution.

u. For *tz* (Old French), use ƶ U+01B6 Z WITH STROKE.

v. For an abbreviation for *Rex*, use ℞ U+211E or ℟ U+211F.

3. Other formatting

v. For underdotted text, use Stylistic Set 7, Underdotted. For letters that lack an underdotted form, use U+0323 COMBINING DOT BELOW.

4. Junicode on the web

Because Junicode is a very large font, web pages should use a subsetted version to speed loading. The process of making a subsetted font is explained in the Feature Reference. The variable version of the font is better for web use than the static fonts, since one variable font file can do the work of many static font files.

All major web browsers (Firefox, Chrome, Safari, Edge) are capable of accessing all of Junicode's characters via OpenType features, use of which promotes accessibility and searchability. When building a web page, study which features will be needed and write them into the appropriate element or class definition of the page's CSS style sheet. For example, if you use the curly form of the zigzag (U+035B) anywhere, you are likely to want it everywhere, and so it should be included in the CSS styling for the <body> element:

```
body {
  font-family: Junicode;
  font-feature-settings: "cv81" 2;
}
```

But the **hlig** feature, if applied to the whole text, will produce many unwanted effects, so it should be included in a class definition to be used in a <span> applied just to the target sequence:

```
.que {
  font-feature-settings: "hlig" on;
}
filio<span class="que">qꝫ</span>
```

The illustrations here use the low-level CSS font-feature-settings property. There are higher-level properties for some OpenType features, but as these are not (yet) universally supported by browsers, and some implementations are buggy, it is best to stick with font-feature-settings for now.

For the purposes addressed in this document, the font-feature-settings for the <body> element should probably be as follows:

```
font-feature-settings: 'cv69' 2, 'cv80' 1, 'cv81' 2;
```

And the following classes should be defined:

```
.super {
 font-feature-settings: 'sups' on, 'cv84' 39;
}
.que {
  font-feature-settings: 'hlig' on;
}
.deleted {
  font-feature-settings: 'ss07' on;
}
```