



Junicode

the font for medievalists



*specimens and feature reference*

for version 2





# Contents

[About Junicode](#)

[Specimens](#)

[Feature Reference](#)

[Introduction](#)

- [A Case-Related Features](#)
- [B Numbers and Sequencing](#)
- [C Superscripts and Subscripts](#)
- [D Ornaments](#)
- [E Alphabetic Variants](#)
- [F Greek](#)
- [G Punctuation](#)
- [H Abbreviations](#)
- [I Combining Marks](#)
- [J Currency and Weights](#)
- [K Gothic](#)
- [L Runic](#)
- [M Ligatures and Digraphs](#)
- [N Required Features](#)
- [O Non-MUFI Code Points](#)



## About Junicode

The design of Junicode is based on scans of George Hickes, *Linguarum vett. septentrionalium thesaurus grammatico-criticus et archaeologicus* (Oxford, 1703–5). This massive two-volume folio is a fine example of the work of the Oxford University Press at this period: printed in multiple types (for every language had to have its proper type) and lavishly illustrated with engravings of manuscript pages, coins and artifacts.

The type used for Hickes’s *Thesaurus* resembles those assembled by John Fell (1625–86) and bequeathed by him to the University of Oxford. To my eye, however, it looks more like the “Pica Roman” purchased by the University in 1692 than like any of Fell’s. For printing in Old English, this type was supplemented by the “Pica Saxon” commissioned by the early Anglo-Saxonist Franciscus Junius (1591–1677) and bequeathed by him to the University. Specimens of both can be found in *A Specimen of the Several Sorts of Letter Given to the University by Dr. John Fell, Sometime Lord Bishop of Oxford. To Which Is Added the Letter Given by Mr. F. Junius* (Oxford, 1693). Junius’s Pica Saxon was mixed freely with Pica Roman in printing the *Thesaurus*.

Junicode includes two Greek faces, one for the roman font and another for the italic. The first is newly designed to harmonize with Junicode, and the other, accompanying the italic face, is based on type designed by Alexander Wilson (1714–86) of Glasgow and used in numerous books published by the Foulis Press, most notably the great Glasgow Homer.

#### 4 JUNICODE

The font dates from 1998, when the developer began to revise his older (early 1990s) “Junius” fonts for medievalists to take account of the Unicode system standard, then relatively new. The font’s name, a contraction of “Junius Unicode,” was supposed to be a stopgap, serving until a more suitable name could be found, but the name “Junicode” is now so well known that it can’t be changed. The project has been active for its entire history, responding to frequent requests from users and changes in font technology; the developer is now pushing towards the release of Junicode version 2, an extended font family of five weights and five widths, with both static and variable versions, which aims to promote best practices in the presentation of medieval texts, especially in the area of accessibility. This aspect of the font is explored in the Introduction to the Feature Reference.

*Users need to be aware that Junicode version 2 is a beta, unstable and incomplete. Until the official release of version 2 (probably sometime in 2022), versions of Junicode numbered 1.NNN should be used for projects that require stability. The latest stable version can be downloaded [here](#).*

# Specimens

## Old and Middle English

Wē æthrynon mid ūrum ārum þā yðan þæs dēopan wāles; wē gesāwon ēac þā muntas ymbe þære sealtan sē strande, and wē mid āðēnedum hrægle and gesundfullum windum þær gewīcodon on þām gemārum þære fægerestan þēode. Þā yðan getācniað þisne dēopan cræft, and þā muntas getācniað ēac þā micelnyssa þisses cræftes. (Regular)

Siþen þe sege and þe assault watz sesed at Troye,  
þe borȝ brittened and brent to brondez and askez,  
þe tulk þat þe trammes of tresoun þer wroȝt  
Watz tried for his tricherie, þe trewest on erthe:  
Hit watz Ennias þe athel, and his highe kynde,  
þat siþen depreced prouinces, and patrounes bicomē  
Welneȝe of al þe wele in þe west iles. (SemiExpanded)

*Apply the OpenType feature ss02 (Stylistic Set 2) for insular letter-forms.*

Her cýnepulŕ benam riȝebŕýht hiŕ piŕeŕ ȝ peŕŕeaxna piotan ƿop unŕýhtum ðeðū buton hamtúnŕŕiŕe  
ȝ he hæŕðe þa oþ he oŕŕloȝ þone aldoŕmon þe hī lenȝeŕt punode ȝ hiene þa cýnepulŕ on andŕeð aðŕæŕðe  
ȝ h þæŕ punaðe oþ þæt hine án ŕŕán oŕŕtanȝ æt ŕŕýŕeteŕŕlodan ȝ he ŕŕeŕ þone aldoŕmon cumbŕan ȝ  
ŕe cýnepulŕ oŕt miclum ȝeŕeohtum ŕeaht uuþ bŕetpalū. (SemiCondensed)

## Old Icelandic

*For Nordic shapes of þ and ð in an English context, specify the appropriate language (e.g. Icelandic or Norwegian); or apply the OpenType ss01 (Stylistic Set 1) feature.*

Um haustit sendi Mǫrðr Valgarðsson orð at Gunnarr myndi vera einn heimi, en lið alt myndi vera niðri í eyjum at lúka heyverkum. Riðu þeir Gizurr Hvíti ok Geirr Goði austr

## 6 JUNICODE

yfir ár, þegar þeir spurðu þat, ok austr yfir sanda til Hofs. Þá sendu þeir orð Starkaði undir Þríhyrningi; ok fundusk þeir þar allir er at Gunnari skyldu fara, ok réðu hversu at skyldi fara. (SemiExpanded Medium)

## Runic

[illegible]

## German

Ich sag üch aber / minen fründen / Fōzchtēd üch nit vo2 denen die den lyb tōdend / vnd darnach nichts habennd  
das fy mer thūgind. Ich wil üch aber zeigē vo2 welchem ir üch fōzchten sollend. Fōzchtend üch vo2 dem / der  
/ nach dem er tōdet hat / ouch macht hat zewerffen inn die hell: ja ich sag üch / vo2 dem selben fōzchtēd üch.  
Koufft man nit fünff Sparen vñ zween pfennig? (Condensed)

## Latin

*Junicode contains the most common Latin abbreviations, making it suitable for diplomatic editions of Latin texts.*

Adiuuanos dñ salutaris noster & pp̄t gl̄am nominis tui dn̄e libanos & pp̄tius esto peccatis nostris  
pp̄ter nomen tuum. Ne forte dicant ingentib: ubi est dñ eorum & innotescat innationib: corā  
oculis nr̄is. Posuerunt morticina feruorū ruorū esca uolatilib: celi carnes scōꝝ tuoꝝ bestiis tenice.  
Facti sum⁹ obp̄brium uicinis nr̄is. (Light)

# Gothic

jabai auk has gasaihvip huk pana habandan kunpi in galiuge stada anakumbjandan, niu mipwissei is siukis wisandins timrjada du galiugagudam gasalip matjan? fraqistniip auk sa unmahteiga ana peinamma witubnja bropar in pize Xristus gaswalt. swap pan frawaurkjandans wipra bropruns, slahandans ize gahugd siuka, du Xristau frawaurkeip. (SemiCondensed Light)

*Use ss19 to produce Gothic letters automatically from transliterated text.*

ԳԼԵՒԻ ԴՈՒՍ ԴՆՏԻՓԻ ՓՈՒ ՓԼՈՒ ԿԵՆԴԻՆ ԿՈՆՓԻ ԻՆ ԴԼԻՈՒԴԵ ՏԴԻԴԻ ԴՈՒԿՈՄ-  
ԵԴՆԻԴՆ, ՈՒՆ ՄԻՓՎԻՏՏԵԻ ԻՏ ՏԻՈՒՏ ՎԻՏԻՆԻՏ ԴԻՄԿԵԴԻԴ ԴՈ ԴԼԻՈՒԴԻՆԴՈՒՄ



ΓΛΣΛΛΙΨ ΜΛΤΓΛΝ? ΓΛΒΛΙ ΛΗΚ ΘΛΣ ΓΛΣΛΙΘΙΨ ΨΗΚ ΨΛΝΛ ΗΛΒΛΝΔΛΝ ΚΗΝΨΙ ΙΝ ΓΛ-  
ΛΙΗΓΕ ΣΤΛΔΛ ΛΗΛΚΗΜΒΓΛΝΔΛΝ, ΝΙΗ ΜΙΨΥΙΣΣΕΙ ΙΣ ΣΙΗΚΙΣ ΥΙΣΛΝΔΙΝΣ ΤΙΜΚΓΛΔΛ  
ΔΗ ΓΛΛΙΗΓΛΓΗΔΛΜ ΓΛΣΛΛΙΨ ΜΛΤΓΛΝ? (SemiExpanded Bold)

## Sanskrit Transliteration

mānaṁ dvividhaṁ viṣayadvai vidyātsaktyasaktitah  
arthakriyāyāṁ keśadirnārtho 'narthādhimokṣataḥ

sadrśāsadrśatvācca viṣayāviṣayatvataḥ  
śabdasyānyanimittānāṁ bhāve dhīśadasattvataḥ (SemiCondensed Semibold)

## International Phonetic Alphabet

hwan θat a:pril wiθ is ju:rəs so:tə θə druxt ɔf martʃ haθ pe:rsəd to: θə ro:te and ba:ðəd enri  
væin in swiʃf liku:r ɔf hwiʃf vertiu endʒendrəd is θə flu:r hwan zefirus e:k wiθ his swe:tə bræ:θ  
(Regular)

## Greek

βίβλος γενέσεως ἰησοῦ χριστοῦ υἱοῦ δαυιδ υἱοῦ ἀβραάμ. ἀβραάμ ἐγέννησεν τὸν ἰσαάκ, ἰσαάκ  
δὲ ἐγέννησεν τὸν ἱακώβ, ἱακώβ δὲ ἐγέννησεν τὸν ἰούδαν καὶ τοὺς ἀδελφοὺς αὐτοῦ, ἰούδας δὲ  
ἐγέννησεν τὸν φάρες καὶ τὸν ζάρα ἐκ τῆς θαμάρ, φάρες δὲ ἐγέννησεν τὸν ἑσρώμ, ἑσρώμ δὲ  
ἐγέννησεν τὸν ἄράμ, ἄράμ δὲ ἐγέννησεν τὸν ἀμιναδάβ, ἀμιναδάβ δὲ ἐγέννησεν τὸν ναασσών,  
ναασσών δὲ ἐγέννησεν τὸν σαλμών, σαλμών δὲ ἐγέννησεν τὸν βόες ἐκ τῆς ραχά (Regular)

βίβλος γενέσεως ἰησοῦ χριστοῦ υἱοῦ δαυιδ υἱοῦ ἀβραάμ. ἀβραάμ ἐγέννησεν τὸν ἰσαάκ,  
ἰσαάκ δὲ ἐγέννησεν τὸν ἱακώβ, ἱακώβ δὲ ἐγέννησεν τὸν ἰούδαν καὶ τοὺς ἀδελφοὺς αὐτοῦ,  
ἰούδας δὲ ἐγέννησεν τὸν φάρες καὶ τὸν ζάρα ἐκ τῆς θαμάρ, φάρες δὲ ἐγέννησεν τὸν ἑσρώμ,  
ἑσρώμ δὲ ἐγέννησεν τὸν ἄράμ, ἄράμ δὲ ἐγέννησεν τὸν ἀμιναδάβ, ἀμιναδάβ δὲ ἐγέννησεν  
τὸν ναασσών, ναασσών δὲ ἐγέννησεν τὸν σαλμών, σαλμών δὲ ἐγέννησεν τὸν βόες ἐκ τῆς ραχά  
(Italic)

## Lithuanian

*Lithuanian poses several typographical challenges. Make sure Contextual Alternates (calt) is turned on; for ĳ, use ĳ followed by combining dot accent (U+0307) and acute (U+0301).*

Visa žemė turėjo vieną kalbą ir tuos pačius žodžius. Kai žmonės kėlėsi iš rytų, jie

## 8 JUNICODE

rado slėnį Šinaro krašte ir ten įsikūrė. Vieni kitiems sakė: Eime, pasidirbkime plytų ir jas išdekime. – Vietoj akmens jie naudojo plytas, o vietoj kalkių – bitumą. Eime, – jie sakė, – pasistatykime miestą ir bokštą su dangų siekiančia viršūne ir pasidarykime sau vardą, kad nebūtume išblaškyti po visą žemės veidą. (Expanded)

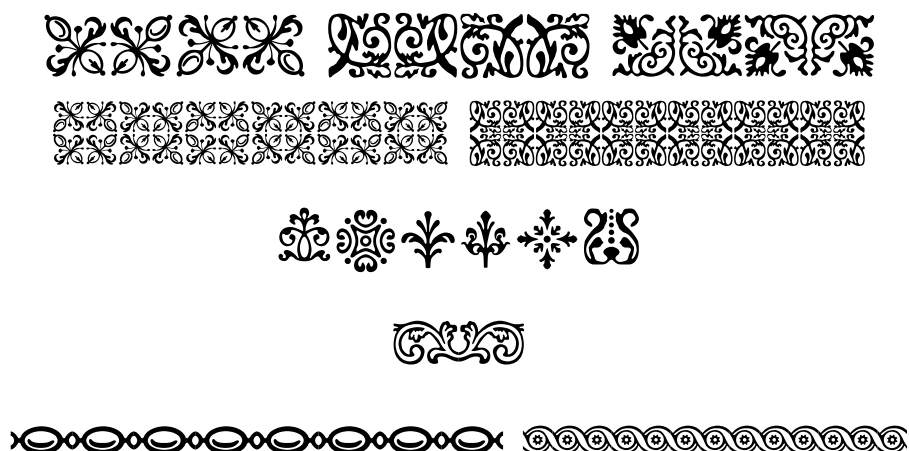
### Polish

*The default shape and position of ogonek in Junicon are suitable for modern Polish. For the medieval Latin e-caudata, consider using cv62.*

Mieszkańcy całej ziemi mieli jedną mowę, czyli jednakowe słowa. A gdy wędrowali ze wschodu, napotkali równinę w kraju Szinear i tam zamieszkali. I mówili jeden do drugiego: Chodźcie, wyrabiamy cegłę i wypalmy ją w ogniu. A gdy już mieli cegłę zamiast kamieni i smołę zamiast zaprawy murarskiej, rzekli: Chodźcie, zbudujemy sobie miasto i wieżę, której wierzchołek będzie sięgał nieba, i w ten sposób uczynimy sobie znak, abyśmy się nie rozproszyli po całej ziemi. (Condensed Bold)

### Fleurons

*Junicon contains a number of fleurons (floral ornaments) copied from a 1785 Caslon specimen book. Access these via the OpenType feature [ornm](#). Fleurons have only one weight and width, and they are the same in roman and italic.*



# Feature Reference

## Introduction

The OpenType features of JunICODE version 2 and its variable counterpart (hereafter referred to together as “JunICODE”) have two purposes. One is to provide convenient access to the rich character set of the Medieval Unicode Font Initiative (MUFI) recommendation. The other is to enable best practices in the presentation of medieval text, promoting accessibility in electronic texts from PDFs to e-books to web pages.

Each character in the MUFI recommendation has a code point<sup>1</sup> associated with it: either the one assigned by Unicode or, where the character is not recognized by Unicode, in the Private Use Area (PUA) of the Basic Multilingual Plane, a block of codes, running from U+E000 to U+F8FF, that are assigned no value by Unicode but instead are available for font designers to use in any way they please.

The problem with PUA code points is precisely their lack of any value. Consider, as a point of comparison, the letter **a** (U+0061). Your computer, your phone, and probably a good many other devices around the house store a good bit of information about this **a**: that it’s a letter in the Latin script, that it’s lowercase, and that the uppercase equivalent is **A** (U+0041). All this information is available to word processors, browsers, and other applications running on your computer.

Now suppose you’re preparing an electronic text containing what MUFI calls LATIN SMALL LETTER NECKLESS A (**a**). It is assigned to code point U+F215, which belongs to the PUA. Beyond that, your computer knows nothing about it: not that it is a variant of **a**, or that it is lowercase, or a letter in the Latin alphabet, or even a character in a language system. A screen reader cannot read, or even spell out, a word with U+F215 in it; a search engine will not recognize the word as containing the letter **a**.

---

<sup>1</sup>A Unicode code point is generally expressed as a four-digit hexadecimal (or base-16) number with a prefix of “U+”. The letter capital “A,” for example, is U+0041 (65 in decimal notation), and lowercase “a” (Middle English yogh) is U+021D.

Junicode offers the full range of MUFI characters—you can enter the PUA code points—but also a solution to the problems posed by those code points. Think of an electronic text (a web page, perhaps, or a PDF) as having two layers: an underlying text, stable and unchanging, and the displayed text, generated by software at the instant it is needed and discarded when it is no longer on the screen. For greatest accessibility the underlying text should contain the plain letter **a** (U+0061) along with markup indicating how it should be displayed. To generate the displayed text, a program called a “layout engine” will (simplifying a bit here) read the markup and apply the OpenType feature **cv02[5]**<sup>2</sup> to the underlying **a**, bypassing the PUA code point, with the result that readers see **a**—the “neckless a.” And yet the letter will still register as **a** with search engines, screen readers, and so on.

This is the Junicode model for text display, but it is not peculiar to Junicode: it is widely considered to be the best practice for displaying text using current font technology.

The full range of OpenType features listed in this document is supported by all major web browsers, LibreOffice, XeTeX, LuaTeX, and (presumably) other document processing applications. All characters listed here are available in Adobe InDesign, though that program supports only a selection of OpenType features. Microsoft Word, unfortunately, supports only Stylistic Sets, ligatures (all but the standard ones in peculiar and probably useless combinations), number variants, and the **Required Features**. In terms of OpenType support, Word is the most primitive of the major text processing applications.

Many MUFI characters cannot be produced by using the OpenType variants of Junicode. These characters fall into three categories:

- Those with Unicode (non-PUA) code points. MUFI has done valuable work obtaining Unicode code points for medieval characters. All such characters (those with hexadecimal codes that *do not* begin with **E** or **F**) are presumed safe to use in accessible and searchable text. However, some of these are covered by Junicode OpenType features for particular reasons.

---

<sup>2</sup>Many OpenType features produce different outcomes depending on an index passed to an application’s layout engine along with the feature tag. Different applications have different ways of entering this index: consult your application’s documentation. Here, the index is recorded in brackets after the feature tag. Users of fontspec (with Xe<sub>La</sub>TeX or Lua<sub>TeX</sub>) should also be aware that fontspec indexes start at zero while OpenType indexes start at one. Therefore all index numbers listed in this document must be reduced by one for use with fontspec.

- Precomposed characters—those consisting of base character + one or more diacritics. For greatest accessibility, these should be entered not as PUA code points, but rather as sequences consisting of base character + one or more diacritics. For example, instead of MUFI U+E498 LATIN SMALL LETTER E WITH DOT BELOW AND ACUTE, use **e** + U+0323 COMBINING DOT BELOW + U+0301 COMBINING ACUTE ACCENT: **é** (when applying combining marks, start with any marks below the character and work downwards, then continue with any marks above the character and work upwards. For example, to make **ô**, place characters in this order: **o**, COMBINING OGONEK U+0328, COMBINING DOT BELOW U+0323, COMBINING MACRON U+0304, COMBINING ACUTE U+0301). Some MUFI characters have marks in unconventional locations, e.g. **ô** LATIN SMALL LETTER O WITH DOT ABOVE AND ACUTE, where the acute appears beside the dot instead of above. This and other characters like it should still be entered as a sequence of base character + marks (here **o**, COMBINING DOT ABOVE U+0307, COMBINING ACUTE U+0301). Junicode will position the marks correctly.
- Characters for which a base character (a Unicode character to which it can be linked) cannot be identified, or for which there may be an inconsistency in the MUFI recommendation. These include:
  - **fl** U+E8AF. This is a ligature of long **s** and **l** with stroke, but there are no base characters with this style of stroke.
  - **ũŨ** U+EFD8 and U+EFD9. MUFI lists these as ligatures (corresponding to the historic ligatures **uu**U, but they cannot be treated as ligatures in the font because a single diacritic is positioned over the glyphs as if they were digraphs like **aa**A).
  - **ppP** U+EBE7 and U+EBE6, for the same reason.
  - **ð** U+F159 LATIN ABBREVIATION SIGN SMALL DE. Neither a variant of **d** nor an eth (**ð**), this character may be a candidate for Unicode encoding.
- Characters for which OpenType programming is not yet available. These will be added as they are located and studied. [Check: U+EBF1, and smcp version.]

These characters should be avoided, even if you are otherwise using MUFI's PUA characters:

- U+F1C5 COMBINING CURL HIGH POSITION. Use U+1DCE COMBINING OGONEK ABOVE. The positioning problem mentioned in the MUFI recommendation is solved in Junicode (and, to be fair, many other fonts with OpenType programming).
- U+F1CA COMBINING DOT ABOVE HIGH POSITION. Use U+0307 COMBINING DOT ABOVE. It will be positioned correctly on any character.

## A. Case-Related Features

### 1. **smcp** – Small Capitals

Converts lowercase letters to small caps; also several symbols and combining marks. All lower- and uppercase pairs (with exceptions noted below) have a small cap equivalent. Lowercase letters without matching caps may lack matching small caps. fghij → FGHIJ.

Note: Precomposed characters defined by MUFI in the Private Use Area have no small cap equivalents. Instead, compose characters using combining diacritics, as outlined in the introduction. For example, **smcp** applied to the sequence **t** + COMBINING OGONEK (U+0328) + COMBINING ACUTE (U+0301) will change **ť** to **ṭ**.

### 2. **c2sc** – Small Capitals from Capitals

Use with **smcp** for all-small-cap text. ABCDE → ABCDE.

### 3. **pcap** – Petite Capitals

Produces small caps in a smaller size than **smcp**. Use these when small caps have to be mixed with lowercase letters. The whole of the basic Latin alphabet is covered, plus several other letters. klmnop → KLMNOP.

### 4. **case** – Case-Sensitive Forms

Produces combining marks that harmonize with capital letters: **Ř**, **Ŷ**, etc. Use of this feature reduces the likelihood that a combining mark will collide with a glyph in the line above.



## 10. zero – Slashed Zero

Produces slashed zero in all number styles: 0 0 0 0. Includes superscripts and subscripts: 0 0 0 0.

## C. Superscripts and Subscripts

### 11. sups – Superscripts

Produces superscript numbers and letters. Only affects lining tabular and oldstyle proportional figures. All lowercase letters of the basic Latin alphabet are covered, and most uppercase letters: 0<sup>123</sup> 4<sup>567</sup> abcde ABDEG. Wherever superscripts are needed (e.g. for footnote numbers), use **sups** instead of the raised and scaled characters generated by some programs. With sups: 4<sup>567</sup>. Scaled: 4<sup>567</sup>.

### 12. subs – Subscripts

Produces subscript numbers. Only affects lining tabular (the default numbers) and old-style proportional figures (use **pnum** and **onum**): 8<sub>901</sub> 2<sub>345</sub>.

## D. Ornaments

### 13. ornm – Ornaments

Produces ornaments (fleurons) in either of two ways: as an indexed variant of the bullet character (U+2022) or as variants of a-z, A-C (all fleurons are available by either method):

As a variant of •: 1=❖, 2=❖, 3=❖, 4=❖, etc., up to 29.

As a variant of a-z, A-C: e=❖, f=❖, g=❖, h=❖, etc.

The method with letters of the alphabet is easier, but the method with bullets will produce a more satisfactory result when text is displayed in an environment where Junicode is not available or **ornm** is not implemented.



## E. Alphabetic Variants

### 14. cv01–cv52 – Basic Latin Variants

These features also affect small cap (**smcp**) and underdotted (**ss07**) forms, where available. Where Junicode has no variant for a Basic Latin letter, the expected **cvNN** feature is skipped, being reserved for future development.

Variant of	cvNN	Variants
A	cv01	1=Ɽ, 2=Ȧ, 3=Ȧ̸
a	cv02	1=Ɽ, 2=Ȧ, 3=Ȧ̸, 4=a, 5=a
B	cv03	No variants available
b	cv04	No variants available
C	cv05	1=Ł
c	cv06	1=Ł
D	cv07	1=Ŏ
d	cv08	1=Ŏ, 2=Ŏ, 3=Ŏ (for 1, see also ss02)
E	cv09	1=Ł, 2=Ł
e	cv10	1=Ł, 2=Ł, 3=Ł
F	cv11	1=Ł
f	cv12	1=Ł, 2=Ł, 3=Ł, 4=Ł, 5=Ł, 6=Ł
G	cv13	1=Ł, 2=Ł, 3=Ł
g	cv14	1=Ł, 2=Ł, 3=Ł, 4=Ł, 5=Ł, 6=Ł, 7=Ł
H	cv15	1=Ł
h	cv16	1=Ł, 2=Ł
I	cv17	1=Ł, 2=Ł
i	cv18	1=Ł, 2=Ł, 3=Ł
J	cv19	1=Ł

# 16 JUnicode

Variant of	cvNN	Variants
j	cv20	1=j, 2=ĵ, 3=j
K	cv21	No variants available
k	cv22	1=k, 2=k̄, 3=ķ, 4=ķ̄
L	cv23	No variants available
l	cv24	1=ł
M	cv25	1=Ṁ, 2=Ṃ, 3=Ṅ
m	cv26	1=m, 2=ṁ, 3=ṃ
N	cv27	1=Ñ
n	cv28	1=ṇ, 2=ṅ, 3=ṇ̄, 4=ṇ̄̄
O	cv29	1=Ō
o	cv30	1=ō
P	cv31	1=Ṗ
p	cv32	No variants available
Q	cv33	1=Ṙ
q	cv34	1=ṙ
R	cv35	1=Ṛ
r	cv36	1=r, 2=ṛ
S	cv37	1=Ṣ, 2=Ṥ
s	cv38	1=ṣ, 2=s̄, 3=ṣ̄, 4=ṣ̄̄, 5=ṣ̄̄̄, 6=ṣ̄̄̄̄
T	cv39	1=Ṭ
t	cv40	1=ṭ, 2=ṭ̄
U	cv41	No variants available
u	cv42	No variants available
V	cv43	No variants available

Variant of	cvNN	Variants
v	cv44	1= <del>v</del> , 2= <del>v</del> , 3= <del>v</del> , 4= <del>v</del>
W	cv45	No variants available
w	cv46	No variants available
X	cv47	No variants available
x	cv48	1= <del>x</del> , 2= <del>x</del> , 3= <del>x</del> , 4= <del>x</del>
Y	cv49	1= <del>Y</del>
y	cv50	1=y, 2=y
Z	cv51	1=Z
z	cv52	1= <del>z</del> , 2= <del>z</del>

## 15. cv53–cv67 – Other Latin Letters

Some features affect both upper- and lowercase forms. **cv62** also affects combining **e** with ogonek, accessible via **ss10** with the entity reference **&\_eogo**; . In this range, **cvNN** features are not reserved for future development, since Junicode already uses or reserves all of the available **cvNN** features.

Variant of	cvNN	Variants
Ą (U+0104)	cv53	1=Ą, 2= <del>Ą</del> , 3=Ą
ą (U+0105)	cv54	1=ą, 2= <del>ą</del>
æ (U+A733)	cv55	1=æ, 2=æ
Æ (U+00C6)	cv56	1=dE
æ (U+00E6)	cv57	1= <del>æ</del> , 2= <del>æ</del> , 3= <del>æ</del>
Ō (U+A734)	cv58	1=dO
ø (U+A735)	cv59	1=dø, 2=xø, 3=xø
ø (U+A739)	cv60	1=xø
đ (U+0111)	cv61	1=d <sup>ʹ</sup>
Ę, ę ... (U+0118, U+0119)	cv62	1=Ę, ę ...; 2=Ę, ę ...

Variant of	cvNN	Variants
ȝ, ȝ (U+021C, U+021D)	cv63	1=ȝ, ȝ
ƿ (U+A749)	cv64	1=ƿ
ø (U+00F8)	cv65	1=o, 2=ø, 3=o, 4=o
þ, þ (U+A765)	cv66	1=þ, þ
?	cv67	Reserved for future use

## 16. ss01 – Alternate thorn and eth

Produces Nordic thorn and eth (þǿÞ) when the language is English, and English thorn and eth (þǿþ) with any other language, reversing the font’s usual usage. This also affects small caps, crossed thorn (þ þ—see also [cv67](#)), combining mark eth (U+1DD9, ̥ ̥), and enlarged thorn and eth (see [ss06](#)). This feature depends on [loca](#) (Localized Forms), which in most applications will always be enabled.

## 17. ss02 – Insular Letter-Forms

Produces insular letter-forms, e.g. ȝƿȝȝȝȝ. Does not affect capitals (except W), as these do not commonly have insular shapes in early manuscripts. For these, enter the Unicode code points or use the Character Variant ([cvNN](#)) features.

## 18. ss04 – High Overline

Produces a high overline over letters used as roman numbers:  $\overline{cdijlmvx} \overline{CDIJLMVXO}$ .

## 19. ss05 – Medium-High Overline

Produces a medium-high overline over (or through the ascenders of) letters used as roman numbers, and some others as well:  $\overline{bcdhijklmfvxp}$ .

## 20. ss06 – Enlarged Minuscles

Lowercase letters that match the height of normal ones, but with a higher x-height, e.g. abcdefg. Covers the whole of the basic Latin alphabet and several other letters: consult the MUFI recommendation for details, and if you are using the variable version of the



## F. Greek

Junicode has two distinct styles of Greek. In the roman face, it is upright and modern, especially designed to harmonize with Junicode’s Latin letters. In the italic, it is slanted and old-style, being based on the eighteenth-century Greek type designed by Alexander Wilson and used by the Foulis Press in Glasgow. Both Greek styles include the full polytonic and monotonic character sets: αβγδεζ αβγδεζ.

To set Greek properly (especially polytonic text) requires that both `locl` and `ccmp` be active, as they should be by default in most text processing applications (but in MS Word they must be explicitly enabled by checking the “kerning” box on the “Advanced” tab of the Font Dialog).

Modern monotonic Greek should be set using only characters from the Unicode “Greek and Coptic” range (U+0370–U+03FF). When monotonic text is set in all caps, Junicode suppresses accents automatically (except in single-letter words, for which you must substitute unaccented forms manually). This substitution is not performed on text containing visually identical letters from the “Greek Extended” range (U+0F00–U+1FFF). Thus when setting polytonic Greek, one should use (for example) Α (U+1FBB), not Α (U+0386), though they look the same.

You can set polytonic Greek either by entering code points from the Greek Extended range or by entering sequences of base characters and diacritics. When using the latter method, you must first make sure the language for the text in question (whether a single word, a short passage, or a complete document) is set to Greek, and then enter characters in canonical order (that is, the sequence defined by Unicode as equivalent to the composite character). The order is as follows: 1. base character; 2. diacritics positioned either above or in front of the base character, working from left to right or bottom to top; 3. the *ypogegrammeni* (U+0345), or for capitals, if you prefer, the *prosgegrammeni* (U+1FBE).

For example, the sequence ω (U+03C9) ͂ (U+0313) ̓ (U+0301) ̈́ (U+0345) produces ω̂̃̄. Substitute capital Ω (U+03A9) and the result is Ω̂̃̄. If you prefer a different result, place U+034F COMBINING GRAPHEME JOINER somewhere in the sequence of combining marks—for example, before the *ypogegrammeni* to make Ω̂̃̄̅.

### 26. ss03 – Alternate Greek

Provides alternate shapes of β γ θ π φ χ ω: β γ θ π φ χ ω. These are chiefly useful in linguistics, as they harmonize with IPA characters.

## G. Punctuation

MUFI encodes nearly twenty marks of punctuation in the PUA. In Junicode these can be accessed in either of two ways: all are indexed variants of . (period), and all are associated with the Unicode marks of punctuation they most resemble (but it should not be inferred that the medieval marks are semantically identical with the Unicode marks, or that there is an etymological relationship between them). The first method will be easier for most to use, but the second is more likely to yield acceptable fallbacks in environments where Junicode is not available.

Marks with Unicode encoding are not included here, as they can safely be entered directly. In MUFI 4.0 several marks have PUA encodings, but have since that release been assigned Unicode code points: *paragraphus* (¶ U+2E4D), medieval comma (; U+2E4C), *punctus elevatus* (: U+2E4E), *virgula suspensiva* (⁂ U+2E4A), triple dagger (‡ U+2E4B).

### 27. ss18 – Old-Style Punctuation Spacing

Colons, semicolons, parentheses, quotation marks and several other glyphs are spaced as in early printed books.

### 28. cv69 – Variants of ꝛꝛ (U+204A / U+2E52, Tironian nota)

1=ꝛꝛ, 2=ꝛꝛ.

### 29. cv70 – Variants of . (period)

1=., 2=., 3=., 4=., 5=., 6=., 7=., 8=., 9=., 10=., 11=., 12=., 13=., 14=., 15=., 16=., 17=., 18=., 19=., 20=.. This feature provides access to all non-Unicode MUFI punctuation marks. Some of them are available via other features (see below).

### 30. cv71 – Variant of · (U+00B7, middle dot)

1=· (*distinctio*).

### 31. cv72 – Variants of , (comma)

1=., 2=.

**32. cv73 – Variants of ; (semicolon)**

1=; (*punctus versus*), 2=., 3=:., 4=;., 5=;̇.

**33. cv74 – Variants of ∷ (U+2E4E, *punctus elevatus*)**

1=∷, 2=∴, 3=∴̇, 4=∴̇ (̇ *punctus flexus*).

**34. cv75 – Variant of ! (exclamation mark)**

1=! (*punctus exclamativus*).

**35. cv76 – Variants of ? (question mark)**

1=¿, 2=¿, 3=¿̇. Shapes of the *punctus interrogativus*.

**36. cv77 – Variant of ~ (ASCII tilde)**

1=~ (same as MUFI U+F1F9, “wavy line”).

**37. cv78 – Variant of \* (asterisk)**

1=∴. MUFI defines U+F1EC as a *signe de renvoi*. Manuscripts employ a number of shapes (of which this is one) for this purpose. Juniconde defines it as a variant of the asterisk—the most common modern *signe de renvoi*.

**38. cv79 – Variants of / (slash)**

1=℄, 2=℄. The first of these is Unicode, U+2E4E.

**H. Abbreviations****39. cv80 – Variant of ꝛ (U+A75D, rum abbreviation)**

1=ꝛ.



#### 40. **cv81** – Variants of ͡ (U+035B, combining zigzag above)

1=͡, 2=͢, 3=ͣ. Positioning of the zigzag can differ from that of other combining marks, e.g. ͤ, ͥ, ͦ. If **calt** “Contextual Alternates” is enabled (as it is by default in most apps), variant forms of **cv81** [2] will be used with several letters, e.g. ͧ, ͨ, ͩ. Enable **case** for forms that harmonize with capitals (ͪ ͫ ͬ ͭ), **smcp** for forms that harmonize with small caps (ͮ ͯ Ͱ ͱ).

#### 41. **cv82** – Variants of Ͱ (U+A770)

1=Ͱ, 2=ͱ. **cv82** [1] produces the baseline *-us* abbreviation (same as MUFI U+F1A6). MUFI also has an uppercase baseline *-us* abbreviation (U+F1A5), but as there is no uppercase version of U+A770 to pair it with, it is indexed separately here.

#### 42. **cv83** – Variants of Ͳ (U+A76B, “et” abbreviation)

1=Ͳ, 2=ͳ. [1] is identical in shape to a semicolon, but as it is semantically the same as U+A76B, it is preferable to use that character with this feature. [2] produces a subscript version of the character, a common variant in printed books.

## I. Combining Marks

#### 43. **cv84** – MUFI combining marks (variants of U+0304)

MUFI encodes a number of combining marks in the PUA (with code points between E000 and F8FF), but when these characters are entered directly, they can interfere with searching and accessibility, and some important applications fail to position them correctly over their base characters. To avoid these problems, enter U+0304 (̄, COMBINING MACRON) and apply **cv84**, with the appropriate index, to both mark and base character. This collection of marks does not include any Unicode-encoded marks (from the “Combining Diacritical Marks” ranges), as these can safely be entered directly. It does include three marks (**cv84** [36], [37] and [38]) that lack MUFI code points but are used to form MUFI characters.

This feature may often appear to have no effect. When this happens it is because an application replaced a sequence like **a U+0304** with a precomposed character like **ā** (U+0101) before Junicode’s OpenType programming had a chance to work. This process is called normalization, and it usually has the effect of simplifying text processing tasks,

but can sometimes prevent the proper functioning of OpenType features. To disable it, place the character U+034F COMBINING GRAPHEME JOINER (don't waste time puzzling over the name) between the base character and the combining mark (or the first combining mark). For example, to produce the combination **ū**, enter **u U+034F U+0304**. (Without U+034F, you would get **ū**).

These marks can sometimes be produced by other **cvNN** features, which may be preferable to **cv84** as providing more suitable fallbacks for applications that do not support Character Variant (**cvNN**) features.

For some marks with PUA code points, users may find it easier to use **entities** than this feature.

These marks are not affected by most other features. This is to preserve flexibility, given the rule that the feature that produces them must be applied to both the mark and the base character. For example, if **smcp** “Small Caps” changed **cv84[11]** <sup>b</sup> to **[12]** <sup>b</sup>, it would be impossible to produce the sequence **N<sup>b</sup>AA** with the diacritic properly positioned.

1=◌̐	11=◌̑	21=◌̒	31=◌̓
2=◌̑	12=◌̒	22=◌̓	32=◌̔
3=◌̒	13=◌̓	23=◌̔	33=◌̕
4=◌̓	14=◌̔	24=◌̕	34=◌̖
5=◌̔	15=◌̕	25=◌̖	35=◌̗
6=◌̕	16=◌̖	26=◌̗	36=◌̘
7=◌̖	17=◌̗	27=◌̘	37=◌̙
8=◌̗	18=◌̘	28=◌̙	
9=◌̘	19=◌̙	29=◌̚	
10=◌̙	20=◌̚	30=◌̛	

#### 44. **ss10** – Entity References for Combining Marks

Instead of **cv84** for representing non-Unicode combining marks, some users may wish to use XML/HTML-style entities. When **ss10** is turned on (preferably for the entire text), these entities appear as combining marks and are correctly positioned over base characters. For example, the letter **e** followed by **&\_eogo;** will yield **ē**. An advantage of entities is that they are (unlike the PUA code points or the indexes of **cv84**) mnemonic and thus easy to use. A disadvantage is that searches cannot ignore combining marks entered by this method as they can using the **cv84** method. (Every method of entering non-Unicode combining marks has disadvantages: users should weigh these, choose a method, and stick with it.)

If you use any of these entities in a work intended for print publication, you should call your publisher's attention to them, since they will likely have their own method of representing them.

<code>&amp;_ansc;</code> →	<code>&amp;_idotl;</code> →	<code>&amp;_orr;</code> →
<code>&amp;_an;</code> →	<code>&amp;_j;</code> →	<code>&amp;_oru;</code> →
<code>&amp;_ar;</code> →	<code>&amp;_jdotl;</code> →	<code>&amp;_q;</code> →
<code>&amp;_arsc;</code> →	<code>&amp;_ksc;</code> →	<code>&amp;_ru;</code> →
<code>&amp;_bsc;</code> →	<code>&amp;_munc;</code> →	<code>&amp;_tsc;</code> →
<code>&amp;_dsc;</code> →	<code>&amp;_oogo;</code> →	<code>&amp;_y;</code> →
<code>&amp;_eogo;</code> →	<code>&amp;_oslash;</code> →	<code>&amp;_thorn;</code> →
<code>&amp;_emac;</code> →	<code>&amp;_omac;</code> →	

## 45. ss20 – Low Diacritics

The MUFI recommendation includes a number of precomposed characters with base letters b, h, k, þ, ð and ð and combining marks (U+0363), (U+0364), (U+0304/cv84[18]), (U+0366), (U+036C), (U+1DE2), (U+036D), (U+036E), (U+1DE6) and (U+0304/cv84[21]). Instead of being positioned above ascender height as usual (e.g. ), the MUFI glyphs have the marks positioned above the x-height (e.g. ). Using the MUFI code points for these precomposed glyphs can interfere with searching and drastically reduce accessibility. Users of Junicode should instead use a sequence of base character + combining mark, and apply **ss20** to the two glyphs. A variant shape of eth () that accommodates the combining mark will be substituted for the normal base character (but this is not necessary for the other base characters). Examples: , , , , , .

**ss20** affects only the diacritics and base characters listed here; other combinations (e.g. , ) are not affected. It will therefore probably be safe to apply this feature to the whole text if it is needed anywhere.

## 46. cv85 – Variant of (U+1DD3, combining open a)

1=.

## 47. cv86 – Variant of (U+1DD8, combining insular d)

1=.

**48. cv87 – Variant of ̂ (U+1DE3, combining r rotunda)**

1=ř.

**49. cv88 – Variant of combining dieresis (U+0308)**

1=ö. This also affects precomposed characters on which this variant dieresis may occur, e.g. ä.

**50. cv89 – Variant of ̄ (U+0305, combining overline)**

1=ô.

**51. cv91 – Variants of short horizontal stroke (U+0335)**

1=⸱, 2=⸫, 3=⸬. This character can be used with letters with ascenders or descenders, e.g. **đ ħ þ p**. **cv91[1]** widens the stroke, and **cv91[2]** and **[3]** offset the stroke to the right or left. Via **calt** “Contextual Alternates,” this offset is performed automatically for many characters with ascenders and descenders, and so it should rarely be necessary to use an index with **cv91**.

**52. cv92 – Variant of breve below (U+032E)**

1=◌̣. Position the mark after the middle of three glyphs, and apply **cv92** to both the mark and (at least) the middle glyph. This mark is not available via **cv84**.

**J. Currency and Weights****53. cv93 – Variants of ₪ (U+0044, generic currency sign)**

1=ℓ	8=gr	15=ℳ	22=₣
2=⌘	9=Φ	16=℔	23=₧
3=⌘	10=⌘	17=₧	24=ℳ
4=ℓ	11=ℳ	18=₧	25=ℳ
5=₧	12=ℳ	19=₧	26=₧
6=₧	13=₧	20=₧	27=₧
7=₧	14=₧	21=₧	



result you want. In that case, it may be necessary to manually edit the result. `fisc flodu ahof` → `ƿlȝk ƿlƿðl ƿnlƿ`.

## 61. `ss13` – Elder Futhark

Changes Latin letters to their equivalents in the Elder Futhark. Because of the variability of the runic alphabet, this method of transliteration may not produce the result you want. In that case, it may be necessary to manually edit the result. `ABCDEFGF` → `ƿβ<MMƿX`.

## 62. `ss14` – Younger Futhark

Changes Latin letters to their equivalents in the Younger Futhark. Because of the variability of the runic alphabet, this method of transliteration may not produce the result you want. In that case, it may be necessary to manually edit the result. `ABCDEFGF` → `†β††††`.

## 63. `ss15` – Long Branch to Short Twig

In combination with `ss14`, converts long branch (the default for the Younger Futhark) to short twig runes: `†β††††` → `††††††`.

## 64. `rtlm` – Right to Left Mirrored Forms

Produces mirrored runes, e.g. `ƿβ<MMƿX` → `XƿMβ>JβJ`. This feature cannot change the direction of text.

# L. Ligatures and Digraphs

## 65. `hlig` – Historic Ligatures

Produces ligatures for combinations that should not ordinarily be rendered as ligatures in modern text.<sup>3</sup> Most of these are from the MUFI recommendation, ranges B.1.1(b)

---

<sup>3</sup>Some fonts define `hlig` differently, as including all ligatures in which at least one element is an archaic character, e.g. those involving long s (f). In Junicode, however, a historic ligature is defined not by the characters it is composed of, but rather by the join between them. If two characters (though modern) should not be joined except in certain historic contexts, they form a historic ligature. If they should be joined in all contexts (even if archaic), the ligature is not historic and should be defined in `liga`.

and B.1.4. This feature does not produce digraphs (which have a phonetic value), for which see [ss17](#). The ligatures:

af→ǻ	ck→ck	hr→ħ	PP→PP	ft̃r→ft̃r
aƿ→ǿ	ðð→ðð	hf→f̃h	pp→pp	f̃p→f̃p
ag→ǿ	ey→ey	hf→f̃h	ƿp→ƿp	ττ→ττ
al→ǻ	fä→f̃ä	kr→k̃r	Ps→P̃s	UE→ŨE
an→ǻ	gd→ǻ	kf→k̃f	ps→p̃s	ue→ũe
aN→ǻ	gð→ǻ	kf→k̃f	Psi→P̃si	UU→ŨU
ap→ǿ	gð→ǻ	ll→ll	ƿi→ƿ̃i	uu→ũu
ar→ǻ	gg→ǻ	nƿ→ñƿ	q̃p→q̃p	pp→p̃p
aR→ǻ	gg→ǻ	oc→ǻ	q̃z/q̃z→q̃z/q̃z	pr→p̃r
ap→ǿ	go→ǻ	Oʀ→Õʀ	Qʀ→Q̃ʀ	ps→p̃s
bb→bb	gp→ǻ	oʀ→ǻ	qʀ→q̃ʀ	p̃f→p̃f
bg→ǻ	gr→ǻ	Oʀ→Õʀ	fä→f̃ä	p̃f→p̃f
ch→ǻ	Hr→H̃r	oʀ→ǻ	fch→f̃ch	p̃p→p̃p

Notes: For **a** and **a** see [cv99](#) above. For the ligature **n** to work properly, U+017F **f** must be entered directly, not by applying an OpenType feature to **s**.

## 66. **dlig** – Discretionary Ligatures

Produces lesser-used ligatures, but also roman numbers, e.g. ii, II, xi, XI. The lesser-used ligatures: **ct**, **fp**, **str**, **st**, **tr**, **tt**, **ty**.

## 67. **ss17** – Rare Digraphs

By “digraph” we mean conjoined letters that represent a phonetic value: the most common examples for western languages are **æ** and **œ** (though these, because they are so common, are not included in this feature). Use of this feature in web pages enables easier searches: for example, producing **ǻu** from **ǻu** allows the word to be searched as “ǻu.” The digraphs covered by this feature are **a**, **æ**, **ai**, **æ**, **æ**, **œ**, **wy**, plus capital and small cap equivalents and digraph + diacritic combinations anticipated in the MUFI recommendation. To produce such a digraph + diacritic combination, either type a letter + diacritic combination as the second element of the digraph or type the diacritic after the second element. For example, **a** + **ú** yields **á**, and so does **a** + **u** + U+0301 (combining acute accent). To produce a digraph + diacritic combination not covered by MUFI (e.g.

à), you may have to place U+034F COMBINING GRAPHEME JOINER (see cv84 above) between the second element of the digraph and the combining mark. Without U+034F: aō. With U+034F: āō.

## N. Required Features

Required features, which provide some of the font’s most basic functionality—ligatures, support for other features, kerning, and more—include **ccmp** (Glyph Composition/Decomposition), **calt** (Contextual Alternates), **liga** (Standard Ligatures), **loca** (Localized Forms), **rlic** (Required Ligatures), **kern** (Horizontal Kerning), and **mark/mkml** (Mark Positioning). In MS Word these features have to be explicitly enabled on the Advanced tab of the Font dialog (Ctrl-D or Cmd-D: enable Kerning, Standard Ligatures, and Contextual Alternates, and the others will be enabled automatically), but in most other applications they are enabled by default.

## O. Non-MUFI Code Points

Characters in Junicode that do not have Unicode code points should be accessed via OpenType features whenever possible. MUFI/PUA code points should be used only in applications that do not support OpenType, or that support it only partially (for example, MS Word). For certain characters that lack either Unicode or MUFI code points, code points in the Supplementary Private Use Area-A (plane 15) are available.

U+F0000 <i>À</i>	U+F0008 <i>Ā</i>	U+F0010 <i>Ḿ</i>	U+F0018 <i>Ṽ</i>
U+F0001 <i>Ȧ</i>	U+F0009 <i>Ē</i>	U+F0011 <i>ō</i>	U+F0019 <i>Ṡ</i>
U+F0002 <i>Ȧ̇</i>	U+F000A <i>ē</i>	U+F0012 <i>ø</i>	U+F001A <i>Ṣ</i>
U+F0003 <i>Ā</i>	U+F000B <i>f</i>	U+F0013 <i>o</i>	U+F001B <i>ṡ</i>
U+F0004 <i>Ā̇</i>	U+F000C <i>ī</i>	U+F0014 <i>o</i>	U+F001C <i>Ṥ</i>
U+F0005 <i>Ȯ</i>	U+F000D <i>j</i>	U+F0015 <i>q</i>	U+F001D <i>Ṥ</i>
U+F0006 <i>Ȯ̇</i>	U+F000E <i>ĵ</i>	U+F0016 <i>ḡ</i>	U+F001E <i>Ṥ</i>
U+F0007 <i>Ȯ̇</i>	U+F000F <i>Ĵ</i>	U+F0017 <i>ṽ</i>	U+F001F <i>Ṥ</i>

*This document was set in 12pt Junicode SemiExpanded  
using the X<sub>Y</sub>LA<sub>T</sub>E<sub>X</sub> typesetting system with fontspec for font management.  
The source for the document, Feature\_Reference.tex, is available at  
<https://github.com/psb1558/Junicode-font>.*