English and Old English Localisation Support for datatool Package

Nicola L. C. Talbot

2025-03-12 version 1.1

Abstract

This is the English and Old English localisation support for the datatool package (version 3.0+). This needs to be installed in addition to datatool. To ensure regional support, you will also need to install datatool-regions.

Contents

1 Introduction

This bundle provides the English and Old English modules for datatool v3.0+. The files simply need to be installed on TEX's path. (They will be ignored if a pre-3.0 version of datatool is installed.) The datatool-base package (which is automatically loaded by datatool) uses tracklang's interface for detecting localisation settings and finding the appropriate files. If you use babel or polyglossia, make sure that you specify the document languages before the first package to load tracklang.

For example:

```
\usepackage[british]{babel}
\usepackage{datatool-base}
```

Alternatively, if you are not using a language package, simply use the locales option. For example:

```
\usepackage[locales={en-GB}]{datatool-base}
```

Any option that can be passed to datatool-base can also be passed to datatool but if datatool-base has already been loaded, it will be too late to use the locales option. For example:

```
\usepackage[locales={en-GB}]{datatool}
```

But not:

Incorrect!

```
\usepackage{datatool-base}
\usepackage[locales={en-GB}]{datatool}
```

If another package that also loads tracklang is loaded first, then datatool-base can pick up the settings from that. For example:

```
\usepackage[en-GB]{datetime2}
\usepackage{datatool}
```

For Old English (Anglo-Saxon), you need the three letter language code “ang” and the four letter script code: “Latn” (Latin) or “Runr” (Runic). Old English support is very limited and mainly provided as an example of how to support extended Latin and non-Latin languages. See section 3 for further information.

If the regional file isn't installed or if no region is associated with the locale then only the language settings will be implemented. For example:

```
\usepackage[english]{babel}
\usepackage{datatool-base}
```

In this case there is no region so only `datatool-english.ldf` will be loaded.

Supplementary packages provided with datatool can also have the locales provided. For example:

```
\documentclass{article}
\usepackage[locales={en}]{datagidx}
\newgidx{index}{Index}
\newterm{élan}
\newterm{elephant}
\newterm{élite}
\newterm{elk}
\begin{document}
\gls{elk}, \gls{élan} \gls{élite} and \gls{elephant}.
\printterms
```

As with datatool, these supplementary packages internally load datatool-base so if that has already been loaded, then the localisation support should already have been set.

The glossaries package also loads datatool-base and, as from glossaries version 4.55, it will check for the new datatool-base commands. If they are defined then `\printnoidxglossary` will switch to this new method of sorting otherwise it will fallback on the old method. So installing datatool-english will not only affect datatool and its associated packages but also glossaries. For example:

```
\documentclass{article}
\usepackage[locales={en},index,style=indexgroup]{glossaries}
\makenoidxglossaries
\newterm{élan}
\newterm{elephant}
\newterm{élite}
\newterm{elk}
\begin{document}
\gls{elk}, \gls{élan} \gls{élite} and \gls{elephant}.
\printnoidxglossary[type=index]
```

Each `datatool-`⟨*lang*⟩`.ldf` language file should redefine:

`\DTLandname`

Should expand to the appropriate translation of “and” (used when formatting comma-separated lists).

Ideally, the language file should also redefine the following commands, which are used by `\DTLgetDataTypeName`, although these are mostly provided for debugging. (The Anglo-Saxon support doesn’t bother to redefine them.)

`\DTLdatatypeunsetname`

Should expand to the appropriate translation of “unknown” (to indicate an unknown data type).

`\DTLdatatypestringname`

Should expand to the appropriate translation of “string” (to indicate a string data type).

`\DTLdatatypeintegername`

Should expand to the appropriate translation of “integer” (to indicate an integer data type).

`\DTLdatatypedecimalname`

Should expand to the appropriate translation of “decimal” (to indicate a decimal data type).

`\DTLdatatypecurrencyname`

Should expand to the appropriate translation of “currency” (to indicate a currency data type).

`\DTLdatatypedatetimename`

Should expand to the appropriate translation of “datetime” (to indicate a datetime data type).

`\DTLdatatypedatename`

Should expand to the appropriate translation of “date” (to indicate a date data type).

`\DTLdatatypetimename`

Should expand to the appropriate translation of “time” (to indicate a time data type).

`\DTLdatatypeinvalidname`

Should expand to the appropriate translation of “invalid” (to indicate an invalid data type identifier).

Additionally, the following commands take arguments that may simply ignore the argument and expand to fixed text:

`\dtlnonlettergroup``{`⟨*char*⟩`}`

Used for non-letter groups. This would typically ignore the argument and expand to the appropriate translation of “Symbols”.

`\dtlnumbergroup``{`⟨*char*⟩`}`

Used for number groups. This would typically ignore the argument and expand to the appropriate translation of “Numbers”.

`\dtlcurrencygroup``{`⟨*char*⟩`}`

Used for currency groups. This would typically ignore the argument and expand to the appropriate translation of “Currency”.

`\dtldatetimegroup``{`⟨*char*⟩`}`

Used for datetime groups. This would typically ignore the argument and expand to the appropriate translation of “Timestamps”.

`\dtldategroup``{`⟨*char*⟩`}`

Used for date groups. This would typically ignore the argument and expand to the appropriate translation of “Dates”.

`\dtltimegroup``{`⟨*char*⟩`}`

Used for time groups. This would typically ignore the argument and expand to the appropriate translation of “Times”.

The `\dtllettergroup` would more typically convert the argument to title case.

2 English (“en”)

The `datatool-english.ldf` file provides support for English orthography (section 2.1), parsing dates and times (experimental, see section 2.2), and for providing fixed text translations.

Aside from the letter group commands, described in section 2.1, the only fixed text for datatool-base is `\DTLandname` and the textual data types used by `\DTLgetDataTypeName`. These are redefined implicitly via the command `\DTLenTranslations`, which allows an alternative definition of `\DTLandname` to be provided without having to keep appending to the caption hook.

`\DTLenLocaleHook`

The `datatool-english.ldf` file provides an intermediary command `\DTLenLocaleHook` that’s automatically added to the captions hook. This performs all the datatool-base English language redefinitions. There should be little reason to actually use this command anywhere unless you’re not using a language package with caption hooks.

Options provided by `datatool-english.ldf` may be set with `\DTLsetLocaleOptions{`**`en`**`}{`⟨*key=val list*⟩`}`. Currently, there is only one option.

`and`=⟨*setting*⟩ (initial word)

This determines the definition of `\DTLandname` within `\DTLenSetAndName` used by the translations hook `\DTLenTranslations`. The value may be: **`word`** (define `\DTLandname` to expand to “and”) or **`amp`** (define `\DTLandname` to expand to `\&`).

Aside from `datatool-english.ldf`, there is also `databib-english.ldf` to provide localisation support for databib (see section 2.3) and `person-english.ldf` to provide localisation support for the person package (see section 2.4).

2.1 Orthography

The `datatool-english.ldf` file provides support for English orthography. This deals with how words are sorted according to the English alphabet. The way that the new datatool v3.0 sorting commands work is to use a handler function that converts the original content into a byte string. Since there are no byte array data types in TeX, this is implemented by converting the original string into an ASCII string in such a way that sorting by character code will produce the desired order.

With the newer `\DTLsortwordlist`, this processing is performed once for each item in the list prior to sorting to allow for a faster character code sort of a temporary sequence variable that stores both the original (actual value) and the transformed (sort value) item. This allows the original item to be restored afterwards. (You can use `\show` or `\clist_show:N` on the comma-separated list command afterwards to double-check this information.)

For example, suppose the document starts with:

datatool v3.0+

```
\documentclass{article}
\usepackage[locales={en}]{datatool}
```

If `datatool-english.ldf` is installed:

With Localisation

```
\newcommand{\mylist}{élan,zebra,elephant,ant,élite,elk}
\DTLsortwordlist{\mylist}{\DTLsortwordcasehandler}
Sorted list: \DTLformatlist{\mylist}.
```

Sorted list: ant, élan, elephant, élite, elk and zebra.

However, if `datatool-english.ldf` isn't installed (or if English support hasn't been requested) the resulting list would be:

No Localisation

```
\newcommand{\mylist}{élan,zebra,elephant,ant,élite,elk}
\DTLsortwordlist{\mylist}{\DTLsortwordcasehandler}
Sorted list: \DTLformatlist{\mylist}.
```

Sorted list: ant, elephant, elk, zebra, élan & élite.

This is because a simple character code comparison has been used. (Note also the use of the default & instead of "and".)

The `\DTLsortwordcasehandler` function is for case-sensitive word sorting, whereas the `\DTLsortwordhandler` function is for case-insensitive word sorting and simply converts each element to lowercase before processing using the locale handler. There are analogous functions for letter-order comparisons which strip spaces and hyphens before processing (see the datatool manual for further details).

The `\DTLsortdata` command, provided by datatool v3.0+ for sorting databases, works in a similar way to `\DTLsortwordlist` and uses the same handler functions. For example:

```
\documentclass{article}
\usepackage[locales={en}]{datatool}

\DTLaction{new}
\DTLaction[ assign = { name = Annie, age = 40 } ]{new row}
\DTLaction[ assign = { name = Ele, age = 80 } ]{new row}
\DTLaction[ assign = { name = Éleanor, age = 28 } ]{new row}
\DTLaction[ assign = { name = Zack, age = 47 } ]{new row}
\DTLaction[ assign = { name = Elva, age = 53 } ]{new row}
\DTLaction[ assign = { name = Æthelwulf, age = 95 } ]{new row}

\DTLsortdata{}{name}
\begin{document}
\DTLaction{display}
```

If `datatool-english.ldf` is installed, then the order will be: Æthelwulf, Annie, Ele, Éleanor, Elva, Zack.

Without `datatool-english.ldf`, the order will be: Annie, Ele, Elva, Zack, Æthelwulf, Éleanor.

If letter groups are required, an extra column can be added to the database with the corresponding letter group obtained from sorting:

```
\DTLsortdata[save-group]{}{name}
```

The letter group is typically the first letter of the sort value or the actual value but this may not be the case for some languages, so the code for obtaining the first letter of a word is adjusted by the language hook.

The groups obtained from sorting are letter groups if the sorted item starts with an alphabetical character. If the value is determined to be a currency, it will be considered part of the currency group, if the value is determined to be a number (without a currency prefix) then it will be considered part of the number group, otherwise it will be in the non-letter group.

The titles for all these groups are obtained with the datatool-base commands `\dtllettergroup`, `\dtlnonlettergroup`, `\dtlnumbergroup` and `\dtlcurrencygroup` which are all redefined by the language hook.

In the case of English, the currency group title simply expands to "Currency", the number group title simply expands to "Numbers", the non-letter group title simply expands to "Symbols". The letter group title expands to its argument converted to title case.

If you need to make any adjustments, the following commands are provided by `datatool-english.ldf` for the handler and groups.

`\DTLenLocaleHandler``{`⟨*tl-var*⟩`}`

Converts the token list variable containing the sort value. You may redefine this command to add support for additional characters, but bear in mind that the more complex the regular expression the longer the document build.

`\DTLenLocaleGetGroupString``{`⟨*actual*⟩`}{`⟨*sort*⟩`}{`⟨*tl-var*⟩`}`

Sets ⟨*tl-var*⟩ to the applicable content in preparation for extracting the initial letter, where ⟨*actual*⟩ is the original value and ⟨*sort*⟩ is the sort value obtained after processing by the handler. (Afterwards, the expansion of ⟨*tl-var*⟩ will then be passed as the ⟨*word*⟩ argument of `\DTLenLocaleGetInitialLetter`.)
For English, this is straight-forward as the sort value is usually appropriate for this. If you redefine this command to set ⟨*tl-var*⟩ to the actual value then accented characters will be considered separate letter groups. This can lead to odd results.

`\DTLenLocaleGetInitialLetter``{`⟨*word*⟩`}{`⟨*tl-var*⟩`}`

Gets the initial letter of ⟨*word*⟩ and stores it in the given token list variable.

`\DTLenSetLetterGroups`

Redefines the group commands. This can be redefined if different titles are required.

2.2 Dates and Times

This is still an experimental feature. As from v3.0, datatool now has additional data types: date, time and datetime. *Parsing for these is* ***off*** *by default but may be enabled.* Without the language support, only ISO dates, times and timestamps can be parsed. The order (dmy, mdy, ymd) is usually set by the region file (provided with datatool-regions). For example, `datatool-GB.ldf` sets the order to dmy whereas `datatool-US.ldf` sets the order to mdy. The region files only deal with numeric dates and times but have a hook to allow supporting language files to extend this to parsing temporal values that contain textual content, such as month names.

The following commands are provided. Note that they use LaTeX3 syntax, which needs to be enabled.

`\datatool_en_get_monthname_map:n``{`⟨*month*⟩`}`

Gets the number associated with the given English month name. For example, January or Jan should result in 1.

`\datatool_en_if_pm:nTF``{`⟨*text*⟩`} {`⟨*true*⟩`} {`⟨*false*⟩`}`

Tests if ⟨*text*⟩ represents afternoon (“pm”, “in the afternoon”, “in the evening”, “midnight”). This indicates that 12 needs to be added to the hour for 12-hour formats. Not that “noon” is not *after* noon (12 noon +12 would result in 24, which is incorrect for noon) and “midnight” would need to have the hour as 12 not 0 (0 midnight +12 would result in 12, which is incorrect for midnight).

`\datatool_en_monthname:n``{`⟨*num*⟩`}`

Expands to the month name for ⟨*num*⟩ (1 for January, etc).

`\datatool_en_shortmonthname:n``{`⟨*num*⟩`}`

Expands to the abbreviated month name. This will be defined to `\DTMenglishshortmonthname` if that command exists (datetime2-english) or to:

`\datatool_en_shortmonthname_dotless:n``{`⟨*num*⟩`}`

Expands to a three-letter abbreviation without a trailing dot.

An alternative command is also provided:

`\datatool_en_shortmonthname_dotted:n``{`⟨*num*⟩`}`

This expands to a three or four letter abbreviation followed by a dot except for short month names, such as May, that aren’t abbreviated.

2.3 Support for databib

The datatool-english bundle provides English language support for the databib package. Options may be set with `\DTLsetLocaleOptions{`**`en/databib`**`}{`⟨*key*=*val list*⟩`}`.

Currently, there is only one option:

`short-month-style``=`⟨*style*⟩ (initial dotted)

This determines the month name abbreviations for databib’s abbrv style. The value may be **`dotted`** (use dotted month abbreviations) or **`dotless`** (use three letter abbreviations with no dot).

`\DataBibEnglish`

The `databib-english.ldf` file defines the intermediary command `\DataBibEnglish` that redefines all the fixed text commands. This command is automatically added to the captions hook if databib is loaded.

2.4 Support for person

`\DataToolPersonEnglish`

The datatool-english bundle provides English language support for the person package. There are currently no options. The `person-english.ldf` file defines the intermediary command `\DataToolPersonEnglish` that redefines all the fixed text commands. This command is automatically added to the captions hook if the person package is loaded.

3 Old English (Anglo-Saxon, “ang”)

This bundle also supplies very limited support for Old English (Anglo-Saxon). Only UTF-8 encoding is supported. This is primarily to provide an example of how to support a language with multiple scripts (or simply a language with an extended Latin or non-Latin script). You will need to identify the script as well as the language. For the Latin script:

```
\usepackage[locales={ang-Latn}]{datatool-base}
```

For the Runic script:

```
\usepackage[locales={ang-Runr}]{datatool-base}
```

Note that the script indicates the script of the input or source text. That is, the text used in the document source code, which may not correspond to the glyphs visible in the PDF file.

For example, suppose a package provides a command called, say `\runic`, which expects Latin characters in the argument but the font encoding ensures that those characters appear as runes in the PDF (for example, the hypothetical command `\runic{fuþorc}` in the source code would be rendered as ᚠᚢᚦᚩᚱᚳ in the PDF). In this case, the source is Latin and so “ang-Latn” is needed when specifying the locale.

If, however, the source code actually contains characters from the Runic Unicode block (with an appropriate font that supports those characters), then the source is Runic and so “ang-Runr” is needed when specifying the locale.

If there is no captions hook, the last locale to be tracked is the one that will be in effect. You can switch using the following:

`\DTLangLatnLocaleHook`

Switch to “ang-Latn” (if `datatool-ang-Latn.ldf` has been loaded, undefined otherwise).

`\DTLangRunrLocaleHook`

Switch to “ang-Runr” (if `datatool-ang-Runr.ldf` has been loaded, undefined otherwise).

The “ang-Latn” localisation support provides the following option, which may be set using `\DTLsetLocaleOptions{`**`ang-Latn`**`}{`⟨*key*=*val list*⟩`}`.

`/ang-Latn/order`=⟨*setting*⟩ (initially harley)

Determines the letter order. The ⟨*setting*⟩ may be: **`harley`** (A-Z, Ƿ, Ð, Æ, Þ), **`stowe`** (ending with Ƿ, Ð, Þ), **`titus`** (ending with Ƿ, Þ, Ð) or **`futhorc`** (starts f, u, þ, o, r, c).

The “ang-Runr” localisation support provides the following option, which may be set using `\DTLsetLocaleOptions{`**`ang-Runr`**`}{`⟨*key*=*val list*⟩`}`.

`/ang-Runr/order`=⟨*setting*⟩ (initially punc-futhorc)

Determines the rune order. The ⟨*setting*⟩ may be: **`punc-unicode`** (punctuation first, followed by Unicode order for the rest), **`punc-futhorc`** (Old English Rune Poem order, punctuation first), or **`futhorc-punc`** (Old English Rune Poem order, punctuation last). Note that the “punc-unicode” order puts the punctuation first, followed by the other runes, even though the punctuation runes actually come towards the end of the Runic block. The Old English Rune Poem order is fuþorc order, with the additional non-poem runes appended at the end. The Tironian et (which isn’t a rune) is added to the end of the non-punctuation runes.

Both ang-Latn and ang-Runr define `\DTLandname` to the Tironian et (⁊). This is done via the following hooks:

`\DTLangLatnTranslations`

This is defined in `datatool-ang-Latn.ldf` to update the definition of `\DTLandname` and is implemented by `\DTLangLatnLocaleHook`.

`\DTLangRunrTranslations`

This is defined in `datatool-ang-Runr.ldf` to update the definition of `\DTLandname` and is implemented by `\DTLangRunrLocaleHook`.

The other language-sensitive name commands aren't provided by the Anglo-Saxon support.

There is no other localisation support (that is, no other fixed text or date/time support). The remaining language-sensitive commands are reset with `\DTLresetLanguage` back to their initial definitions.

Suppose the document uses fontspec with fonts that support the required characters, as follows:

```
\usepackage{fontspec}
\setromanfont{Noto Serif}
\setsansfont{Noto Sans}
\setmonofont{Noto Sans Mono}
\newfontfamily\runefont{Noto Sans Runic}
\NewDocumentCommand{\textrune}{m}{{\runefont #1}}
```

and datatool-base has been loaded with support for both Latin and Runic Anglo-Saxon:

```
\usepackage[locales={ang-Runr,ang-Latn}]{datatool-base}
```

and suppose the command `\myLatnList` is defined to expand to the comma-separated list `f,u,r,c,þ,o` and `\myRunrList` is defined to expand to the comma-separated list `\textrune{ᚢ},\textrune{ᚳ},\textrune{ᚩ},\textrune{ᚠ},\textrune{ᚦ},\textrune{ᚱ}` then these lists can be sorted in the document as follows:

```
\DTLangLatnLocaleHook
\DTLsetLocaleOptions{ang-Latn}{order=harley}
\DTLsortwordlist{\myLatnList}{\DTLsortletterhandler}
Sorted list (Harley): \DTLformatlist{\myLatnList}.

\DTLsetLocaleOptions{ang-Latn}{order=futhorc}
\DTLsortwordlist{\myLatnList}{\DTLsortletterhandler}
Sorted list (fuþorc): \DTLformatlist{\myLatnList}.

\DTLangRunrLocaleHook
\DTLsetLocaleOptions{ang-Runr}{order=futhorc-punc}
\DTLsortwordlist{\myRunrList}{\DTLsortwordhandler}
Sorted list: \DTLformatlist{\myRunrList}.
```

Sorted list (Harley): c, f, o, r, u ⁊ þ.
Sorted list (fuþorc): f, u, þ, o, r ⁊ c.
Sorted list: ᚠ, ᚢ, ᚦ, ᚩ, ᚱ ⁊ ᚳ.

Note that the letter handler is best with "ang-Latn" as there are many Anglo-Saxon words with hyphens. (Bosworth Toller's Anglo-Saxon dictionary ignores hyphens in the order. For example, "á-ǽlan" comes between "aad" and "áǽðan".)

4 The Code

4.1 Root English LDF for datatool-base (`datatool-english.ldf`)

Provide support for English (tracklang root language “english”). NB number group and decimal symbols and currencies should go in region files.

```
\TrackLangProvidesResource{english}[2025/03/12 v1.1 (NLCT)]
```

Try loading `datatool-english-`⟨*encoding*⟩`.ldf`

```
\TrackLangRequestResource{english-\TrackLangEncodingName}
{
```

Not found, fallback on `datatool-english-ascii.ldf`

```
  \csuse{datatool_locale_warn:nn}{datatool-english}%
   {%
     No support for `english' with
     encoding `\TrackLangEncodingName'
     (falling back on US-ASCII)%
   }%
  \TrackLangRequireResource{english-ascii}
}
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Any options may be defined with

```
\datatool_locale_define_keys:nn { en } { ... }
```

or

```
\datatool_locale_define_keys:nn { en / subgroup } { ... }
```

(Syntax as per `\keys_define:nn`)

4.1.1 Orthography rules

NB `\DTLenLocaleHandler` is defined in the applicable encoding file loaded above.

Get the first letter of the word according to the locale’s alphabet and store in token list variable.

```
\newcommand \DTLenLocaleGetInitialLetter [ 2 ]
 {
   \datatool_get_first_letter:nN { #1 } #2
 }
```

`\DTLenLocaleGetGroupString{`⟨*actual*⟩`}{`⟨*sort value*⟩`}{`⟨*tl-var*⟩`}`

Use the sort value rather than the actual value. This ensures that the accents are stripped, but it will mean that the currency and punctuation marks will have their initial marker. Bear in mind that this is only used for values that have been identified as strings. It’s not used by other data types.

```
\newcommand \DTLenLocaleGetGroupString [ 3 ]
 {
   \tl_set:Nn #3 { #2 }
 }
```

Provide intermediary command that defines the letter group commands for English.

```
\newcommand \DTLenSetLetterGroups
 {
  \renewcommand \dtllettergroup [ 1 ]
    { \text_titlecase_first:n { ##1 } }
  \renewcommand \dtlnonlettergroup [ 1 ] { Symbols }
  \renewcommand \dtlnumbergroup [ 1 ] { Numbers }
  \renewcommand \dtlcurrencygroup [ 2 ] { Currency }
  \renewcommand \dtldatetimegroup [ 1 ] { Timestamps }
  \renewcommand \dtldategroup [ 1 ] { Dates }
  \renewcommand \dtltimegroup [ 1 ] { Times }
}
```

```
\newcommand \DTLenSetAndName
 {
  \renewcommand \DTLandname { and }
 }
```

```
\newcommand \DTLenTranslations
 {
  \DTLenSetAndName
  \renewcommand \DTLdatatypeunsetname { unset }
  \renewcommand \DTLdatatypestringname { string }
  \renewcommand \DTLdatatypeintegername { integer }
  \renewcommand \DTLdatatypedecimalname { decimal }
  \renewcommand \DTLdatatypecurrencyname { currency }
  \renewcommand \DTLdatatypedatetimename { date-time }
  \renewcommand \DTLdatatypedatename { date }
  \renewcommand \DTLdatatypetimename { time }
  \renewcommand \DTLdatatypeinvalidname { invalid }
 }
```

Provide options

```
\datatool_locale_define_keys:nn { en }
 {
    and .choice:,
    and / word .code:n =
     {
      \renewcommand \DTLenSetAndName
       {
         \renewcommand \DTLandname { and }
       }
      \tl_if_eq:NnT \l_datatool_current_language_tl { en }
       { \DTLenSetAndName }
     } ,
    and / amp .code:n =
     {
      \renewcommand \DTLenSetAndName
       {
         \renewcommand \DTLandname { \& }
       }
      \tl_if_eq:NnT \l_datatool_current_language_tl { en }
       { \DTLenSetAndName }
     } ,
 }
```

4.1.2 Dates and Times

This section is still experimental.

An appropriate region needs to also be specified and loaded to support parsing English dates. (The region determines dmy, mdy or ymd.)

Provide a shortcut command to get the month name to number mapping.

```
\cs_new:Nn \datatool_en_get_monthname_map:n
 {
   \datatool_region_get_monthname_map:n { en / #1 }
 }
```

Define month name to number mapping for this language.

```
\datatool_region_set_monthname_map:nn { en / January } { 01 }
\datatool_region_set_monthname_map:nn { en / Jan } { 01 }
\datatool_region_set_monthname_map:nn { en / February } { 02 }
\datatool_region_set_monthname_map:nn { en / Feb } { 02 }
\datatool_region_set_monthname_map:nn { en / March } { 03 }
\datatool_region_set_monthname_map:nn { en / Mar } { 03 }
\datatool_region_set_monthname_map:nn { en / April } { 04 }
\datatool_region_set_monthname_map:nn { en / Apr } { 04 }
\datatool_region_set_monthname_map:nn { en / May } { 05 }
\datatool_region_set_monthname_map:nn { en / June } { 06 }
\datatool_region_set_monthname_map:nn { en / Jun } { 06 }
\datatool_region_set_monthname_map:nn { en / July } { 07 }
\datatool_region_set_monthname_map:nn { en / Jul } { 07 }
\datatool_region_set_monthname_map:nn { en / August } { 08 }
\datatool_region_set_monthname_map:nn { en / Aug } { 08 }
\datatool_region_set_monthname_map:nn { en / September } { 09 }
\datatool_region_set_monthname_map:nn { en / Sep } { 09 }
\datatool_region_set_monthname_map:nn { en / Sept } { 09 }
\datatool_region_set_monthname_map:nn { en / October } { 10 }
\datatool_region_set_monthname_map:nn { en / Oct } { 10 }
\datatool_region_set_monthname_map:nn { en / November } { 11 }
\datatool_region_set_monthname_map:nn { en / Nov } { 11 }
\datatool_region_set_monthname_map:nn { en / December } { 12 }
\datatool_region_set_monthname_map:nn { en / Dec } { 12 }
```

This file currently only deals with parsing dates that have month names and possibly weekday names. It does not yet have support for providing formatting (since that can be done with datetime2).

The functions below that go from month number to name are mainly just for the benefit of `databib-english.ldf` although may be used in a custom definition of `\DataToolDateFmt` if required.

If datetime2 has already been loaded, use the command provided by `datetime2-english-base.ldf`.

```
\cs_if_exist:NTF \DTMenglishmonthname
 {
   \cs_new:Nn \datatool_en_monthname:n
    {
      \DTMenglishmonthname { #1 }
    }
 }
 {
   \cs_new:Nn \datatool_en_monthname:n
    {
      \int_case:nn { #1 }
       {
         { 1 } { January }
         { 2 } { February }
         { 3 } { March }
         { 4 } { April }
         { 5 } { May }
         { 6 } { June }
         { 7 } { July }
         { 8 } { August }
         { 9 } { September }
         { 10 } { October }
         { 11 } { November }
         { 12 } { December }
       }
    }
 }
```

Similarly for short month names.

```
 \cs_if_exist:NTF \DTMenglishshortmonthname
  {
    \cs_new:Nn \datatool_en_shortmonthname:n
     {
       \DTMenglishshortmonthname { #1 }
     }
  }
  {
    \cs_new:Nn \datatool_en_shortmonthname:n
     {
       \datatool_en_shortmonthname_dotless:n { #1 }
     }
  }
```

Dotless short month style.

```
 \cs_new:Nn \datatool_en_shortmonthname_dotless:n
  {
    \int_case:nn { #1 }
     {
       { 1 } { Jan }
       { 2 } { Feb }
       { 3 } { Mar }
       { 4 } { Apr }
       { 5 } { May }
       { 6 } { Jun }
       { 7 } { Jul }
       { 8 } { Aug }
```

```
      { 9 } { Sep }
      { 10 } { Oct }
      { 11 } { Nov }
      { 12 } { Dec }
    }
 }
```

Dotted short month style. (Abbreviated month names end with a dot.)

```
\cs_new:Nn \datatool_en_shortmonthname_dotted:n
 {
   \int_case:nn { #1 }
    {
       { 1 } { Jan. }
       { 2 } { Feb. }
       { 3 } { Mar. }
       { 4 } { Apr. }
       { 5 } { May }
       { 6 } { June }
       { 7 } { July }
       { 8 } { Aug. }
       { 9 } { Sept. }
       { 10 } { Oct. }
       { 11 } { Nov. }
       { 12 } { Dec. }
    }
 }
```

Regular expression to match week day name.

```
\regex_const:Nn \c_datatool_en_weekday_name_regex
 {
   Monday | Mon\.? | Tuesday | Tues?\.? | Wednesday | Wed\.?
  | Thursday | Thur?\.? | Friday | Fri\.?
  | Saturday | Sat\.? | Sunday | Sun\.?
 }
```

Regular expression to match month name.

```
\regex_const:Nn \c_datatool_en_month_name_regex
 {
   January | Jan\.? | February | Feb\.? | March | Mar\.?
   | April | Apr\.? | May | June | Jun\.? | July | Jul\.?
   | August | Aug\.? | September | Sept?\.?
   | October | Oct\.? | November | Nov\.? | December | Dec\.?
 }
```

Regular expression to match am/pm identifiers.

```
\regex_const:Nn \c_datatool_en_am_regex
 {
   a \.? m \.? | A \.? M \.? | in \s the \s morning | [nN]oon | [Mm]idday
 }
```

Any match on the following will add 12 to the hour by the pm conditional function:

```
\regex_const:Nn \c_datatool_en_pm_regex
 {
   p \.? m \.? | P \.? M \.? | in \s the \s (?: afternoon | evening )
  | [Mm]idnight
 }
```

Provide conditional.

```
\prg_new_conditional:Npnn \datatool_en_if_pm:n #1
   { T, F, TF }
 {
   \regex_match:NnTF \c_datatool_en_pm_regex { #1 }
    { \prg_return_true: }
    { \prg_return_false: }
 }
```

Match ordinal suffix.

```
\regex_const:Nn \c_datatool_en_ordinal_suffix_regex
 {
   st | nd | rd | th
 }
```

The naming scheme for date regex constants needs to be in the form

`\c_datatool_`⟨*language*⟩`_`⟨*style*⟩`_date_regex` (non-anchored)

and

`\c_datatool_`⟨*language*⟩`_anchored_`⟨*style*⟩`_date_regex` (anchored)

There should be three captured groups in the order that matches ⟨*style*⟩ For example, the style ddmmyyyy should have the day of month number (1-31) in the first captured group, the month in the second group, and the year number in the third. Styles with yyyy should have the century included in the year. Styles with yy need to have the century added to the two-digit year.

Note that the "dd" and "mm" don't mean that two digits are required. (In fact, here the month name not number is expected.) The control sequence names are simply designed to match the naming scheme used by the regionless numeric regex constants provided by `datatime-base.sty`.

```
\regex_const:Nn \c_datatool_en_ddmmyyyy_date_regex
 {
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
   \s+
   ( \ur{c_datatool_en_month_name_regex}  )
    \, ? \s+ ( \d+ )
 }
```

```
\regex_const:Nn \c_datatool_en_anchored_ddmmyyyy_date_regex
 {
   \A \s*
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
   \s+
   ( \ur{c_datatool_en_month_name_regex}  )
    \, ? \s+ ( \d+ )
   \s* \Z
 }
```

```
\regex_const:Nn \c_datatool_en_ddmmyy_date_regex
 {
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
   \s+
   ( \ur{c_datatool_en_month_name_regex}  )
    \, ? \s+ \ur{c_datatool_apostrophe_regex} ? ( \d{2} )
 }
\regex_const:Nn \c_datatool_en_anchored_ddmmyy_date_regex
 {
   \A \s*
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
   \s+
   ( \ur{c_datatool_en_month_name_regex}  )
    \, ? \s+ \ur{c_datatool_apostrophe_regex} ? ( \d{2} )
   \s* \Z
 }
\regex_const:Nn \c_datatool_en_mmddyyyy_date_regex
 {
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_en_month_name_regex}  )
   \s+
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
    \, ? \s+ ( \d+ )
 }
\regex_const:Nn \c_datatool_en_anchored_mmddyyyy_date_regex
 {
   \A \s*
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_en_month_name_regex}  )
   \s+
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
    \, ? \s+ ( \d+ )
   \s* \Z
 }
\regex_const:Nn \c_datatool_en_mmddyy_date_regex
 {
   (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
   ( \ur{c_datatool_en_month_name_regex}  )
   \s+
   ( \ur{c_datatool_day_of_month_regex} )
   \ur{c_datatool_en_ordinal_suffix_regex} ?
    \, ? \s+ \ur{c_datatool_apostrophe_regex} ? ( \d{2} )
 }
\regex_const:Nn \c_datatool_en_anchored_mmddyy_date_regex
 {
   \A \s*
```

```
    (?: \ur{c_datatool_en_weekday_name_regex} \, ? \s+ ) ?
    ( \ur{c_datatool_en_month_name_regex}  )
    \s+
    ( \ur{c_datatool_day_of_month_regex} )
    \ur{c_datatool_en_ordinal_suffix_regex} ?
     \, ? \s+ \ur{c_datatool_apostrophe_regex} ? ( \d{2} )
    \s* \Z
  }
```

Year first formats are a bit unlikely with month names but provided for completeness.

```
\regex_const:Nn \c_datatool_en_yyyymmdd_date_regex
  {
    ( \d+ ) \s+
    ( \ur{c_datatool_en_month_name_regex}  )
    \s+
    ( \ur{c_datatool_day_of_month_regex} )
    \ur{c_datatool_en_ordinal_suffix_regex} ?
    (?: \, ? \s+ \ur{c_datatool_en_weekday_name_regex} ) ?
  }
```

```
\regex_const:Nn \c_datatool_en_anchored_yyyymmdd_date_regex
  {
    \A \s*
    ( \d+ ) \s+
    ( \ur{c_datatool_en_month_name_regex}  )
    \s+
    ( \ur{c_datatool_day_of_month_regex} )
    \ur{c_datatool_en_ordinal_suffix_regex} ?
    (?: \, ? \s+ \ur{c_datatool_en_weekday_name_regex} ) ?
    \s* \Z
  }
```

This is a bit unusual but provided for completeness.

```
\regex_const:Nn \c_datatool_en_yymmdd_date_regex
  {
    \ur{c_datatool_apostrophe_regex} ? ( \d{2} ) \s+
    ( \ur{c_datatool_en_month_name_regex}  )
    \s+
    ( \ur{c_datatool_day_of_month_regex} )
    \ur{c_datatool_en_ordinal_suffix_regex} ?
    (?: \, ? \s+ \ur{c_datatool_en_weekday_name_regex} ) ?
  }
```

```
\regex_const:Nn \c_datatool_en_anchored_yymmdd_date_regex
  {
    \A \s*
    \ur{c_datatool_apostrophe_regex} ? ( \d{2} ) \s+
    ( \ur{c_datatool_en_month_name_regex}  )
    \s+
    ( \ur{c_datatool_day_of_month_regex} )
    \ur{c_datatool_en_ordinal_suffix_regex} ?
    (?: \, ? \s+ \ur{c_datatool_en_weekday_name_regex} ) ?
    \s* \Z
  }
```

The naming scheme for time regex constants needs to be in the form

\c_datatool_⟨*language*⟩_⟨*variant*⟩_hhmmss_time_regex (non-anchored)

and

\c_datatool_⟨*language*⟩_⟨*variant*⟩_anchored_hhmmss_time_regex (anchored)

Both the minutes and seconds are optional here. These will default to 00 if omitted.

```
\regex_const:Nn \c_datatool_en_colon_hhmmss_time_regex
 {
   ( \ur{c_datatool_hour_regex} )
   (?: \: ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \: ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \s*
     (
         \ur{c_datatool_en_am_regex}
       | \ur{c_datatool_en_pm_regex}
     )
   ) ?
 }
```

```
\regex_const:Nn \c_datatool_en_colon_anchored_hhmmss_time_regex
 {
   \A \s*
   ( \ur{c_datatool_hour_regex} )
   (?: \: ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \: ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \s*
     (
         \ur{c_datatool_en_am_regex}
       | \ur{c_datatool_en_pm_regex}
     )
    ) ?
   \s* \Z
 }
```

```
\regex_const:Nn \c_datatool_en_dot_hhmmss_time_regex
 {
   ( \ur{c_datatool_hour_regex} )
   (?: \. ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \. ( \ur{c_datatool_minsec_regex} ) ) ?
   (?: \s*
     (
         \ur{c_datatool_en_am_regex}
       | \ur{c_datatool_en_pm_regex}
     )
    ) ?
 }
```

```
\regex_const:Nn \c_datatool_en_dot_anchored_hhmmss_time_regex
 {
   \A \s*
   ( \ur{c_datatool_hour_regex} )
   (?: \. ( \ur{c_datatool_minsec_regex} ) ) ?
```

```
    (?: \. ( \ur{c_datatool_minsec_regex} ) ) ?
    (?: \s*
      (
          \ur{c_datatool_en_am_regex}
        | \ur{c_datatool_en_pm_regex}
      )
     ) ?
    \s* \Z
  }
```

```
\newcommand \DTLenLocaleHook
  {
   \renewcommand
     \DTLCurrentLocaleWordHandler
     { \DTLenLocaleHandler }
   \renewcommand
     \DTLCurrentLocaleGetInitialLetter
     { \DTLenLocaleGetInitialLetter }
   \renewcommand
     \DTLCurrentLocaleGetGroupString
     { \DTLenLocaleGetGroupString }
   \DTLenSetLetterGroups
   \let
    \DTLCurrentLocaleGetMonthNameMap
    \datatool_en_get_monthname_map:n
   \let
    \DTLCurrentLocaleIfpmTF
    \datatool_en_if_pm:nTF
```

Allow region files to construct control sequence names based on this naming scheme:

```
   \tl_set:Nn \l_datatool_current_language_tl { en }
```

Textual commands:

```
   \DTLenTranslations
  }
```

```
\ExplSyntaxOff
```

```
\TrackLangAddToCaptions{\DTLenLocaleHook}
```

4.2 Root English US-ASCII LDF for datatool-base (`datatool-english-ascii.ldf`)

```
\TrackLangProvidesResource{english-ascii}[2025/03/12 v1.1 (NLCT)]
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Minimal support for `\DTLenLocaleHandler`.

```
\newcommand \DTLenLocaleHandler [ 1 ]
  {
```

Ensure punctuation characters are grouped together by prefixing them with straight double-quote. This ensures that they will all be placed before any alphabetical characters.

```
   \regex_replace_all:nnN
```

Version 1.1 added category code check.

```
    { \cO([[:punct:]]) } { \cO\" \1 }
   #1
  }
```

Switch off LaTeX3 syntax.

```
\ExplSyntaxOff
```

4.3 Root English ISO 8859-1 LDF for datatool-base (`datatool-english-latin1.ldf`)

```
\TrackLangProvidesResource{english-latin1}[2025/03/12 v1.1 (NLCT)]
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Provide Latin-1 version of `\DTLenLocaleHandler`.

```
\newcommand \DTLenLocaleHandler [ 1 ]
{
  \regex_replace_case_all:nN
  {
    { [ \x{C0} - \x{C5} ] } { A }
    { \x{C6} } { AE }
    { \x{C7} } { C }
    { [ \x{C8} - \x{CB} ] } { E }
    { [ \x{CC} - \x{CF} ] } { I }
    { \x{D1} } { N }
    { [ \x{D2} - \x{D6} \x{D8} ] } { O }
    { [ \x{D9} - \x{DC} ]} { U }
    { \x{DD} } { Y }
```

Version 1.1: treat eth as thorn

```
    { \x{DE} | \x{D0} } { TH }
    { \x{DF} } { ss }
    { [ \x{E0} - \x{E5} ] } { a }
    { \x{E6} } { ae }
    { \x{E7} } { c }
    { [ \x{E8} - \x{EB} ] } { e }
    { [ \x{EC} - \x{EF} ] } { i }
    { \x{F1} } { n }
    { [ \x{F2} - \x{F6} \x{F8} ] } { o }
    { [ \x{F9} - \x{FC} ] } { u }
    { [ \x{FD} \x{FF} ] } { y }
```

Version 1.1: treat eth as thorn

```
    { \x{FE} | \x{F0} } { th }
```

Ensure currency symbols are grouped together by prefixing all currency symbols with `$` but don't change math shift. Version 1.1 added math shift check.

```
    { \cM(.) } { \1 }
    { (\ur{l_datatool_currencysigns_regex}) } { \cO\$ \1 }
```

Ensure punctuation characters are grouped together by prefixing them with straight double-quote. This ensures that they will all be placed before any alphabetical characters. Version 1.1 added category code check.

```
    { \cO([[:punct:]]) } { \cO\" \1 }
  }
  #1
}
```

Switch off LaTeX3 syntax.

```
\ExplSyntaxOff
```

4.4 Root English UTF-8 LDF for datatool-base `datatool-english-utf8.ldf`

```
\TrackLangProvidesResource{english-utf8}[2025/03/12 v1.1 (NLCT)]
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Only provide limited support for extended characters to allow for foreign names. The more cases added, the longer the processing time.

```
\newcommand \DTLenLocaleHandler [ 1 ]
{
  \regex_replace_case_all:nN
  {
    {À|Á|Â|Ã|Ä|Å} {A}
    {Æ} {AE}
    {Ç} {C}
    {È|É|Ê|Ë} {E}
    {Ì|Í|Î|Ï} {I}
    {Ñ} {N}
    {Ò|Ó|Ô|Õ|Ö|Ø} {O}
    {Ù|Ú|Û|Ü} {U}
    {Ý} {Y}
```

Version 1.1: treat eth as thorn

```
    {Þ|Ð} {TH}
    {ß} {ss}
    {à|á|â|ã|ä|å} {a}
    {æ} {ae}
    {ç} {c}
    {è|é|ê|ë} {e}
    {ì|í|î|ï} {i}
    {ñ} {n}
    {ò|ó|ô|õ|ö|ø} {o}
    {ù|ú|û|ü} {u}
    {ý|ÿ} {y}
```

Version 1.1: treat eth as thorn

```
    {þ|ð} {th}
    {ſ} {s}
    {Ĳ} {Ij}
    {ĳ} {ij}
    {Ł} {L}
    {ł} {l}
    {Œ} {OE}
    {œ} {oe}
    {Ƿ} {W}
    {ƿ} {w}
```

Ensure currency symbols are grouped together by prefixing all currency symbols with $ but don't change math shift. Version 1.1 added math shift check.

```
    { \cM(.) } { \1 }
    { (\ur{l_datatool_currencysigns_regex}) } { \cO\$ \1 }
```

Ensure common punctuation characters are grouped together by prefixing them with straight double-quote. This ensures that they will all be placed before any alphabetical characters.

Treat closing quote the same as straight apostrophe.

```
    { ’ } { \cO\" \cO\' }
```

Treat opening quote the same as backtick.

```
    { ‘ } { \cO\" \cO\` }
```

Treat opening and closing double quotes the same as straight double-quote.

```
    { (“|”) } { \cO\" \cO\" }
```

Treat em-dash and en-dash the same as hyphen/minus.

```
    { (—|–) } { \cO\" \cO\- }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
    { \cO([[:punct:]]) } { \cO\" \1 }
  }
  #1
}
```

Switch off LaTeX3 syntax:

```
\ExplSyntaxOff
```

4.5 Canadian English LDF for datatool-base (`datatool-en-CA.ldf`)

```
\TrackLangProvidesResource{en-CA}[2025/03/12 v1.1 (NLCT)]
```

Load root language file. Note that datatool-base should automatically load the region file `datatool-CA.ldf` if it is installed, so only `datatool-english.ldf` needs to be loaded (if not already done so).

```
\TrackLangRequireResource{english}
```

Need LaTeX3 syntax on:

```
\ExplSyntaxOn
```

Provide both official and unofficial options for the number group and decimal characters. Official style has a space number group and a decimal dot. This will allow a normal space, `\,` (thin space command) or the thin space Unicode character U+2009 as the number group separator when parsing and will use `\,` when formatting.

```
\cs_new:Nn \datatool_en_CA_set_numberchars_official:
 {
   \datatool_set_thinspace_group_decimal_char:n { . }
 }
```

Unofficial style has a comma number group and a decimal dot.

```
\cs_new:Nn \datatool_en_CA_set_numberchars_unofficial:
 {
   \DTLsetnumberchars { , } { . }
 }
```

Provide intermediate command to set the applicable number group and decimal character.

```
\newcommand \datatoolenCASetNumberChars
 {
   \bool_if:NT \l_datatool_region_set_numberchars_bool
    {
      \datatool_en_CA_set_numberchars_official:
    }
 }
```

Provide `\DTLsetLocaleOptions` en-CA option to switch between styles.

```
\datatool_locale_define_keys:nn { en-CA }
 {
   number-style .choices:nn =
    { official , unofficial }
    {
      \exp_args:NNe \renewcommand
        \datatoolenCASetNumberChars
        {
           \exp_not:N \bool_if:NT
            \exp_not:N \l_datatool_region_set_numberchars_bool
             {
               \exp_not:c { datatool_en_CA_set_numberchars_ \l_keys_choice_tl : }
             }
         }
      \datatool_if_current_lang_region:nnT { en } { CA }
       {
         \datatoolenCASetNumberChars
       }
    } ,
}
```

Provide intermediary command to add to the captions hook.

```
\newcommand \DTLenCALocaleHook
 {
   \datatoolenCASetNumberChars
 }
```

LaTeX3 syntax no longer needed.

```
\ExplSyntaxOff
```

Add to captions hook (this is needed for languages but not language-less regions).

```
\TrackLangAddToCaptions{\DTLenCALocaleHook}
```

4.6 South African English LDF for datatool-base (`datatool-en-ZA.ldf`)

```
\TrackLangProvidesResource{en-ZA}[2025/03/12 v1.1 (NLCT)]
```

Load root language file. Note that datatool-base should automatically load the region file `datatool-ZA.ldf` if it is installed, so only `datatool-english.ldf` needs to be loaded (if not already done so).

```
\TrackLangRequireResource{english}
```

Need LaTeX3 syntax on:

```
\ExplSyntaxOn
```

Provide dialect options for the number group and decimal characters. Space style has a space number group and a decimal dot. This will allow a normal space, `\,` (thin space command) or the thin space Unicode character U+2009 as the number group separator when parsing and will use `\,` when formatting.

```
\cs_new:Npn \datatool_en_ZA_set_numberchars_thinspace:
 {
   \datatool_set_thinspace_group_decimal_char:n { . }
 }
```

Comma style has a comma number group and a decimal dot.

```
\cs_new:Npn \datatool_en_ZA_set_numberchars_comma:
 {
   \DTLsetnumberchars { , } { . }
```

```
  }
```

Provide intermediate command to set the applicable number group and decimal character.

```
\newcommand \datatoolenZASetNumberChars
  {
    \bool_if:NT \l_datatool_region_set_numberchars_bool
     {
       \datatool_en_ZA_set_numberchars_thinspace:
     }
  }
```

Provide `\DTLsetLocaleOptions` en-ZA option to switch between styles.

```
\datatool_locale_define_keys:nn { en-ZA }
  {
    number-style .choices:nn =
     { thinspace , comma }
     {
       \exp_args:NNe \renewcommand
         \datatoolenZASetNumberChars
         {
            \exp_not:N \bool_if:NT
             \exp_not:N \l_datatool_region_set_numberchars_bool
              {
                \exp_not:c { datatool_en_ZA_set_numberchars_ \l_keys_choice_tl : }
              }
          }
```

Note that this will be reset when the dialect changes unless `\DTLsetLocaleOptions{ZA}{number-style=dialect}` is used.

```
       \datatool_if_current_lang_region:nnT { en } { ZA }
        {
          \datatoolenZASetNumberChars
        }
     } ,
  }
```

Provide a hook using the required naming scheme.

```
\newcommand \DTLenZALocaleHook
  {
    \datatoolenZASetNumberChars
  }
```

LaTeX3 syntax no longer needed.

```
\ExplSyntaxOff
```

Add to captions hook (this is needed for languages but not language-less regions).

```
\TrackLangAddToCaptions{\DTLenZALocaleHook}
```

4.7 Root English LDF for databib (`databib-english.ldf`)

```
\TrackLangProvidesResource{english}[2025/03/12 v1.1 (NLCT)]
```

LaTeX3 commands needed:

```
\ExplSyntaxOn
```

Month names used by the abbrv style.

```
\cs_new:Nn \databib_en_shortmonthname:n
  {
    \datatool_en_shortmonthname_dotted:n { #1 }
  }
```

Define `\DTLsetLocaleOptions` options for en/databib. The only option deals with switching the short month names for the abbrv style between three-letter dotless names and abbreviated names that end with a dot.

```
\datatool_locale_define_keys:nn { en / databib }
  {
    short-month-style .choice: ,
    short-month-style / dotted .code:n =
     {
       \cs_set:Nn \databib_en_shortmonthname:n
        {
          \datatool_en_shortmonthname_dotted:n { #1 }
        }
     } ,
    short-month-style / dotless .code:n =
     {
       \cs_set:Nn \databib_en_shortmonthname:n
        {
          \datatool_en_shortmonthname_dotless:n { #1 }
        }
     } ,
  }
```

Fixed text translations.

```
\newcommand \DataBibEnglish
  {
    \tl_set:Nn \ofname { of }
    \tl_set:Nn \inname { in }
    \tl_set:Nn \etalname { et ~ al. }
    \tl_set:Nn \editorname { editor }
    \tl_set:Nn \editorsname { editors }
    \tl_set:Nn \volumename { volume }
    \tl_set:Nn \numbername { number }
    \tl_set:Nn \pagesname { pages }
    \tl_set:Nn \pagename { page }
    \tl_set:Nn \editionname { edition }
    \tl_set:Nn \techreportname { Technical ~ report }
    \tl_set:Nn \mscthesisname { Master's ~ thesis }
    \tl_set:Nn \phdthesisname { PhD ~ thesis }
    \tl_set:Nn \DTLbibaccessedname { accessed }
    \renewcommand \dtl@monthname [ 1 ]
     {
       \datatool_en_monthname:n { ##1 }
     }
    \renewcommand \dtl@abbrvmonthname [ 1 ]
     {
       \databib_en_shortmonthname:n { ##1 }
     }
  }
```

Finished with LaTeX3 commands.

```
\ExplSyntaxOff
```

Add to captions hook.

```
\TrackLangAddToCaptions{\DataBibEnglish}
```

4.8 Root English LDF for person (`person-english.ldf`)

```
\TrackLangProvidesResource{english}[2025/03/12 v1.1 (NLCT)]
```

If `\DTLsetLocaleOptions` options need to be provided, the sub-module should be "person". For example,

```
\datatool_locale_define_keys:nn { en / person } {⟨key-value list⟩}
```

Provide intermediate command to add to captions hook.

```
\newcommand{\DataToolPersonEnglish}{%
```

Third person pronouns

```
  \PersonSetLocalisation{male}{pronoun}{he}%
  \PersonSetLocalisation{female}{pronoun}{she}%
  \PersonSetLocalisation{nonbinary}{pronoun}{they}%
  \PersonSetLocalisation{unknown}{pronoun}{they}%
  \PersonSetLocalisation{male}{pluralpronoun}{they}%
  \PersonSetLocalisation{female}{pluralpronoun}{they}%
  \PersonSetLocalisation{nonbinary}{pluralpronoun}{they}%
  \PersonSetLocalisation{unknown}{pluralpronoun}{they}%
```

Third person objective pronouns

```
  \PersonSetLocalisation{male}{objpronoun}{him}%
  \PersonSetLocalisation{female}{objpronoun}{her}%
  \PersonSetLocalisation{nonbinary}{objpronoun}{them}%
  \PersonSetLocalisation{unknown}{objpronoun}{them}%
  \PersonSetLocalisation{male}{pluralobjpronoun}{them}%
  \PersonSetLocalisation{female}{pluralobjpronoun}{them}%
  \PersonSetLocalisation{nonbinary}{pluralobjpronoun}{them}%
  \PersonSetLocalisation{unknown}{pluralobjpronoun}{them}%
```

Third person possessive adjectives

```
  \PersonSetLocalisation{male}{possadj}{his}%
  \PersonSetLocalisation{female}{possadj}{her}%
  \PersonSetLocalisation{nonbinary}{possadj}{their}%
  \PersonSetLocalisation{unknown}{possadj}{their}%
  \PersonSetLocalisation{male}{pluralpossadj}{their}%
  \PersonSetLocalisation{female}{pluralpossadj}{their}%
  \PersonSetLocalisation{nonbinary}{pluralpossadj}{their}%
  \PersonSetLocalisation{unknown}{pluralpossadj}{their}%
```

Third person possessive pronouns

```
  \PersonSetLocalisation{male}{posspronoun}{his}%
  \PersonSetLocalisation{female}{posspronoun}{hers}%
  \PersonSetLocalisation{nonbinary}{posspronoun}{theirs}%
  \PersonSetLocalisation{unknown}{posspronoun}{theirs}%
  \PersonSetLocalisation{male}{pluralposspronoun}{theirs}%
  \PersonSetLocalisation{female}{pluralposspronoun}{theirs}%
  \PersonSetLocalisation{nonbinary}{pluralposspronoun}{theirs}%
  \PersonSetLocalisation{unknown}{pluralposspronoun}{theirs}%
```

Second person pronouns

```
  \PersonSetLocalisation{male}{pronoun2}{you}%
  \PersonSetLocalisation{female}{pronoun2}{you}%
  \PersonSetLocalisation{nonbinary}{pronoun2}{you}%
  \PersonSetLocalisation{unknown}{pronoun2}{you}%
```

```
\PersonSetLocalisation{male}{pluralpronoun2}{you}%
\PersonSetLocalisation{female}{pluralpronoun2}{you}%
\PersonSetLocalisation{nonbinary}{pluralpronoun2}{you}%
\PersonSetLocalisation{unknown}{pluralpronoun2}{you}%
```

Second person objective pronouns

```
\PersonSetLocalisation{male}{objpronoun2}{you}%
\PersonSetLocalisation{female}{objpronoun2}{you}%
\PersonSetLocalisation{nonbinary}{objpronoun2}{you}%
\PersonSetLocalisation{unknown}{objpronoun2}{you}%
\PersonSetLocalisation{male}{pluralobjpronoun2}{you}%
\PersonSetLocalisation{female}{pluralobjpronoun2}{you}%
\PersonSetLocalisation{nonbinary}{pluralobjpronoun2}{you}%
\PersonSetLocalisation{unknown}{pluralobjpronoun2}{you}%
```

Second person possessive adjective

```
\PersonSetLocalisation{male}{possadj2}{your}%
\PersonSetLocalisation{female}{possadj2}{your}%
\PersonSetLocalisation{nonbinary}{possadj2}{your}%
\PersonSetLocalisation{unknown}{possadj2}{your}%
\PersonSetLocalisation{male}{pluralpossadj2}{your}%
\PersonSetLocalisation{female}{pluralpossadj2}{your}%
\PersonSetLocalisation{nonbinary}{pluralpossadj2}{your}%
\PersonSetLocalisation{unknown}{pluralpossadj2}{your}%
```

Second person possessive pronoun

```
\PersonSetLocalisation{male}{posspronoun2}{yours}%
\PersonSetLocalisation{female}{posspronoun2}{yours}%
\PersonSetLocalisation{nonbinary}{posspronoun2}{yours}%
\PersonSetLocalisation{unknown}{posspronoun2}{yours}%
\PersonSetLocalisation{male}{pluralposspronoun2}{yours}%
\PersonSetLocalisation{female}{pluralposspronoun2}{yours}%
\PersonSetLocalisation{nonbinary}{pluralposspronoun2}{yours}%
\PersonSetLocalisation{unknown}{pluralposspronoun2}{yours}%
```

Relationships

```
\PersonSetLocalisation{male}{child}{son}%
\PersonSetLocalisation{female}{child}{daughter}%
\PersonSetLocalisation{nonbinary}{child}{child}%
\PersonSetLocalisation{unknown}{child}{child}%
\PersonSetLocalisation{male}{pluralchild}{sons}%
\PersonSetLocalisation{female}{pluralchild}{daughters}%
\PersonSetLocalisation{nonbinary}{pluralchild}{children}%
\PersonSetLocalisation{unknown}{pluralchild}{children}%
\PersonSetLocalisation{male}{parent}{father}%
\PersonSetLocalisation{female}{parent}{mother}%
\PersonSetLocalisation{nonbinary}{parent}{parent}%
\PersonSetLocalisation{unknown}{parent}{parent}%
\PersonSetLocalisation{male}{pluralparent}{fathers}%
\PersonSetLocalisation{female}{pluralparent}{mothers}%
\PersonSetLocalisation{nonbinary}{pluralparent}{parents}%
\PersonSetLocalisation{unknown}{pluralparent}{parents}%
\PersonSetLocalisation{male}{sibling}{brother}%
\PersonSetLocalisation{female}{sibling}{sister}%
\PersonSetLocalisation{nonbinary}{sibling}{sibling}%
\PersonSetLocalisation{unknown}{sibling}{sibling}%
```

```
  \PersonSetLocalisation{male}{pluralsibling}{brothers}%
  \PersonSetLocalisation{female}{pluralsibling}{sisters}%
  \PersonSetLocalisation{nonbinary}{pluralsibling}{siblings}%
  \PersonSetLocalisation{unknown}{pluralsibling}{siblings}%
```

Gender translations

```
  \PersonSetLocalisation{male}{gender}{male}%
  \PersonSetLocalisation{female}{gender}{female}%
  \PersonSetLocalisation{nonbinary}{gender}{non-binary}%
  \PersonSetLocalisation{unknown}{gender}{unknown}%
```

Gender identification labels for `\newperson`

```
  \PersonSetMaleLabels{Male,MALE,M,m}%
  \PersonSetFemaleLabels{Female,FEMALE,F,f}%
  \PersonSetNonBinaryLabels{non-binary,Nonbinary,Non-Binary,NONBINARY,NON-BINARY,N,n}%
}
```

Add to captions hook.

```
\TrackLangAddToCaptions{\DataToolPersonEnglish}
```

4.9 Root Old English (Anglo Saxon) LDF for datatool-base (`datatool-anglosaxon.ldf`)

Provide support for Old English (tracklang root language “anglosaxon”). This is mainly provided as an example for dealing with languages that have multiple scripts and non-ASCII letters. As such, it doesn’t provide an translations or date/time formatting.

The file `datatool-ang-`⟨*script*⟩`.ldf` should already have been found if installed correctly. This is a fallback to catch an unrecognised script and encoding combination.

```
\TrackLangProvidesResource{anglosaxon}[2025/03/12 v1.1 (NLCT)]
```

Try loading `datatool-ang-`⟨*script*⟩`-`⟨*encoding*⟩`.ldf`

```
\TrackLangRequestResource{ang-\CurrentTrackedDialectScript-\TrackLangEncodingName}
{%
```

Not found.

```
  \csuse{datatool_locale_warn:n}{datatool-anglosaxon}%
   {%
     No support for `anglosaxon' with script `\CurrentTrackedDialectScript'
     and encoding `\TrackLangEncodingName'%
   }%
}
```

4.10 Root Old English (Anglo Saxon) Latin Script LDF for datatool-base (`datatool-ang-Latn.ldf`)

Provide support for Old English (tracklang root language “anglosaxon”) using Latin script. This is mainly provided as an example for dealing with languages that have multiple scripts and non-ASCII letters. As such, it doesn’t provide any translations (except for “and”) or date/time formatting.

```
\TrackLangProvidesResource{ang-Latn}[2025/03/12 v1.1 (NLCT)]
```

Try loading `datatool-ang-Latn-`⟨*encoding*⟩`.ldf`

```
\TrackLangRequestResource{ang-Latn-\TrackLangEncodingName}
{%
```

Not found.

```
   \csuse{datatool_locale_warn:n}{datatool-ang-Latn}%
    {%
      No support for `anglosaxon' with script `Latn'
      and encoding `\TrackLangEncodingName'.%
    }%
   \endinput
 }
```

Switch on LaTeX3 syntax.

```
 \ExplSyntaxOn
```

Any options may be defined with

```
\datatool_locale_define_keys:nn { ang-Latn } { ... }
```

or

```
\datatool_locale_define_keys:nn { ang-Latn / subgroup } { ... }
```

(Syntax as per `\keys_define:nn`)

4.10.1 Orthography rules

NB `\DTLangLatnLocaleHandler` is defined in the applicable encoding file loaded above.

Get the first letter of the word according to the locale's alphabet and store in token list variable.

```
 \newcommand \DTLangLatnLocaleGetInitialLetter [ 2 ]
  {
    \datatool_get_first_letter:nN { #1 } #2
  }
```

`\DTLangLatnLocaleGetGroupString{`⟨*actual*⟩`}{`⟨*sort value*⟩`}{`⟨*tl*⟩`}`

Get the text that will be used to obtain the letter group (the actual value not the sort value).

```
 \newcommand \DTLangLatnLocaleGetGroupString [ 3 ]
  {
    \datatool_sort_preprocess:Nn #3 { #1 }
    \datatool_angLatn_process_letter_group:N #3
    \bool_lazy_and:nnT
      { \tl_if_head_eq_charcode_p:nN { #1 } - }
      { \bool_not_p:n { \tl_if_head_eq_charcode_p:nN { #2 } " } }
     {
       \exp_args:NNe \tl_set:Nn #3 { \tl_tail:N #3 }
     }
  }

 \newcommand \DTLangLatnLocaleHook
  {
   \DTLresetLanguage
   \renewcommand
```

```
    \DTLCurrentLocaleWordHandler
    { \DTLangLatnLocaleHandler }
  \renewcommand
    \DTLCurrentLocaleGetInitialLetter
    { \DTLangLatnLocaleGetInitialLetter }
  \renewcommand
    \DTLCurrentLocaleGetGroupString
    { \DTLangLatnLocaleGetGroupString }
  \DTLangLatnTranslations
```

Allow region files to construct control sequence names based on this naming scheme:

```
  \tl_set:Nn \l_datatool_current_language_tl { angLatn }
 }
```

```
\ExplSyntaxOff
```

```
\TrackLangAddToCaptions{\DTLangLatnLocaleHook}
```

4.11 Root Anglo Saxon Latin Script UTF-8 LDF for datatool-base (datatool-ang-Latn-u ldf)

```
\TrackLangProvidesResource{ang-Latn-utf8}[2025/03/12 v1.1 (NLCT)]
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Only provide limited support for additional characters to allow for foreign names. The more cases added, the longer the processing time.

The extended Latin characters and characters derived from Runic are multi-byte which means that with inputenc they will be treated as multiple tokens. This means that the characters need to be converted into single-byte characters such that the character code of those single-byte characters have the same ordering (numerically) as the Latin characters. Remember that the result isn't intended for typesetting but is just intended for a character code comparison.

```
\newcommand \DTLangLatnLocaleHandler [ 1 ]
{
  \datatool_angLatn_order_harley:N #1
}
```

Harley order: A-Z, Ƿ, Ð, Æ, Þ

```
\cs_new:Nn \datatool_angLatn_order_harley:N
 {
  \regex_replace_case_all:nN
  {
    {À|Á|Â|Ã|Ä|Ā} {A}
    {Ç|Ċ} {C}
    {È|É|Ê|Ë|Ē} {E}
    {Ì|Í|Î|Ï|Ī} {I}
    {Ñ} {N}
    {Ò|Ó|Ô|Õ|Ö} {O}
    {Œ} {OE}
    {Ù|Ú|Û|Ü|Ū} {U}
    {Ý} {Y}
    {Ƿ|Ꝩ} { \cL\[ }
    {Ð} { \cL\\ }
    {Æ|Ǣ|Ǽ|Æ̀|Æ̂} { \cL\] }
```

```
     {Þ|Ꝥ} { \cL\^ }
     {à|á|â|ã|ä|ā} {a}
     {ç|ċ} {c}
     {è|é|ê|ë|ē} {e}
     {ì|í|î|ï|ī} {i}
     {ñ} {n}
     {ò|ó|ô|õ|ö} {o}
     {œ} {oe}
     {ſ} {s}
     {ß} {ss}
     {ù|ú|û|ü|ū} {u}
     {ý} {y}
     {ƿ|ᵹ} { \cL\{ }
     {ð} { \cL\| }
     {æ|ǣ|ǽ|æ̀|æ̂} { \cL\} }
     {þ|ꝥ} { \cL\~ }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
     { \cO([[:punct:]]) } { \cO\" \1 }
   }
   #1
  }
```

Stowe order has A-Z, Ƿ, Ð, Þ (not sure where Æ goes so I've made it equivalent to AE for now).

```
 \cs_new:Nn \datatool_angLatn_order_stowe:N
  {
   \regex_replace_case_all:nN
   {
     {À|Á|Â|Ã|Ä|Ą} {A}
     {Æ|Ǣ|Ǽ|Æ̀|Æ̂} {AE}
     {Ç|Ċ} {C}
     {È|É|Ê|Ë|Ę} {E}
     {Ì|Í|Î|Ï} {I}
     {Ñ} {N}
     {Ò|Ó|Ô|Õ|Ö} {O}
     {Œ} {OE}
     {Ꞇ|Ꞇ̃} {T}
     {Ù|Ú|Û|Ü} {U}
     {Ý} {Y}
     {Ƿ|Ᵹ} { \cL\[ }
     {Ð} { \cL\\ }
     {Þ|Ꝥ} { \cL\] }
     {à|á|â|ã|ä|ą} {a}
     {æ|ǣ|ǽ|æ̀|æ̂} {ae}
     {ç|ċ} {c}
     {è|é|ê|ë|ę} {e}
     {ì|í|î|ï|ı} {i}
     {ñ} {n}
     {ò|ó|ô|õ|ö} {o}
     {œ} {oe}
     {ꞇ|ꞇ̃} {t}
     {ù|ú|û|ü} {u}
     {ý} {y}
     {ƿ|ᵹ} { \cL\{ }
```

```
      {ð} { \cL\| }
      {þ|ꝥ} { \cL\} }
      {⁊} { \cL\~ }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
      { \cO([[:punct:]]) } { \cO\" \1 }
    }
    #1
  }
```

Titus has order A-Z, Ƿ–Þ–Ð (again, not sure where Æ goes).

```
\cs_new:Nn \datatool_angLatn_order_titus:N
  {
   \regex_replace_case_all:nN
   {
      {À|Á|Â|Ã|Ä|Ą} {A}
      {Æ|Ǣ|Ǽ|Æ̀|Æ̂} {AE}
      {Ç|Ċ} {C}
      {È|É|Ê|Ë|Ę} {E}
      {Ì|Í|Î|Ï} {I}
      {Ñ} {N}
      {Ò|Ó|Ô|Õ|Ö} {O}
      {Œ} {OE}
      {Ꞇ|Ꞇ̃} {T}
      {Ù|Ú|Û|Ü} {U}
      {Ý} {Y}
      {Ƿ|Ỽ} { \cL\[ }
      {Ð} { \cL\\ }
      {Þ|Ꝥ} { \cL\] }
      {à|á|â|ã|ä|ą} {a}
      {æ|ǣ|ǽ|æ̀|æ̂} {ae}
      {ç|ċ} {c}
      {è|é|ê|ë|ę} {e}
      {ì|í|î|ï|ı} {i}
      {ñ} {n}
      {ò|ó|ô|õ|ö} {o}
      {œ} {oe}
      {ꞇ|ꞇ̃} {t}
      {ù|ú|û|ü} {u}
      {ý} {y}
      {ƿ|ỽ} { \cL\{ }
      {ð} { \cL\| }
      {þ|ꝥ} { \cL\} }
      {⁊} { \cL\~ }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
      { \cO([[:punct:]]) } { \cO\" \1 }
    }
    #1
  }
```

Fuþorc order.

```
\cs_new:Nn \datatool_angLatn_order_futhorc:N
  {
```

```
  \regex_replace_case_all:nN
  {
    {f} {\cL\x{31}}
    {u} {\cL\x{32}}
    {þ} {\cL\x{33}}
    {o} {\cL\x{34}}
    {r} {\cL\x{35}}
    {c} {\cL\x{36}}
    {g} {\cL\x{37}}
    {w} {\cL\x{38}}
    {h} {\cL\x{39}}
    {n} {\cL\:}
    {i} {\cL\;}
    {j} {\cL\<}
    {ï} {\cL\=}
    {p} {\cL\>}
    {x} {\cL\@}
    {st} {\cL\x{51}}
    {s} {\cL\x{41}}
    {t} {\cL\x{42}}
    {b} {\cL\x{43}}
    {ea} {\cL\x{4C}}
    {e} {\cL\x{44}}
    {m} {\cL\x{45}}
    {l} {\cL\x{46}}
    {ŋ} {\cL\x{47}}
    {œ} {\cL\x{48}}
    {d} {\cL\x{49}}
    {a} {\cL\x{4A}}
    {æ} {\cL\x{4B}}
    {y} {\cL\x{4D}}
    {k} {\cL\x{4E}}
    {ḡ} {\cL\x{4F}}
    {q} {\cL\x{50}}
    {ę|ᴇ} {\cL\x{52}}
    {į} {\cL\x{53}}
    {ī} {\cL\x{54}}
    {c̄|k̄} {\cL\x{55}}
    {⁊} {\cL\~}
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
    { \cO([[:punct:]]) } { \cO\" \1 }
  }
  #1
 }
```

We also need to merge accented characters for the letter groups. Note that since we're only concerned with the first letter, we only need a single replace not replace all.

```
\cs_new:Nn \datatool_angLatn_process_letter_group:N
 {
  \regex_replace_case_once:nN
  {
    {À|Á|Â|Ã|Ä|Ā} {A}
    {Ç|Ċ} {C}
    {È|É|Ê|Ë|Ē} {E}
    {Ì|Í|Î|Ï|Ī} {I}
    {Ñ} {N}
    {Ò|Ó|Ô|Õ|Ö} {O}
    {Œ} {OE}
    {Ù|Ú|Û|Ü|Ū} {U}
    {Ý} {Y}
    {Æ|Ǣ|Ǽ|Æ̀|Æ̂} {Æ}
    {Ƿ|Ɣ} {Ƿ}
    {Þ|Ꝥ} {Þ}
    {à|á|â|ã|ä|ā} {a}
    {ç|ċ} {c}
    {è|é|ê|ë|ē} {e}
    {ì|í|î|ï|ī} {i}
    {ñ} {n}
    {ò|ó|ô|õ|ö} {o}
    {œ} {oe}
    {ſ} {s}
    {ß} {ss}
    {ù|ú|û|ü|ū} {u}
    {ý} {y}
    {ƿ|ɣ} {ƿ}
    {æ|ǣ|ǽ|æ̀|æ̂} {æ}
    {þ|ꝥ} {þ}
  }
  #1
 }
\datatool_locale_define_keys:nn { ang-Latn }
 {
    order .choice:,
    order / harley .code:n =
     {
       \renewcommand \DTLangLatnLocaleHandler [1]
        {
          \datatool_angLatn_order_harley:N ##1
        }
     },
    order / stowe .code:n =
     {
       \renewcommand \DTLangLatnLocaleHandler [1]
        {
          \datatool_angLatn_order_stowe:N ##1
        }
     },
    order / titus .code:n =
     {
       \renewcommand \DTLangLatnLocaleHandler [1]
        {
          \datatool_angLatn_order_titus:N ##1
        }
     },
    order / futhorc .code:n =
     {
       \renewcommand \DTLangLatnLocaleHandler [1]
        {
          \datatool_angLatn_order_futhorc:N ##1
        }
     },
 }
```

Translations: only providing “and” (Tironian et)

```
\newcommand \DTLangLatnTranslations
{
  \tl_set:Nn \DTLandname { ⁊ }
}
```

Switch off LaTeX3 syntax.

```
\ExplSyntaxOff
```

4.12 Root Old English (Anglo Saxon) Runic Script LDF for datatool-base (`datatool-ang-Runr.ldf`)

Provide support for Old English (tracklang root language “anglosaxon”) using Runic script.

This is mainly provided as an example for dealing with languages that have multiple scripts and non-ASCII letters. As such, it doesn’t provide any translations (except for “and”) or date/time formatting.

> Identifying “Runr” as the script indicates that the text source uses Runic characters. If a package, such as allrunes, is used that maps Latin characters to runes (for example, `\textarm{fuÞark}`) then the script should be set to “Latn”.

```
\TrackLangProvidesResource{ang-Runr}[2025/03/12 v1.1 (NLCT)]
```

Try loading `datatool-ang-Runr-`⟨*encoding*⟩`.ldf`

```
\TrackLangRequestResource{ang-Runr-\TrackLangEncodingName}
{%
```

Not found.

```
  \csuse{datatool_locale_warn:n}{datatool-ang-Runr}%
   {%
     No support for `anglosaxon' with script `Runr'
     and encoding `\TrackLangEncodingName'.%
   }%
  \endinput
}
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

Any options may be defined with

```
\datatool_locale_define_keys:nn { ang-Runr } { ... }
```

or

```
\datatool_locale_define_keys:nn { ang-Runr / subgroup } { ... }
```

(Syntax as per `\keys_define:nn`)

4.12.1 Orthography rules

NB `\DTLangRunrLocaleHandler` is defined in the applicable encoding file loaded above.

Get the first letter of the word according to the locale's alphabet and store in token list variable.

```
\newcommand \DTLangRunrLocaleGetInitialLetter [ 2 ]
 {
   \datatool_get_first_grapheme:nN { #1 } #2
 }
```

`\DTLangRunrLocaleGetGroupString{`⟨*actual*⟩`}{`⟨*sort value*⟩`}{`⟨*tl*⟩`}`

Get the text that will be used to obtain the letter group (the actual value not the sort value).

```
\newcommand \DTLangRunrLocaleGetGroupString [ 3 ]
 {
   \datatool_sort_preprocess:Nn #3 { #1 }
 }
```

```
\newcommand \DTLangRunrLocaleHook
 {
  \DTLresetLanguage
  \renewcommand
    \DTLCurrentLocaleWordHandler
    { \DTLangRunrLocaleHandler }
  \renewcommand
    \DTLCurrentLocaleGetInitialLetter
    { \DTLangRunrLocaleGetInitialLetter }
  \renewcommand
    \DTLCurrentLocaleGetGroupString
    { \DTLangRunrLocaleGetGroupString }
  \DTLangRunrTranslations
```

Allow region files to construct control sequence names based on this naming scheme:

```
  \tl_set:Nn \l_datatool_current_language_tl { angRunr }
 }
```

```
\ExplSyntaxOff
```

```
\TrackLangAddToCaptions{\DTLangRunrLocaleHook}
```

4.13 Root Anglo Saxon Runic Script UTF-8 LDF for datatool-base (`datatool-ang-Runr-u` `ldf`)

```
\TrackLangProvidesResource{ang-Runr-utf8}[2025/03/12 v1.1 (NLCT)]
```

Switch on LaTeX3 syntax.

```
\ExplSyntaxOn
```

The runic characters are multi-byte which means that with inputenc they will be treated as multiple tokens. This means that the characters need to be converted into single-byte characters such that the character code of those single-byte characters have the same ordering (numerically) as the runic characters. Remember that the result isn't intended for typesetting but is just intended for a character code comparison.

```
\newcommand \DTLangRunrLocaleHandler [ 1 ] {
  \datatool_angRunr_order_punc_futhorc:n { #1 }
}
```

If PDFLaTeX, we can't match greater than FF with `\x`, so need different regular expressions depending on whether or not we have a native Unicode engine.

```
\datatool_if_unicode_engine:TF
 {
```

Unicode order (mostly Unicode order). The punctuation characters are placed first, the rest follows Unicode order.

```
  \cs_new:Nn \datatool_angRunr_order_punc_unicode:n
   {
    \regex_replace_case_all:nN
    {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

```
      { \x{16A0} } { \cL\x{31} }
```

U+16A1 RUNIC LETTER V

```
      { \x{16A1} } { \cL\x{32} }
```

U+16A2 RUNIC LETTER URUZ UR U

```
      { \x{16A2} } { \cL\x{33} }
```

U+16A3 RUNIC LETTER YR

```
      { \x{16A3} } { \cL\x{34} }
```

U+16A4 RUNIC LETTER Y

```
      { \x{16A4} } { \cL\x{35} }
```

U+16A5 RUNIC LETTER W

```
      { \x{16A5} } { \cL\x{36} }
```

U+16A6 RUNIC LETTER THURISAZ THURS THORN

```
      { \x{16A6} } { \cL\x{37} }
```

U+16A7 RUNIC LETTER ETH

```
      { \x{16A7} } { \cL\x{38} }
```

U+16A8 RUNIC LETTER ANSUZ A

```
      { \x{16A8} } { \cL\x{39} }
```

U+16A9 RUNIC LETTER OS O

```
      { \x{16A9} } { \cL\x{3A} }
```

U+16AA RUNIC LETTER AC A

```
      { \x{16AA} } { \cL\x{3B} }
```

U+16AB RUNIC LETTER AESC

```
      { \x{16AB} } { \cL\x{3C} }
```

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

```
      { \x{16AC} } { \cL\x{3D} }
```

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

```
      { \x{16AD} } { \cL\x{3E} }
```

U+16AE RUNIC LETTER O

```
        { \x{16AE} } { \cL\x{3F} }
```

U+16AF RUNIC LETTER OE

```
        { \x{16AF} } { \cL\x{40} }
```

U+16B0 RUNIC LETTER ON

```
        { \x{16B0} } { \cL\x{41} }
```

U+16B1 RUNIC LETTER RAIDO RAD REID R

```
        { \x{16B1} } { \cL\x{42} }
```

U+16B2 RUNIC LETTER KAUNA

```
        { \x{16B2} } { \cL\x{43} }
```

U+16B3 RUNIC LETTER CEN

```
        { \x{16B3} } { \cL\x{44} }
```

U+16B4 RUNIC LETTER KAUN K

```
        { \x{16B4} } { \cL\x{45} }
```

U+16B5 RUNIC LETTER G

```
        { \x{16B5} } { \cL\x{46} }
```

U+16B6 RUNIC LETTER ENG

```
        { \x{16B6} } { \cL\x{47} }
```

U+16B7 RUNIC LETTER GEBO GYFU G

```
        { \x{16B7} } { \cL\x{48} }
```

U+16B8 RUNIC LETTER GAR

```
        { \x{16B8} } { \cL\x{49} }
```

U+16B9 RUNIC LETTER WUNJO WYNN W

```
        { \x{16B9} } { \cL\x{4A} }
```

U+16BA RUNIC LETTER HAGLAZ H

```
        { \x{16BA} } { \cL\x{4B} }
```

U+16BB RUNIC LETTER HAEGL H

```
        { \x{16BB} } { \cL\x{4C} }
```

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H

```
        { \x{16BC} } { \cL\x{4D} }
```

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H

```
        { \x{16BD} } { \cL\x{4E} }
```

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

```
        { \x{16BE} } { \cL\x{4F} }
```

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N

```
        { \x{16BF} } { \cL\x{50} }
```

U+16C0 RUNIC LETTER DOTTED-N

```
        { \x{16C0} } { \cL\x{51} }
```

U+16C1 RUNIC LETTER ISAZ IS ISS I

```
        { \x{16C1} } { \cL\x{52} }
```

U+16C2 RUNIC LETTER E

`{ \x{16C2} } { \cL\x{53} }`

U+16C3 RUNIC LETTER JERAN J

`{ \x{16C3} } { \cL\x{54} }`

U+16C4 RUNIC LETTER GER

`{ \x{16C4} } { \cL\x{55} }`

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE

`{ \x{16C5} } { \cL\x{56} }`

U+16C6 RUNIC LETTER SHORT-TWIG-AR A

`{ \x{16C6} } { \cL\x{57} }`

U+16C7 RUNIC LETTER IWAZ EOH

`{ \x{16C7} } { \cL\x{58} }`

U+16C8 RUNIC LETTER PERTHO PEORTH P

`{ \x{16C8} } { \cL\x{59} }`

U+16C9 RUNIC LETTER ALGIZ EOLHX

`{ \x{16C9} } { \cL\x{5A} }`

U+16CA RUNIC LETTER SOWILO S

`{ \x{16CA} } { \cL\x{5B} }`

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

`{ \x{16CB} } { \cL\x{5C} }`

U+16CC RUNIC LETTER SHORT-TWIG-SOL S

`{ \x{16CC} } { \cL\x{5D} }`

U+16CD RUNIC LETTER C

`{ \x{16CD} } { \cL\x{5E} }`

U+16CE RUNIC LETTER Z

`{ \x{16CE} } { \cL\x{5F} }`

U+16CF RUNIC LETTER TIWAZ TIR TYR T

`{ \x{16CF} } { \cL\x{60} }`

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T

`{ \x{16D0} } { \cL\x{61} }`

U+16D1 RUNIC LETTER D

`{ \x{16D1} } { \cL\x{62} }`

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

`{ \x{16D2} } { \cL\x{63} }`

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B

`{ \x{16D3} } { \cL\x{64} }`

U+16D4 RUNIC LETTER DOTTED-P

`{ \x{16D4} } { \cL\x{65} }`

U+16D5 RUNIC LETTER OPEN-P

`{ \x{16D5} } { \cL\x{66} }`

U+16D6 RUNIC LETTER EHWAZ EH E
`{ \x{16D6} } { \cL\x{67} }`

U+16D7 RUNIC LETTER MANNAZ MAN M
`{ \x{16D7} } { \cL\x{68} }`

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M
`{ \x{16D8} } { \cL\x{69} }`

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M
`{ \x{16D9} } { \cL\x{6A} }`

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L
`{ \x{16DA} } { \cL\x{6B} }`

U+16DB RUNIC LETTER DOTTED-L
`{ \x{16DB} } { \cL\x{6C} }`

U+16DC RUNIC LETTER INGWAZ
`{ \x{16DC} } { \cL\x{6D} }`

U+16DD RUNIC LETTER ING
`{ \x{16DD} } { \cL\x{6F} }`

U+16DE RUNIC LETTER DAGAZ DAEG D
`{ \x{16DE} } { \cL\x{70} }`

U+16DF RUNIC LETTER OTHALAN ETHEL O
`{ \x{16DF} } { \cL\x{71} }`

U+16E0 RUNIC LETTER EAR
`{ \x{16E0} } { \cL\x{72} }`

U+16E1 RUNIC LETTER IOR
`{ \x{16E1} } { \cL\x{73} }`

U+16E2 RUNIC LETTER CWEORTH
`{ \x{16E2} } { \cL\x{74} }`

U+16E3 RUNIC LETTER CALC
`{ \x{16E3} } { \cL\x{75} }`

U+16E4 RUNIC LETTER CEALC
`{ \x{16E4} } { \cL\x{76} }`

U+16E5 RUNIC LETTER STAN
`{ \x{16E5} } { \cL\x{77} }`

U+16E6 RUNIC LETTER LONG-BRANCH-YR
`{ \x{16E6} } { \cL\x{78} }`

U+16E7 RUNIC LETTER SHORT-TWIG-YR
`{ \x{16E7} } { \cL\x{79} }`

U+16E8 RUNIC LETTER ICELANDIC-YR
`{ \x{16E8} } { \cL\x{7A} }`

U+16E9 RUNIC LETTER Q
`{ \x{16E9} } { \cL\x{7B} }`

U+16EA RUNIC LETTER X

```
    { \x{16EA} } { \cL\x{7C} }
```

U+16EE RUNIC ARLAUG SYMBOL

```
    { \x{16EE} } { \cL\x{7D} \cO\! }
```

U+16EF RUNIC TVIMADUR SYMBOL

```
    { \x{16EF} } { \cL\x{7D} \cO\" }
```

U+16F0 RUNIC BELGTHOR SYMBOL

```
    { \x{16F0} } { \cL\x{7D} \cO\# }
```

U+16F1 RUNIC LETTER K

```
    { \x{16F1} } { \cL\x{7D} \cO\$ }
```

U+16F2 RUNIC LETTER SH

```
    { \x{16F2} } { \cL\x{7D} \cO\% }
```

U+16F3 RUNIC LETTER OO

```
    { \x{16F3} } { \cL\x{7D} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
    { \x{16F4} } { \cL\x{7D} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
    { \x{16F5} } { \cL\x{7D} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
    { \x{16F6} } { \cL\x{7D} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
    { \x{16F7} } { \cL\x{7D} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
    { \x{16F8} } { \cL\x{7D} \cO\+ }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
    { \x{16EB} } { \cO\! \cO\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
    { \x{16EC} } { \cO\! \cO\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
    { \x{16ED} } { \cO\! \cO\# }
```

Tironian et

```
    { \x{204A} } { \cL\x{7E} }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
    { \cO([[:punct:]]) } { \cO\! \cO\$ \1 }
  }
  #1
 }
```

Old English Rune Poem order (fuþorc). Not all characters are included in the poem, so they are appended at the end. Punctuation characters come first.

```
\cs_new:Nn \datatool_angRunr_order_punc_futhorc:n
 {
  \regex_replace_case_all:nN
  {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

`{ \x{16A0} } { \cL\x{31} }`

U+16A2 RUNIC LETTER URUZ UR U

`{ \x{16A2} } { \cL\x{32} }`

U+16A6 RUNIC LETTER THURISAZ THURS THORN

`{ \x{16A6} } { \cL\x{33} }`

U+16A9 RUNIC LETTER OS O

`{ \x{16A9} } { \cL\x{34} }`

U+16B1 RUNIC LETTER RAIDO RAD REID R

`{ \x{16B1} } { \cL\x{35} }`

U+16B3 RUNIC LETTER CEN

`{ \x{16B3} } { \cL\x{36} }`

U+16B7 RUNIC LETTER GEBO GYFU G

`{ \x{16B7} } { \cL\x{37} }`

U+16B9 RUNIC LETTER WUNJO WYNN W

`{ \x{16B9} } { \cL\x{38} }`

U+16BB RUNIC LETTER HAEGL H

`{ \x{16BB} } { \cL\x{39} }`

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

`{ \x{16BE} } { \cL\x{3A} }`

U+16C1 RUNIC LETTER ISAZ IS ISS I

`{ \x{16C1} } { \cL\x{3B} }`

U+16C4 RUNIC LETTER GER

`{ \x{16C4} } { \cL\x{3C} }`

U+16C7 RUNIC LETTER IWAZ EOH

`{ \x{16C7} } { \cL\x{3D} }`

U+16C8 RUNIC LETTER PERTHO PEORTH P

`{ \x{16C8} } { \cL\x{3E} }`

U+16C9 RUNIC LETTER ALGIZ EOLHX

`{ \x{16C9} } { \cL\x{3F} }`

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

`{ \x{16CB} } { \cL\x{40} }`

U+16CF RUNIC LETTER TIWAZ TIR TYR T

`{ \x{16CF} } { \cL\x{41} }`

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

`{ \x{16D2} } { \cL\x{42} }`

U+16D6 RUNIC LETTER EHWAZ EH E

`{ \x{16D6} } { \cL\x{43} }`

U+16D7 RUNIC LETTER MANNAZ MAN M

`{ \x{16D7} } { \cL\x{44} }`

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L

`{ \x{16DA} } { \cL\x{45} }`

U+16DD RUNIC LETTER ING

`{ \x{16DD} } { \cL\x{46} }`

U+16DF RUNIC LETTER OTHALAN ETHEL O

`{ \x{16DF} } { \cL\x{47} }`

U+16DE RUNIC LETTER DAGAZ DAEG D

`{ \x{16DE} } { \cL\x{48} }`

U+16AA RUNIC LETTER AC A

`{ \x{16AA} } { \cL\x{49} }`

U+16AB RUNIC LETTER AESC

`{ \x{16AB} } { \cL\x{4A} }`

U+16A3 RUNIC LETTER YR

`{ \x{16A3} } { \cL\x{4B} }`

U+16E1 RUNIC LETTER IOR

`{ \x{16E1} } { \cL\x{4C} }`

U+16E0 RUNIC LETTER EAR

`{ \x{16E0} } { \cL\x{4D} }`

The remainder are in Unicode order, except for the punctuation. Some of these may need to be merged into the above, as applicable.

U+16A1 RUNIC LETTER V

`{ \x{16A1} } { \cL\x{4E} }`

U+16A4 RUNIC LETTER Y

`{ \x{16A4} } { \cL\x{4F} }`

U+16A5 RUNIC LETTER W

`{ \x{16A5} } { \cL\x{50} }`

U+16A7 RUNIC LETTER ETH

`{ \x{16A7} } { \cL\x{51} }`

U+16A8 RUNIC LETTER ANSUZ A

`{ \x{16A8} } { \cL\x{52} }`

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

`{ \x{16AC} } { \cL\x{53} }`

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

`{ \x{16AD} } { \cL\x{54} }`

U+16AE RUNIC LETTER O

`{ \x{16AE} } { \cL\x{55} }`

U+16AF RUNIC LETTER OE

`{ \x{16AF} } { \cL\x{56} }`

U+16B0 RUNIC LETTER ON

`{ \x{16B0} } { \cL\x{57} }`

U+16B2 RUNIC LETTER KAUNA

`{ \x{16B2} } { \cL\x{58} }`

U+16B4 RUNIC LETTER KAUN K

`{ \x{16B4} } { \cL\x{59} }`

U+16B5 RUNIC LETTER G

`{ \x{16B5} } { \cL\x{5A} }`

U+16B6 RUNIC LETTER ENG

`{ \x{16B6} } { \cL\x{5B} }`

U+16B8 RUNIC LETTER GAR

`{ \x{16B8} } { \cL\x{5C} }`

U+16BA RUNIC LETTER HAGLAZ H

`{ \x{16BA} } { \cL\x{5D} }`

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H

`{ \x{16BC} } { \cL\x{5E} }`

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H

`{ \x{16BD} } { \cL\x{5F} }`

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N

`{ \x{16BF} } { \cL\x{60} }`

U+16C0 RUNIC LETTER DOTTED-N

`{ \x{16C0} } { \cL\x{61} }`

U+16C2 RUNIC LETTER E

`{ \x{16C2} } { \cL\x{62} }`

U+16C3 RUNIC LETTER JERAN J

`{ \x{16C3} } { \cL\x{63} }`

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE

`{ \x{16C5} } { \cL\x{64} }`

U+16C6 RUNIC LETTER SHORT-TWIG-AR A

`{ \x{16C6} } { \cL\x{65} }`

U+16CA RUNIC LETTER SOWILO S

`{ \x{16CA} } { \cL\x{66} }`

U+16CC RUNIC LETTER SHORT-TWIG-SOL S

`{ \x{16CC} } { \cL\x{67} }`

U+16CD RUNIC LETTER C

`{ \x{16CD} } { \cL\x{68} }`

U+16CE RUNIC LETTER Z

`{ \x{16CE} } { \cL\x{69} }`

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T

`{ \x{16D0} } { \cL\x{6A} }`

U+16D1 RUNIC LETTER D

`{ \x{16D1} } { \cL\x{6B} }`

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B

`{ \x{16D3} } { \cL\x{6C} }`

U+16D4 RUNIC LETTER DOTTED-P

`{ \x{16D4} } { \cL\x{6D} }`

U+16D5 RUNIC LETTER OPEN-P

`{ \x{16D5} } { \cL\x{6E} }`

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M

`{ \x{16D8} } { \cL\x{6F} }`

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M

`{ \x{16D9} } { \cL\x{70} }`

U+16DB RUNIC LETTER DOTTED-L

`{ \x{16DB} } { \cL\x{71} }`

U+16DC RUNIC LETTER INGWAZ

`{ \x{16DC} } { \cL\x{72} }`

U+16E2 RUNIC LETTER CWEORTH

`{ \x{16E2} } { \cL\x{73} }`

U+16E3 RUNIC LETTER CALC

`{ \x{16E3} } { \cL\x{74} }`

U+16E4 RUNIC LETTER CEALC

`{ \x{16E4} } { \cL\x{75} }`

U+16E5 RUNIC LETTER STAN

`{ \x{16E5} } { \cL\x{76} }`

U+16E6 RUNIC LETTER LONG-BRANCH-YR

`{ \x{16E6} } { \cL\x{77} }`

U+16E7 RUNIC LETTER SHORT-TWIG-YR

`{ \x{16E7} } { \cL\x{78} }`

U+16E8 RUNIC LETTER ICELANDIC-YR

`{ \x{16E8} } { \cL\x{79} }`

U+16E9 RUNIC LETTER Q

`{ \x{16E9} } { \cL\x{7A} }`

U+16EA RUNIC LETTER X

`{ \x{16EA} } { \cL\x{7B} }`

U+16EE RUNIC ARLAUG SYMBOL

`{ \x{16EE} } { \cL\x{7C} \cO\! }`

U+16EF RUNIC TVIMADUR SYMBOL

`{ \x{16EF} } { \cL\x{7C} \cO\" }`

U+16F0 RUNIC BELGTHOR SYMBOL

`{ \x{16F0} } { \cL\x{7D} \cO\# }`

U+16F1 RUNIC LETTER K

`{ \x{16F1} } { \cL\x{7D} \cO\$ }`

U+16F2 RUNIC LETTER SH

`{ \x{16F2} } { \cL\x{7D} \cO\% }`

U+16F3 RUNIC LETTER OO

```
        { \x{16F3} } { \cL\x{7D} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
        { \x{16F4} } { \cL\x{7D} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
        { \x{16F5} } { \cL\x{7D} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
        { \x{16F6} } { \cL\x{7D} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
        { \x{16F7} } { \cL\x{7D} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
        { \x{16F8} } { \cL\x{7D} \cO\+ }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
        { \x{16EB} } { \cO\! \cO\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
        { \x{16EC} } { \cO\! \cO\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
        { \x{16ED} } { \cO\! \cO\# }
```

Tironian et

```
        { \x{204A} } { \cL\x{7E} }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
        { \cO([[:punct:]]) } { \cO\! \cO\$ \1 }
      }
      #1
     }
```

Old English Rune Poem order (fuþorc, punctuation last). Not all characters are included in the poem, so they are appended at the end.

```
   \cs_new:Nn \datatool_angRunr_order_futhorc_punc:n
    {
     \regex_replace_case_all:nN
     {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

```
        { \x{16A0} } { \cL\x{31} }
```

U+16A2 RUNIC LETTER URUZ UR U

```
        { \x{16A2} } { \cL\x{32} }
```

U+16A6 RUNIC LETTER THURISAZ THURS THORN

```
        { \x{16A6} } { \cL\x{33} }
```

U+16A9 RUNIC LETTER OS O

```
        { \x{16A9} } { \cL\x{34} }
```

U+16B1 RUNIC LETTER RAIDO RAD REID R

```
        { \x{16B1} } { \cL\x{35} }
```

U+16B3 RUNIC LETTER CEN

`{ \x{16B3} } { \cL\x{36} }`

U+16B7 RUNIC LETTER GEBO GYFU G

`{ \x{16B7} } { \cL\x{37} }`

U+16B9 RUNIC LETTER WUNJO WYNN W

`{ \x{16B9} } { \cL\x{38} }`

U+16BB RUNIC LETTER HAEGL H

`{ \x{16BB} } { \cL\x{39} }`

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

`{ \x{16BE} } { \cL\x{3A} }`

U+16C1 RUNIC LETTER ISAZ IS ISS I

`{ \x{16C1} } { \cL\x{3B} }`

U+16C4 RUNIC LETTER GER

`{ \x{16C4} } { \cL\x{3C} }`

U+16C7 RUNIC LETTER IWAZ EOH

`{ \x{16C7} } { \cL\x{3D} }`

U+16C8 RUNIC LETTER PERTHO PEORTH P

`{ \x{16C8} } { \cL\x{3E} }`

U+16C9 RUNIC LETTER ALGIZ EOLHX

`{ \x{16C9} } { \cL\x{3F} }`

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

`{ \x{16CB} } { \cL\x{40} }`

U+16CF RUNIC LETTER TIWAZ TIR TYR T

`{ \x{16CF} } { \cL\x{41} }`

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

`{ \x{16D2} } { \cL\x{42} }`

U+16D6 RUNIC LETTER EHWAZ EH E

`{ \x{16D6} } { \cL\x{43} }`

U+16D7 RUNIC LETTER MANNAZ MAN M

`{ \x{16D7} } { \cL\x{44} }`

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L

`{ \x{16DA} } { \cL\x{45} }`

U+16DD RUNIC LETTER ING

`{ \x{16DD} } { \cL\x{46} }`

U+16DF RUNIC LETTER OTHALAN ETHEL O

`{ \x{16DF} } { \cL\x{47} }`

U+16DE RUNIC LETTER DAGAZ DAEG D

`{ \x{16DE} } { \cL\x{48} }`

U+16AA RUNIC LETTER AC A

`{ \x{16AA} } { \cL\x{49} }`

U+16AB RUNIC LETTER AESC

```
{ \x{16AB} } { \cL\x{4A} }
```

U+16A3 RUNIC LETTER YR

```
{ \x{16A3} } { \cL\x{4B} }
```

U+16E1 RUNIC LETTER IOR

```
{ \x{16E1} } { \cL\x{4C} }
```

U+16E0 RUNIC LETTER EAR

```
{ \x{16E0} } { \cL\x{4D} }
```

The remainder are in Unicode order, except for the punctuation. Some of these may need to be merged into the above, as applicable.

U+16A1 RUNIC LETTER V

```
{ \x{16A1} } { \cL\x{4E} }
```

U+16A4 RUNIC LETTER Y

```
{ \x{16A4} } { \cL\x{4F} }
```

U+16A5 RUNIC LETTER W

```
{ \x{16A5} } { \cL\x{50} }
```

U+16A7 RUNIC LETTER ETH

```
{ \x{16A7} } { \cL\x{51} }
```

U+16A8 RUNIC LETTER ANSUZ A

```
{ \x{16A8} } { \cL\x{52} }
```

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

```
{ \x{16AC} } { \cL\x{53} }
```

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

```
{ \x{16AD} } { \cL\x{54} }
```

U+16AE RUNIC LETTER O

```
{ \x{16AE} } { \cL\x{55} }
```

U+16AF RUNIC LETTER OE

```
{ \x{16AF} } { \cL\x{56} }
```

U+16B0 RUNIC LETTER ON

```
{ \x{16B0} } { \cL\x{57} }
```

U+16B2 RUNIC LETTER KAUNA

```
{ \x{16B2} } { \cL\x{58} }
```

U+16B4 RUNIC LETTER KAUN K

```
{ \x{16B4} } { \cL\x{59} }
```

U+16B5 RUNIC LETTER G

```
{ \x{16B5} } { \cL\x{5A} }
```

U+16B6 RUNIC LETTER ENG

```
{ \x{16B6} } { \cL\x{5B} }
```

U+16B8 RUNIC LETTER GAR

```
{ \x{16B8} } { \cL\x{5C} }
```

U+16BA RUNIC LETTER HAGLAZ H

```
{ \x{16BA} } { \cL\x{5D} }
```

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H
`{ \x{16BC} } { \cL\x{5E} }`

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H
`{ \x{16BD} } { \cL\x{5F} }`

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N
`{ \x{16BF} } { \cL\x{60} }`

U+16C0 RUNIC LETTER DOTTED-N
`{ \x{16C0} } { \cL\x{61} }`

U+16C2 RUNIC LETTER E
`{ \x{16C2} } { \cL\x{62} }`

U+16C3 RUNIC LETTER JERAN J
`{ \x{16C3} } { \cL\x{63} }`

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE
`{ \x{16C5} } { \cL\x{64} }`

U+16C6 RUNIC LETTER SHORT-TWIG-AR A
`{ \x{16C6} } { \cL\x{65} }`

U+16CA RUNIC LETTER SOWILO S
`{ \x{16CA} } { \cL\x{66} }`

U+16CC RUNIC LETTER SHORT-TWIG-SOL S
`{ \x{16CC} } { \cL\x{67} }`

U+16CD RUNIC LETTER C
`{ \x{16CD} } { \cL\x{68} }`

U+16CE RUNIC LETTER Z
`{ \x{16CE} } { \cL\x{69} }`

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T
`{ \x{16D0} } { \cL\x{6A} }`

U+16D1 RUNIC LETTER D
`{ \x{16D1} } { \cL\x{6B} }`

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B
`{ \x{16D3} } { \cL\x{6C} }`

U+16D4 RUNIC LETTER DOTTED-P
`{ \x{16D4} } { \cL\x{6D} }`

U+16D5 RUNIC LETTER OPEN-P
`{ \x{16D5} } { \cL\x{6E} }`

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M
`{ \x{16D8} } { \cL\x{6F} }`

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M
`{ \x{16D9} } { \cL\x{70} }`

U+16DB RUNIC LETTER DOTTED-L
`{ \x{16DB} } { \cL\x{71} }`

U+16DC RUNIC LETTER INGWAZ

```
{ \x{16DC} } { \cL\x{72} }
```

U+16E2 RUNIC LETTER CWEORTH

```
{ \x{16E2} } { \cL\x{73} }
```

U+16E3 RUNIC LETTER CALC

```
{ \x{16E3} } { \cL\x{74} }
```

U+16E4 RUNIC LETTER CEALC

```
{ \x{16E4} } { \cL\x{75} }
```

U+16E5 RUNIC LETTER STAN

```
{ \x{16E5} } { \cL\x{76} }
```

U+16E6 RUNIC LETTER LONG-BRANCH-YR

```
{ \x{16E6} } { \cL\x{77} }
```

U+16E7 RUNIC LETTER SHORT-TWIG-YR

```
{ \x{16E7} } { \cL\x{78} }
```

U+16E8 RUNIC LETTER ICELANDIC-YR

```
{ \x{16E8} } { \cL\x{79} }
```

U+16E9 RUNIC LETTER Q

```
{ \x{16E9} } { \cL\x{7A} }
```

U+16EA RUNIC LETTER X

```
{ \x{16EA} } { \cL\x{7B} }
```

U+16EE RUNIC ARLAUG SYMBOL

```
{ \x{16EE} } { \cL\x{7C} \cO\! }
```

U+16EF RUNIC TVIMADUR SYMBOL

```
{ \x{16EF} } { \cL\x{7C} \cO\" }
```

U+16F0 RUNIC BELGTHOR SYMBOL

```
{ \x{16F0} } { \cL\x{7D} \cO\# }
```

U+16F1 RUNIC LETTER K

```
{ \x{16F1} } { \cL\x{7D} \cO\$ }
```

U+16F2 RUNIC LETTER SH

```
{ \x{16F2} } { \cL\x{7D} \cO\% }
```

U+16F3 RUNIC LETTER OO

```
{ \x{16F3} } { \cL\x{7D} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
{ \x{16F4} } { \cL\x{7D} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
{ \x{16F5} } { \cL\x{7D} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
{ \x{16F6} } { \cL\x{7D} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
{ \x{16F7} } { \cL\x{7D} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
     { \x{16F8} } { \cL\x{7D} \cO\+ }
```

Tironian et

```
     { \x{204A} } { \cL\x{7D} \cO\, }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
     { \x{16EB} } { \cO\x{7E} \cO\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
     { \x{16EC} } { \cO\x{7E} \cO\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
     { \x{16ED} } { \cO\x{7E} \cO\# }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
     { \cO([[:punct:]]) } { \cO\x{7E} \cO\$ \1 }
   }
   #1
  }
}
{
```

Unicode order (mostly Unicode order). The punctuation characters are placed first, the rest follows Unicode order.

```
 \cs_new:Nn \datatool_angRunr_order_punc_unicode:n
  {
   \regex_replace_case_all:nN
   {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

```
     { \x{E1} \x{9A} \x{A0} }
       { \cL\x{31} }
```

U+16A1 RUNIC LETTER V

```
     { \x{E1} \x{9A} \x{A1} }
       { \cL\x{32} }
```

U+16A2 RUNIC LETTER URUZ UR U

```
     { \x{E1} \x{9A} \x{A2} }
       { \cL\x{33} }
```

U+16A3 RUNIC LETTER YR

```
     { \x{E1} \x{9A} \x{A3} }
       { \cL\x{34} }
```

U+16A4 RUNIC LETTER Y

```
     { \x{E1} \x{9A} \x{A4} }
       { \cL\x{35} }
```

U+16A5 RUNIC LETTER W

```
     { \x{E1} \x{9A} \x{A5} }
       { \cL\x{36} }
```

U+16A6 RUNIC LETTER THURISAZ THURS THORN

```
     { \x{E1} \x{9A} \x{A6} }
       { \cL\x{37} }
```

U+16A7 RUNIC LETTER ETH

```
{ \x{E1} \x{9A} \x{A7} }
 { \cL\x{38} }
```

U+16A8 RUNIC LETTER ANSUZ A

```
{ \x{E1} \x{9A} \x{A8} }
 { \cL\x{39} }
```

U+16A9 RUNIC LETTER OS O

```
{ \x{E1} \x{9A} \x{A9} }
 { \cL\: }
```

U+16AA RUNIC LETTER AC A

```
{ \x{E1} \x{9A} \x{AA} }
 { \cL\; }
```

U+16AB RUNIC LETTER AESC

```
{ \x{E1} \x{9A} \x{AB} }
 { \cL\< }
```

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

```
{ \x{E1} \x{9A} \x{AC} }
 { \cL\= }
```

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

```
{ \x{E1} \x{9A} \x{AD} }
 { \cL\> }
```

U+16AE RUNIC LETTER O

```
{ \x{E1} \x{9A} \x{AE} }
 { \cL\? }
```

U+16AF RUNIC LETTER OE

```
{ \x{E1} \x{9A} \x{AF} }
 { \cL\@ }
```

U+16B0 RUNIC LETTER ON

```
{ \x{E1} \x{9A} \x{B0} }
 { \cL\x{41} }
```

U+16B1 RUNIC LETTER RAIDO RAD REID R

```
{ \x{E1} \x{9A} \x{B1} }
 { \cL\x{42} }
```

U+16B2 RUNIC LETTER KAUNA

```
{ \x{E1} \x{9A} \x{B2} }
 { \cL\x{43} }
```

U+16B3 RUNIC LETTER CEN

```
{ \x{E1} \x{9A} \x{B3} }
 { \cL\x{44} }
```

U+16B4 RUNIC LETTER KAUN K

```
{ \x{E1} \x{9A} \x{B4} }
 { \cL\x{45} }
```

U+16B5 RUNIC LETTER G

```
{ \x{E1} \x{9A} \x{B5} }
 { \cL\x{46} }
```

U+16B6 RUNIC LETTER ENG

```
{ \x{E1} \x{9A} \x{B6} }
 { \cL\x{47} }
```

U+16B7 RUNIC LETTER GEBO GYFU G

```
{ \x{E1} \x{9A} \x{B7} }
 { \cL\x{48} }
```

U+16B8 RUNIC LETTER GAR

```
{ \x{E1} \x{9A} \x{B8} }
 { \cL\x{49} }
```

U+16B9 RUNIC LETTER WUNJO WYNN W

```
{ \x{E1} \x{9A} \x{B9} }
 { \cL\x{4A} }
```

U+16BA RUNIC LETTER HAGLAZ H

```
{ \x{E1} \x{9A} \x{BA} }
 { \cL\x{4B} }
```

U+16BB RUNIC LETTER HAEGL H

```
{ \x{E1} \x{9A} \x{BB} }
 { \cL\x{4C} }
```

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H

```
{ \x{E1} \x{9A} \x{BC} }
 { \cL\x{4D} }
```

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H

```
{ \x{E1} \x{9A} \x{BD} }
 { \cL\x{4E} }
```

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

```
{ \x{E1} \x{9A} \x{BE} }
 { \cL\x{4F} }
```

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N

```
{ \x{E1} \x{9A} \x{BF} }
 { \cL\x{50} }
```

U+16C0 RUNIC LETTER DOTTED-N

```
{ \x{E1} \x{9B} \x{80} }
 { \cL\x{51} }
```

U+16C1 RUNIC LETTER ISAZ IS ISS I

```
{ \x{E1} \x{9B} \x{81} }
 { \cL\x{52} }
```

U+16C2 RUNIC LETTER E

```
{ \x{E1} \x{9B} \x{82} }
 { \cL\x{53} }
```

U+16C3 RUNIC LETTER JERAN J

```
{ \x{E1} \x{9B} \x{83} }
 { \cL\x{54} }
```

U+16C4 RUNIC LETTER GER

```
{ \x{E1} \x{9B} \x{84} }
 { \cL\x{55} }
```

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE

```
{ \x{E1} \x{9B} \x{85} }
 { \cL\x{56} }
```

U+16C6 RUNIC LETTER SHORT-TWIG-AR A

```
{ \x{E1} \x{9B} \x{86} }
 { \cL\x{57} }
```

U+16C7 RUNIC LETTER IWAZ EOH

```
{ \x{E1} \x{9B} \x{87} }
 { \cL\x{58} }
```

U+16C8 RUNIC LETTER PERTHO PEORTH P

```
{ \x{E1} \x{9B} \x{88} }
 { \cL\x{59} }
```

U+16C9 RUNIC LETTER ALGIZ EOLHX

```
{ \x{E1} \x{9B} \x{89} }
 { \cL\x{5A} }
```

U+16CA RUNIC LETTER SOWILO S

```
{ \x{E1} \x{9B} \x{8A} }
 { \cL\[ }
```

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

```
{ \x{E1} \x{9B} \x{8B} }
 { \cL\\ }
```

U+16CC RUNIC LETTER SHORT-TWIG-SOL S

```
{ \x{E1} \x{9B} \x{8C} }
 { \cL\] }
```

U+16CD RUNIC LETTER C

```
{ \x{E1} \x{9B} \x{8D} }
 { \cL\^ }
```

U+16CE RUNIC LETTER Z

```
{ \x{E1} \x{9B} \x{8E} }
 { \cL\_ }
```

U+16CF RUNIC LETTER TIWAZ TIR TYR T

```
{ \x{E1} \x{9B} \x{8F} }
 { \cL\` }
```

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T

```
{ \x{E1} \x{9B} \x{90} }
 { \cL\x{61} }
```

U+16D1 RUNIC LETTER D

```
{ \x{E1} \x{9B} \x{91} }
 { \cL\x{62} }
```

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

```
{ \x{E1} \x{9B} \x{92} }
 { \cL\x{63} }
```

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B

```
{ \x{E1} \x{9B} \x{93} }
 { \cL\x{64} }
```

U+16D4 RUNIC LETTER DOTTED-P

```
{ \x{E1} \x{9B} \x{94} }
 { \cL\x{65} }
```

U+16D5 RUNIC LETTER OPEN-P

```
{ \x{E1} \x{9B} \x{95} }
 { \cL\x{66} }
```

U+16D6 RUNIC LETTER EHWAZ EH E

```
{ \x{E1} \x{9B} \x{96} }
 { \cL\x{67} }
```

U+16D7 RUNIC LETTER MANNAZ MAN M

```
{ \x{E1} \x{9B} \x{97} }
 { \cL\x{68} }
```

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M

```
{ \x{E1} \x{9B} \x{98} }
 { \cL\x{69} }
```

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M

```
{ \x{E1} \x{9B} \x{99} }
 { \cL\x{6A} }
```

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L

```
{ \x{E1} \x{9B} \x{9A} }
 { \cL\x{6B} }
```

U+16DB RUNIC LETTER DOTTED-L

```
{ \x{E1} \x{9B} \x{9B} }
 { \cL\x{6C} }
```

U+16DC RUNIC LETTER INGWAZ

```
{ \x{E1} \x{9B} \x{9C} }
 { \cL\x{6D} }
```

U+16DD RUNIC LETTER ING

```
{ \x{E1} \x{9B} \x{9D} }
 { \cL\x{6F} }
```

U+16DE RUNIC LETTER DAGAZ DAEG D

```
{ \x{E1} \x{9B} \x{9E} }
 { \cL\x{70} }
```

U+16DF RUNIC LETTER OTHALAN ETHEL O

```
{ \x{E1} \x{9B} \x{9F} }
 { \cL\x{71} }
```

U+16E0 RUNIC LETTER EAR

```
{ \x{E1} \x{9B} \x{A0} }
 { \cL\x{72} }
```

U+16E1 RUNIC LETTER IOR

```
{ \x{E1} \x{9B} \x{A1} }
 { \cL\x{73} }
```

U+16E2 RUNIC LETTER CWEORTH

```
{ \x{E1} \x{9B} \x{A2} }
 { \cL\x{74} }
```

U+16E3 RUNIC LETTER CALC

```
{ \x{E1} \x{9B} \x{A3} }
 { \cL\x{75} }
```

U+16E4 RUNIC LETTER CEALC

```
{ \x{E1} \x{9B} \x{A4} }
 { \cL\x{76} }
```

U+16E5 RUNIC LETTER STAN

```
{ \x{E1} \x{9B} \x{A5} }
 { \cL\x{77} }
```

U+16E6 RUNIC LETTER LONG-BRANCH-YR

```
{ \x{E1} \x{9B} \x{A6} }
 { \cL\x{78} }
```

U+16E7 RUNIC LETTER SHORT-TWIG-YR

```
{ \x{E1} \x{9B} \x{A7} }
 { \cL\x{79} }
```

U+16E8 RUNIC LETTER ICELANDIC-YR

```
{ \x{E1} \x{9B} \x{A8} }
 { \cL\x{7A} }
```

U+16E9 RUNIC LETTER Q

```
{ \x{E1} \x{9B} \x{A9} }
 { \cL\{ }
```

U+16EA RUNIC LETTER X

```
{ \x{E1} \x{9B} \x{AA} }
 { \cL\| }
```

U+16EE RUNIC ARLAUG SYMBOL

```
{ \x{E1} \x{9B} \x{AE} }
 { \cL\} \cO\! }
```

U+16EF RUNIC TVIMADUR SYMBOL

```
{ \x{E1} \x{9B} \x{AF} }
 { \cL\} \cO\" }
```

U+16F0 RUNIC BELGTHOR SYMBOL

```
{ \x{E1} \x{9B} \x{B0} }
 { \cL\} \cO\# }
```

U+16F1 RUNIC LETTER K

```
{ \x{E1} \x{9B} \x{B1} }
 { \cL\} \cO\$ }
```

U+16F2 RUNIC LETTER SH

```
{ \x{E1} \x{9B} \x{B2} }
 { \cL\} \cO\% }
```

U+16F3 RUNIC LETTER OO

```
{ \x{E1} \x{9B} \x{B3} }
 { \cL\} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
{ \x{E1} \x{9B} \x{B4} }
 { \cL\} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
       { \x{E1} \x{9B} \x{B5} }
        { \cL\} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
       { \x{E1} \x{9B} \x{B6} }
        { \cL\} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
       { \x{E1} \x{9B} \x{B7} }
        { \cL\} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
       { \x{E1} \x{9B} \x{B8} }
        { \cL\} \cO\+ }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
       { \x{E1} \x{9B} \x{AB} }
        { \cO\! \cO\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
       { \x{E1} \x{9B} \x{AC} }
        { \cO\! \cO\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
       { \x{E1} \x{9B} \x{AD} }
        { \cO\! \cO\# }
```

Tironian et

```
       { \x{E2} \x{81} \x{8A} }
         { \cL\~ }
```

Any character in the punctuation class that has category other: Version 1.1 added category code check.

```
       { \cO([[:punct:]]) } { \cO\! \cO\$ \1 }
     }
     #1
    }
```

Old English Rune Poem order (fuþorc). Not all characters are included in the poem, so they are appended at the end. Punctuation characters come first.

```
   \cs_new:Nn \datatool_angRunr_order_punc_futhorc:n
    {
     \regex_replace_case_all:nN
     {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

```
       { \x{E1} \x{9A} \x{A0} }
         { \cL\x{31} }
```

U+16A2 RUNIC LETTER URUZ UR U

```
       { \x{E1} \x{9A} \x{A2} }
         { \cL\x{32} }
```

U+16A6 RUNIC LETTER THURISAZ THURS THORN

```
       { \x{E1} \x{9A} \x{A6} }
         { \cL\x{33} }
```

U+16A9 RUNIC LETTER OS O

```
{ \x{E1} \x{9A} \x{A9} }
  { \cL\x{34} }
```

U+16B1 RUNIC LETTER RAIDO RAD REID R

```
{ \x{E1} \x{9A} \x{B1} }
  { \cL\x{35} }
```

U+16B3 RUNIC LETTER CEN

```
{ \x{E1} \x{9A} \x{B3} }
 { \cL\x{36} }
```

U+16B7 RUNIC LETTER GEBO GYFU G

```
{ \x{E1} \x{9A} \x{B7} }
 { \cL\x{37} }
```

U+16B9 RUNIC LETTER WUNJO WYNN W

```
{ \x{E1} \x{9A} \x{B9} }
 { \cL\x{38} }
```

U+16BB RUNIC LETTER HAEGL H

```
{ \x{E1} \x{9A} \x{BB} }
 { \cL\x{39} }
```

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

```
{ \x{E1} \x{9A} \x{BE} }
 { \cL\: }
```

U+16C1 RUNIC LETTER ISAZ IS ISS I

```
{ \x{E1} \x{9B} \x{81} }
 { \cL\; }
```

U+16C4 RUNIC LETTER GER

```
{ \x{E1} \x{9B} \x{84} }
 { \cL\< }
```

U+16C7 RUNIC LETTER IWAZ EOH

```
{ \x{E1} \x{9B} \x{87} }
 { \cL\= }
```

U+16C8 RUNIC LETTER PERTHO PEORTH P

```
{ \x{E1} \x{9B} \x{88} }
 { \cL\> }
```

U+16C9 RUNIC LETTER ALGIZ EOLHX

```
{ \x{E1} \x{9B} \x{89} }
 { \cL\? }
```

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

```
{ \x{E1} \x{9B} \x{8B} }
 { \cL\@ }
```

U+16CF RUNIC LETTER TIWAZ TIR TYR T

```
{ \x{E1} \x{9B} \x{8F} }
 { \cL\x{41} }
```

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

```
{ \x{E1} \x{9B} \x{92} }
 { \cL\x{42} }
```

U+16D6 RUNIC LETTER EHWAZ EH E

```
{ \x{E1} \x{9B} \x{96} }
 { \cL\x{43} }
```

U+16D7 RUNIC LETTER MANNAZ MAN M

```
{ \x{E1} \x{9B} \x{97} }
 { \cL\x{44} }
```

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L

```
{ \x{E1} \x{9B} \x{9A} }
 { \cL\x{45} }
```

U+16DD RUNIC LETTER ING

```
{ \x{E1} \x{9B} \x{9D} }
 { \cL\x{46} }
```

U+16DF RUNIC LETTER OTHALAN ETHEL O

```
{ \x{E1} \x{9B} \x{9F} }
 { \cL\x{47} }
```

U+16DE RUNIC LETTER DAGAZ DAEG D

```
{ \x{E1} \x{9B} \x{9E} }
 { \cL\x{48} }
```

U+16AA RUNIC LETTER AC A

```
{ \x{E1} \x{9A} \x{AA} }
  { \cL\x{49} }
```

U+16AB RUNIC LETTER AESC

```
{ \x{E1} \x{9A} \x{AB} }
  { \cL\x{4A} }
```

U+16A3 RUNIC LETTER YR

```
{ \x{E1} \x{9A} \x{A3} }
  { \cL\x{4B} }
```

U+16E1 RUNIC LETTER IOR

```
{ \x{E1} \x{9B} \x{A1} }
 { \cL\x{4C} }
```

U+16E0 RUNIC LETTER EAR

```
{ \x{E1} \x{9B} \x{A0} }
 { \cL\x{4D} }
```

The remainder are in Unicode order, except for the punctuation. Some of these may need to be merged into the above, as applicable.

U+16A1 RUNIC LETTER V

```
{ \x{E1} \x{9A} \x{A1} }
  { \cL\x{4E} }
```

U+16A4 RUNIC LETTER Y

```
{ \x{E1} \x{9A} \x{A4} }
  { \cL\x{4F} }
```

U+16A5 RUNIC LETTER W

```
{ \x{E1} \x{9A} \x{A5} }
  { \cL\x{50} }
```

U+16A7 RUNIC LETTER ETH

```
{ \x{E1} \x{9A} \x{A7} }
  { \cL\x{51} }
```

U+16A8 RUNIC LETTER ANSUZ A

```
{ \x{E1} \x{9A} \x{A8} }
  { \cL\x{52} }
```

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

```
{ \x{E1} \x{9A} \x{AC} }
  { \cL\x{53} }
```

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

```
{ \x{E1} \x{9A} \x{AD} }
  { \cL\x{54} }
```

U+16AE RUNIC LETTER O

```
{ \x{E1} \x{9A} \x{AE} }
  { \cL\x{55} }
```

U+16AF RUNIC LETTER OE

```
{ \x{E1} \x{9A} \x{AF} }
  { \cL\x{56} }
```

U+16B0 RUNIC LETTER ON

```
{ \x{E1} \x{9A} \x{B0} }
  { \cL\x{57} }
```

U+16B2 RUNIC LETTER KAUNA

```
{ \x{E1} \x{9A} \x{B2} }
 { \cL\x{58} }
```

U+16B4 RUNIC LETTER KAUN K

```
{ \x{E1} \x{9A} \x{B4} }
 { \cL\x{59} }
```

U+16B5 RUNIC LETTER G

```
{ \x{E1} \x{9A} \x{B5} }
 { \cL\x{5A} }
```

U+16B6 RUNIC LETTER ENG

```
{ \x{E1} \x{9A} \x{B6} }
 { \cL\[ }
```

U+16B8 RUNIC LETTER GAR

```
{ \x{E1} \x{9A} \x{B8} }
 { \cL\\ }
```

U+16BA RUNIC LETTER HAGLAZ H

```
{ \x{E1} \x{9A} \x{BA} }
 { \cL\] }
```

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H

```
{ \x{E1} \x{9A} \x{BC} }
 { \cL\^ }
```

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H

```
{ \x{E1} \x{9A} \x{BD} }
 { \cL\_ }
```

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N

```
{ \x{E1} \x{9A} \x{BF} }
 { \cL\` }
```

U+16C0 RUNIC LETTER DOTTED-N

```
{ \x{E1} \x{9B} \x{80} }
 { \cL\x{61} }
```

U+16C2 RUNIC LETTER E

```
{ \x{E1} \x{9B} \x{82} }
 { \cL\x{62} }
```

U+16C3 RUNIC LETTER JERAN J

```
{ \x{E1} \x{9B} \x{83} }
 { \cL\x{63} }
```

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE

```
{ \x{E1} \x{9B} \x{85} }
 { \cL\x{64} }
```

U+16C6 RUNIC LETTER SHORT-TWIG-AR A

```
{ \x{E1} \x{9B} \x{86} }
 { \cL\x{65} }
```

U+16CA RUNIC LETTER SOWILO S

```
{ \x{E1} \x{9B} \x{8A} }
 { \cL\x{66} }
```

U+16CC RUNIC LETTER SHORT-TWIG-SOL S

```
{ \x{E1} \x{9B} \x{8C} }
 { \cL\x{67} }
```

U+16CD RUNIC LETTER C

```
{ \x{E1} \x{9B} \x{8D} }
 { \cL\x{68} }
```

U+16CE RUNIC LETTER Z

```
{ \x{E1} \x{9B} \x{8E} }
 { \cL\x{69} }
```

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T

```
{ \x{E1} \x{9B} \x{90} }
 { \cL\x{6A} }
```

U+16D1 RUNIC LETTER D

```
{ \x{E1} \x{9B} \x{91} }
 { \cL\x{6B} }
```

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B

```
{ \x{E1} \x{9B} \x{93} }
 { \cL\x{6C} }
```

U+16D4 RUNIC LETTER DOTTED-P

```
{ \x{E1} \x{9B} \x{94} }
 { \cL\x{6D} }
```

U+16D5 RUNIC LETTER OPEN-P

```
{ \x{E1} \x{9B} \x{95} }
 { \cL\x{6E} }
```

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M

```
        { \x{E1} \x{9B} \x{98} }
         { \cL\x{6F} }
```

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M

```
        { \x{E1} \x{9B} \x{99} }
         { \cL\x{70} }
```

U+16DB RUNIC LETTER DOTTED-L

```
        { \x{E1} \x{9B} \x{9B} }
         { \cL\x{71} }
```

U+16DC RUNIC LETTER INGWAZ

```
        { \x{E1} \x{9B} \x{9C} }
         { \cL\x{72} }
```

U+16E2 RUNIC LETTER CWEORTH

```
        { \x{E1} \x{9B} \x{A2} }
         { \cL\x{73} }
```

U+16E3 RUNIC LETTER CALC

```
        { \x{E1} \x{9B} \x{A3} }
         { \cL\x{74} }
```

U+16E4 RUNIC LETTER CEALC

```
        { \x{E1} \x{9B} \x{A4} }
         { \cL\x{75} }
```

U+16E5 RUNIC LETTER STAN

```
        { \x{E1} \x{9B} \x{A5} }
         { \cL\x{76} }
```

U+16E6 RUNIC LETTER LONG-BRANCH-YR

```
        { \x{E1} \x{9B} \x{A6} }
         { \cL\x{77} }
```

U+16E7 RUNIC LETTER SHORT-TWIG-YR

```
        { \x{E1} \x{9B} \x{A7} }
         { \cL\x{78} }
```

U+16E8 RUNIC LETTER ICELANDIC-YR

```
        { \x{E1} \x{9B} \x{A8} }
         { \cL\x{79} }
```

U+16E9 RUNIC LETTER Q

```
        { \x{E1} \x{9B} \x{A9} }
         { \cL\x{7A} }
```

U+16EA RUNIC LETTER X

```
        { \x{E1} \x{9B} \x{AA} }
         { \cL\{ }
```

U+16EE RUNIC ARLAUG SYMBOL

```
        { \x{E1} \x{9B} \x{AE} }
         { \cL\| \cO\! }
```

U+16EF RUNIC TVIMADUR SYMBOL

```
        { \x{E1} \x{9B} \x{AF} }
         { \cL\} \cO\" }
```

U+16F0 RUNIC BELGTHOR SYMBOL

```
   { \x{E1} \x{9B} \x{B0} }
    { \cL\} \cO\# }
```

U+16F1 RUNIC LETTER K

```
   { \x{E1} \x{9B} \x{B1} }
    { \cL\} \cO\$ }
```

U+16F2 RUNIC LETTER SH

```
   { \x{E1} \x{9B} \x{B2} }
    { \cL\} \cO\% }
```

U+16F3 RUNIC LETTER OO

```
   { \x{E1} \x{9B} \x{B3} }
    { \cL\} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
   { \x{E1} \x{9B} \x{B4} }
    { \cL\} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
   { \x{E1} \x{9B} \x{B5} }
    { \cL\} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
   { \x{E1} \x{9B} \x{B6} }
    { \cL\} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
   { \x{E1} \x{9B} \x{B7} }
    { \cL\} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
   { \x{E1} \x{9B} \x{B8} }
    { \cL\} \cO\+ }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
   { \x{E1} \x{9B} \x{AB} }
    { \cO\! \cO\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
   { \x{E1} \x{9B} \x{AC} }
    { \cO\! \cO\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
   { \x{E1} \x{9B} \x{AD} }
    { \cO\! \cO\# }
```

Tironian et

```
   { \x{E2} \x{81} \x{8A} }
     { \cL\~ }
```

Any character in the punctuation class (unlikely to occur in Runic but added for completeness): Version 1.1 added category code check.

```
   { \cO([[:punct:]]) } { \cO\! \cO\$ \1 }
 }
 #1
}
```

Old English Rune Poem order (fuþorc, punctuation last). Not all characters are included in the poem, so they are appended at the end before the punctuation.

```
\cs_new:Nn \datatool_angRunr_order_futhorc_punc:n
 {
  \regex_replace_case_all:nN
  {
```

U+16A0 RUNIC LETTER FEHU FEOH FE F

```
    { \x{E1} \x{9A} \x{A0} }
      { \cL\x{31} }
```

U+16A2 RUNIC LETTER URUZ UR U

```
    { \x{E1} \x{9A} \x{A2} }
      { \cL\x{32} }
```

U+16A6 RUNIC LETTER THURISAZ THURS THORN

```
    { \x{E1} \x{9A} \x{A6} }
      { \cL\x{33} }
```

U+16A9 RUNIC LETTER OS O

```
    { \x{E1} \x{9A} \x{A9} }
      { \cL\x{34} }
```

U+16B1 RUNIC LETTER RAIDO RAD REID R

```
    { \x{E1} \x{9A} \x{B1} }
      { \cL\x{35} }
```

U+16B3 RUNIC LETTER CEN

```
    { \x{E1} \x{9A} \x{B3} }
     { \cL\x{36} }
```

U+16B7 RUNIC LETTER GEBO GYFU G

```
    { \x{E1} \x{9A} \x{B7} }
     { \cL\x{37} }
```

U+16B9 RUNIC LETTER WUNJO WYNN W

```
    { \x{E1} \x{9A} \x{B9} }
     { \cL\x{38} }
```

U+16BB RUNIC LETTER HAEGL H

```
    { \x{E1} \x{9A} \x{BB} }
     { \cL\x{39} }
```

U+16BE RUNIC LETTER NAUDIZ NYD NAUD N

```
    { \x{E1} \x{9A} \x{BE} }
     { \cL\: }
```

U+16C1 RUNIC LETTER ISAZ IS ISS I

```
    { \x{E1} \x{9B} \x{81} }
     { \cL\; }
```

U+16C4 RUNIC LETTER GER

```
    { \x{E1} \x{9B} \x{84} }
     { \cL\< }
```

U+16C7 RUNIC LETTER IWAZ EOH

```
    { \x{E1} \x{9B} \x{87} }
     { \cL\= }
```

U+16C8 RUNIC LETTER PERTHO PEORTH P

```
{ \x{E1} \x{9B} \x{88} }
 { \cL\> }
```

U+16C9 RUNIC LETTER ALGIZ EOLHX

```
{ \x{E1} \x{9B} \x{89} }
 { \cL\? }
```

U+16CB RUNIC LETTER SIGEL LONG-BRANCH-SOL S

```
{ \x{E1} \x{9B} \x{8B} }
 { \cL\@ }
```

U+16CF RUNIC LETTER TIWAZ TIR TYR T

```
{ \x{E1} \x{9B} \x{8F} }
 { \cL\x{41} }
```

U+16D2 RUNIC LETTER BERKANAN BEORC BJARKAN B

```
{ \x{E1} \x{9B} \x{92} }
 { \cL\x{42} }
```

U+16D6 RUNIC LETTER EHWAZ EH E

```
{ \x{E1} \x{9B} \x{96} }
 { \cL\x{43} }
```

U+16D7 RUNIC LETTER MANNAZ MAN M

```
{ \x{E1} \x{9B} \x{97} }
 { \cL\x{44} }
```

U+16DA RUNIC LETTER LAUKAZ LAGU LOGR L

```
{ \x{E1} \x{9B} \x{9A} }
 { \cL\x{45} }
```

U+16DD RUNIC LETTER ING

```
{ \x{E1} \x{9B} \x{9D} }
 { \cL\x{46} }
```

U+16DF RUNIC LETTER OTHALAN ETHEL O

```
{ \x{E1} \x{9B} \x{9F} }
 { \cL\x{47} }
```

U+16DE RUNIC LETTER DAGAZ DAEG D

```
{ \x{E1} \x{9B} \x{9E} }
 { \cL\x{48} }
```

U+16AA RUNIC LETTER AC A

```
{ \x{E1} \x{9A} \x{AA} }
  { \cL\x{49} }
```

U+16AB RUNIC LETTER AESC

```
{ \x{E1} \x{9A} \x{AB} }
  { \cL\x{4A} }
```

U+16A3 RUNIC LETTER YR

```
{ \x{E1} \x{9A} \x{A3} }
  { \cL\x{4B} }
```

U+16E1 RUNIC LETTER IOR

```
{ \x{E1} \x{9B} \x{A1} }
 { \cL\x{4C} }
```

U+16E0 RUNIC LETTER EAR

```
{ \x{E1} \x{9B} \x{A0} }
 { \cL\x{4D} }
```

The remainder are in Unicode order, except for the punctuation. Some of these may need to be merged into the above, as applicable.

U+16A1 RUNIC LETTER V

```
{ \x{E1} \x{9A} \x{A1} }
  { \cL\x{4E} }
```

U+16A4 RUNIC LETTER Y

```
{ \x{E1} \x{9A} \x{A4} }
  { \cL\x{4F} }
```

U+16A5 RUNIC LETTER W

```
{ \x{E1} \x{9A} \x{A5} }
  { \cL\x{50} }
```

U+16A7 RUNIC LETTER ETH

```
{ \x{E1} \x{9A} \x{A7} }
  { \cL\x{51} }
```

U+16A8 RUNIC LETTER ANSUZ A

```
{ \x{E1} \x{9A} \x{A8} }
  { \cL\x{52} }
```

U+16AC RUNIC LETTER LONG-BRANCH-OSS O

```
{ \x{E1} \x{9A} \x{AC} }
  { \cL\x{53} }
```

U+16AD RUNIC LETTER SHORT-TWIG-OSS O

```
{ \x{E1} \x{9A} \x{AD} }
  { \cL\x{54} }
```

U+16AE RUNIC LETTER O

```
{ \x{E1} \x{9A} \x{AE} }
  { \cL\x{55} }
```

U+16AF RUNIC LETTER OE

```
{ \x{E1} \x{9A} \x{AF} }
  { \cL\x{56} }
```

U+16B0 RUNIC LETTER ON

```
{ \x{E1} \x{9A} \x{B0} }
  { \cL\x{57} }
```

U+16B2 RUNIC LETTER KAUNA

```
{ \x{E1} \x{9A} \x{B2} }
 { \cL\x{58} }
```

U+16B4 RUNIC LETTER KAUN K

```
{ \x{E1} \x{9A} \x{B4} }
 { \cL\x{59} }
```

U+16B5 RUNIC LETTER G

```
{ \x{E1} \x{9A} \x{B5} }
 { \cL\x{5A} }
```

U+16B6 RUNIC LETTER ENG

```
{ \x{E1} \x{9A} \x{B6} }
 { \cL\[ }
```

U+16B8 RUNIC LETTER GAR

```
{ \x{E1} \x{9A} \x{B8} }
 { \cL\\ }
```

U+16BA RUNIC LETTER HAGLAZ H

```
{ \x{E1} \x{9A} \x{BA} }
 { \cL\] }
```

U+16BC RUNIC LETTER LONG-BRANCH-HAGALL H

```
{ \x{E1} \x{9A} \x{BC} }
 { \cL\^ }
```

U+16BD RUNIC LETTER SHORT-TWIG-HAGALL H

```
{ \x{E1} \x{9A} \x{BD} }
 { \cL\_ }
```

U+16BF RUNIC LETTER SHORT-TWIG-NAUD N

```
{ \x{E1} \x{9A} \x{BF} }
 { \cL\` }
```

U+16C0 RUNIC LETTER DOTTED-N

```
{ \x{E1} \x{9B} \x{80} }
 { \cL\x{61} }
```

U+16C2 RUNIC LETTER E

```
{ \x{E1} \x{9B} \x{82} }
 { \cL\x{62} }
```

U+16C3 RUNIC LETTER JERAN J

```
{ \x{E1} \x{9B} \x{83} }
 { \cL\x{63} }
```

U+16C5 RUNIC LETTER LONG-BRANCH-AR AE

```
{ \x{E1} \x{9B} \x{85} }
 { \cL\x{64} }
```

U+16C6 RUNIC LETTER SHORT-TWIG-AR A

```
{ \x{E1} \x{9B} \x{86} }
 { \cL\x{65} }
```

U+16CA RUNIC LETTER SOWILO S

```
{ \x{E1} \x{9B} \x{8A} }
 { \cL\x{66} }
```

U+16CC RUNIC LETTER SHORT-TWIG-SOL S

```
{ \x{E1} \x{9B} \x{8C} }
 { \cL\x{67} }
```

U+16CD RUNIC LETTER C

```
{ \x{E1} \x{9B} \x{8D} }
 { \cL\x{68} }
```

U+16CE RUNIC LETTER Z

```
{ \x{E1} \x{9B} \x{8E} }
 { \cL\x{69} }
```

U+16D0 RUNIC LETTER SHORT-TWIG-TYR T

```
{ \x{E1} \x{9B} \x{90} }
 { \cL\x{6A} }
```

U+16D1 RUNIC LETTER D

```
{ \x{E1} \x{9B} \x{91} }
 { \cL\x{6B} }
```

U+16D3 RUNIC LETTER SHORT-TWIG-BJARKAN B

```
{ \x{E1} \x{9B} \x{93} }
 { \cL\x{6C} }
```

U+16D4 RUNIC LETTER DOTTED-P

```
{ \x{E1} \x{9B} \x{94} }
 { \cL\x{6D} }
```

U+16D5 RUNIC LETTER OPEN-P

```
{ \x{E1} \x{9B} \x{95} }
 { \cL\x{6E} }
```

U+16D8 RUNIC LETTER LONG-BRANCH-MADR M

```
{ \x{E1} \x{9B} \x{98} }
 { \cL\x{6F} }
```

U+16D9 RUNIC LETTER SHORT-TWIG-MADR M

```
{ \x{E1} \x{9B} \x{99} }
 { \cL\x{70} }
```

U+16DB RUNIC LETTER DOTTED-L

```
{ \x{E1} \x{9B} \x{9B} }
 { \cL\x{71} }
```

U+16DC RUNIC LETTER INGWAZ

```
{ \x{E1} \x{9B} \x{9C} }
 { \cL\x{72} }
```

U+16E2 RUNIC LETTER CWEORTH

```
{ \x{E1} \x{9B} \x{A2} }
 { \cL\x{73} }
```

U+16E3 RUNIC LETTER CALC

```
{ \x{E1} \x{9B} \x{A3} }
 { \cL\x{74} }
```

U+16E4 RUNIC LETTER CEALC

```
{ \x{E1} \x{9B} \x{A4} }
 { \cL\x{75} }
```

U+16E5 RUNIC LETTER STAN

```
{ \x{E1} \x{9B} \x{A5} }
 { \cL\x{76} }
```

U+16E6 RUNIC LETTER LONG-BRANCH-YR

```
{ \x{E1} \x{9B} \x{A6} }
 { \cL\x{77} }
```

U+16E7 RUNIC LETTER SHORT-TWIG-YR

```
{ \x{E1} \x{9B} \x{A7} }
 { \cL\x{78} }
```

U+16E8 RUNIC LETTER ICELANDIC-YR

```
{ \x{E1} \x{9B} \x{A8} }
 { \cL\x{79} }
```

U+16E9 RUNIC LETTER Q

```
{ \x{E1} \x{9B} \x{A9} }
 { \cL\x{7A} }
```

U+16EA RUNIC LETTER X

```
{ \x{E1} \x{9B} \x{AA} }
 { \cL\{ }
```

U+16EE RUNIC ARLAUG SYMBOL

```
{ \x{E1} \x{9B} \x{AE} }
 { \cL\| \cO\! }
```

U+16EF RUNIC TVIMADUR SYMBOL

```
{ \x{E1} \x{9B} \x{AF} }
 { \cL\} \cO\" }
```

U+16F0 RUNIC BELGTHOR SYMBOL

```
{ \x{E1} \x{9B} \x{B0} }
 { \cL\} \cO\# }
```

U+16F1 RUNIC LETTER K

```
{ \x{E1} \x{9B} \x{B1} }
 { \cL\} \cO\$ }
```

U+16F2 RUNIC LETTER SH

```
{ \x{E1} \x{9B} \x{B2} }
 { \cL\} \cO\% }
```

U+16F3 RUNIC LETTER OO

```
{ \x{E1} \x{9B} \x{B3} }
 { \cL\} \cO\& }
```

U+16F4 RUNIC LETTER FRANKS CASKET OS

```
{ \x{E1} \x{9B} \x{B4} }
 { \cL\} \cO\' }
```

U+16F5 RUNIC LETTER FRANKS CASKET IS

```
{ \x{E1} \x{9B} \x{B5} }
 { \cL\} \cO\( }
```

U+16F6 RUNIC LETTER FRANKS CASKET EH

```
{ \x{E1} \x{9B} \x{B6} }
 { \cL\} \cO\) }
```

U+16F7 RUNIC LETTER FRANKS CASKET AC

```
{ \x{E1} \x{9B} \x{B7} }
 { \cL\} \cO\* }
```

U+16F8 RUNIC LETTER FRANKS CASKET AESC

```
{ \x{E1} \x{9B} \x{B8} }
 { \cL\} \cO\+ }
```

Tironian et

```
{ \x{E2} \x{81} \x{8A} }
  { \cL\} \cO\, }
```

U+16EB RUNIC SINGLE PUNCTUATION

```
      { \x{E1} \x{9B} \x{AB} }
       { \c0\~ \c0\! }
```

U+16EC RUNIC MULTIPLE PUNCTUATION

```
      { \x{E1} \x{9B} \x{AC} }
       { \c0\~ \c0\" }
```

U+16ED RUNIC CROSS PUNCTUATION

```
      { \x{E1} \x{9B} \x{AD} }
       { \c0\~ \c0\# }
```

Any character in the punctuation class (unlikely to occur in Runic but added for completeness): Version 1.1 added category code check.

```
      { \c0([[:punct:]]) } { \c0\~ \c0\$ \1 }
    }
    #1
   }
 }
\datatool_locale_define_keys:nn { ang-Runr }
 {
    order .choice:,
    order / punc-unicode .code:n =
     {
       \renewcommand \DTLangRunrLocaleHandler [1]
        {
          \datatool_angRunr_order_punc_unicode:n { ##1 }
        }
     },
    order / punc-futhorc .code:n =
     {
       \renewcommand \DTLangRunrLocaleHandler [1]
        {
          \datatool_angRunr_order_punc_futhorc:n { ##1 }
        }
     },
    order / futhorc-punc .code:n =
     {
       \renewcommand \DTLangRunrLocaleHandler [1]
        {
          \datatool_angRunr_order_futhorc_punc:n { ##1 }
        }
     },
 }
```

Translations: only providing “and” (Tironian et)

```
\newcommand \DTLangRunrTranslations {
  \tl_set:Nn \DTLandname { ⁊ }
}
```

Switch off LaTeX3 syntax.

```
\ExplSyntaxOff
```

Index