

**Proposed Update** Unicode® Standard Annex #42**UNICODE CHARACTER DATABASE IN XML**

Version	Unicode 17.0.0 (draft 2)
Editor	John Wilcock
Date	2025-07-09
This Version	<a href="https://www.unicode.org/reports/tr42/tr42-37.html">https://www.unicode.org/reports/tr42/tr42-37.html</a>
Previous Version	<a href="https://www.unicode.org/reports/tr42/tr42-36.html">https://www.unicode.org/reports/tr42/tr42-36.html</a>
Latest Version	<a href="https://www.unicode.org/reports/tr42/">https://www.unicode.org/reports/tr42/</a>
Latest Proposed Update	<a href="https://www.unicode.org/reports/tr42/proposed.html">https://www.unicode.org/reports/tr42/proposed.html</a>
Schema	<a href="https://www.unicode.org/reports/tr42/tr42-37.mc">https://www.unicode.org/reports/tr42/tr42-37.mc</a>
Revision	37

**Summary**

This annex describes an XML representation of the Unicode Character Database.

**Status**

This is a **draft** document which may be updated, replaced, or superseded by other documents at any time. Publication does not imply endorsement by the Unicode Consortium. This is not a stable document; it is inappropriate to cite this document as other than a work in progress.

A Unicode Standard Annex (UAX) forms an integral part of the Unicode Standard, but is published online as a separate document. The Unicode Standard may require conformance to normative content in a Unicode Standard Annex, if so specified in the Conformance chapter of that version of the Unicode Standard. The version number of a UAX document corresponds to the version of the Unicode Standard of which it forms a part.

Please submit corrigenda and other comments with the online reporting form [\[Feedback\]](#). Related information that is useful in understanding this annex is found in Unicode Standard Annex #41, “[Common References for Unicode Standard Annexes.](#)” For the latest version of the Unicode Standard, see [\[Unicode\]](#). For a list of current Unicode Technical Reports, see [\[Reports\]](#). For more information about versions of the Unicode Standard, see [\[Versions\]](#). For any errata which may apply to this annex, see [\[Errata\]](#).

**Contents**

- 1 [Introduction](#)
- 2 [Overall schema](#)
  - 2.1 [General principles](#)
  - 2.2 [Namespace](#)
  - 2.3 [Datatypes](#)
  - 2.4 [Root Element](#)
  - 2.5 [Common attributes](#)
  - 2.6 [Ordering of elements](#)
- 3 [Description](#)
- 4 [Repertoire](#)
  - 4.1 [Sets of code points](#)
  - 4.2 [Code point types](#)
  - 4.3 [Group](#)
  - 4.4 [Properties](#)
    - 4.4.1 [Age property](#)
    - 4.4.2 [Name properties](#)
    - 4.4.3 [Name Alias properties](#)
    - 4.4.4 [Block property](#)
    - 4.4.5 [General Category](#)
    - 4.4.6 [Combining properties](#)
    - 4.4.7 [Bidirectionality properties](#)
    - 4.4.8 [Decomposition properties](#)
    - 4.4.9 [Numeric Properties](#)
    - 4.4.10 [Joining properties](#)
    - 4.4.11 [Linebreak properties](#)
    - 4.4.12 [East Asian Width property](#)
    - 4.4.13 [Case properties](#)
    - 4.4.14 [Script properties](#)
    - 4.4.15 [ISO Comment properties](#)
    - 4.4.16 [Hangul properties](#)
    - 4.4.17 [Indic properties](#)
    - 4.4.18 [Identifier and Pattern and programming language properties](#)
    - 4.4.19 [Properties related to function and graphic characteristics](#)
    - 4.4.20 [Properties related to boundaries](#)
    - 4.4.21 [Properties related to ideographs](#)
    - 4.4.22 [Miscellaneous properties](#)
    - 4.4.23 [Unihan properties](#)
    - 4.4.24 [Tangut data](#)
    - 4.4.25 [Nushu data](#)
    - 4.4.26 [Emoji properties](#)
    - 4.4.27 [Unikemmet properties](#)
- 5 [Blocks](#)
- 6 [Named Sequences](#)
- 7 [Normalization Corrections](#)

- 8 [Standardized Variants](#)
- 9 [CJK Radicals](#)
- 10 [Emoji sources](#)
- 11 [Do Not Emit](#)
- 12 [The full schema](#)
- 13 [Examples](#)
- [Acknowledgments](#)
- [Modifications](#)

## 1 Introduction

In working on Unicode implementations, it is often useful to access the full content of the Unicode Character Database (UCD). For example, in establishing mappings from characters to glyphs in fonts, it is convenient to see the character scalar value, the character name, the character East Asian width, along with the shape and metrics of the proposed glyph to map to; looking at all this data simultaneously helps in evaluating the mapping.

Directly accessing the data files that constitute the UCD is sometimes a daunting proposition. The data is dispersed in a number of files of various formats, and there are just enough peculiarities (all justified by the processing power available at the time the UCD representation was designed) to require a fairly intimate knowledge of the data format itself, in addition to the meaning of the data.

Many programming environments (for example, Java or ICU) do give access to the UCD. However, those environments tend to lag behind releases of the standard, or support only some of the UCD content.

Unibook is a wonderful tool to explore the UCD and in many cases is just the ticket; however, it is difficult to use when the task at hand has not been built-in, or when non-UCD data is to be displayed as well.

This annex presents an alternative representation of the UCD, which is meant to overcome these difficulties. We have chosen an XML representation, because parsing becomes a non-issue: there are a number of XML parsers freely available, and using them is often fairly easy. In addition, there are freely available tools that can perform powerful operations on XML data; for example, XPATH and XQUERY engines can be thought of as a "grep" for XML data and XSLT engines can be thought of as "awk" for XML data.

It is important to note that we are interested in exploring the content of the UCD, rather than in using the UCD data to process character streams. Thus, we are not concerned so much by the speed of processing or the size of our representation.

Our representation supports the creation of documents that represent only parts of the UCD, either by not representing all the characters, or by not representing all the properties. This can be useful when only some of the data is needed.

This annex presents only the XML representation format of the UCD. The data itself is part of the [Unicode Character Database](#).

## 2 Overall schema

### 2.1 General principles

Our schema can be used to create and validate documents which are intended to represent properties of Unicode code points, blocks, named sequences, normalization corrections, standardized variants, CJK radicals and emoji sources. A document may represent the values actually assigned in a given version of the UCD, or it may represent a draft version of the UCD, or a private agreement on Private Use characters. The validity of a XML document with respect to the schema defined in this annex does not assert anything about the correctness of the values.

Valid documents may provide values for only some of the code points, or some of the Unicode properties. Furthermore, they may also incorporate non-Unicode properties.

Our schema is defined using English. However, a useful subset of the validity constraints can be captured using a schema language, thereby simplifying the task of validating documents. We have chosen Relax NG [ISO 19757], in the compact syntax, as the schema language. It is important to stress that the schema which is defined in English imposes more constraints on the documents than can be validated with the Relax NG schema.

An important characteristic of Relax NG is that its schemas do not modify or augment the infoset of the documents. Therefore, it is possible to process our XML representation without using the schema. Also, the schema is relatively straightforward and can be converted mechanically to other schema languages.

While our XML representation is not intended to be used during processing of characters and strings, it is still a design principle for our schema to support the relatively efficient representation of the UCD. This is achieved by an inheritance mechanism, similar to property inheritance in CSS or in XSL:FO (see section 4.3 Group).

Many invariants impose constraints on the values of the different properties for a given code point. For example, if the value of the Numeric Type property is None, then the value of the Numeric Value property should be the empty string; and if the value of the Other Alphabetic property is true, then the value of the Alphabetic property should be true. Those invariants are not captured in the schema.

### 2.2 Namespace

The namespace for our elements is "http://www.unicode.org/ns/2003/ucd/1.0". Our attributes are in the empty namespace.

```
[namespace declaration, 1] =
  default namespace ucd = "http://www.unicode.org/ns/2003/ucd/1.0"
```

In all our examples, we assume that this namespace is the default one.

### 2.3 Datatypes

We use a standard XML Schema datatypes:

```
[datatypes declaration, 2] =
  # default; datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"
```

Characters are pervasive in the UCD, and will need to be represented. Representing characters directly by themselves would seem the most obvious choice; for example, we could express that the decomposition of U+00E8 is "&#x0065;&#x0300;", that is have exactly two characters in (the infoset of) the XML document. However, the current XML specification limits the set of characters that can be part of a document. Another problem is that the various tools (XML parser, XPATH engine, etc.) may equate U+00E8 with U+0065 U+0300, thus making it difficult to figure out which of the two sequences is contained in the database (which is sometimes important for our purposes). Therefore, we chose instead to represent characters by their code points; we follow the usual convention of four to six hexadecimal digits (uppercase) and code points in a sequence separated by space; for example, the decomposition of U+00E8 will be represented by the nine characters "0065 0300" in the infoset.

```
[datatype for code points, 3] =
  single-code-point = xsd:string { pattern = "([1-9A-F]{10})[0-9A-F]{4}" }
  one-or-more-code-points = list { single-code-point + }
```

```
zero-or-more-code-points = list { single-code-point * }
two-code-points = list { single-code-point, single-code-point }
```

## 2.4 Root Element

The root element of valid documents is a `ucd`.

```
[schema start, 4] =
start =
  element ucd { ucd.content }
```

## 2.5 Common attributes

A large number of properties are boolean. We uniformly use the values `Y` and `N` for those:

```
[boolean, 5] =
boolean = "Y" | "N"
```

## 2.6 Ordering of elements

In elements that hold lists of child elements, such as `repertoire`, `group`, or `standardized-variants`, the schema does not require that the child elements be in any particular order.

## 3 Description

The root element may have a `description` child element, which in turn contains any string, which is meant to describe what the XML document purports to describe.

It is recommended that if the document purports to represent the UCD of some Unicode version, the `description` be selected in accord with the rules listed in [\[Versions\]](#); and conversely, that documents which do not purport to represent the UCD be described as such.

```
[description, 6] =
ucd.content &=
  element description { text }?
```

## 4 Repertoire

The `repertoire` child element of the `ucd` element describes the code points and their properties. As we will see shortly, code points can be described individually or as part of a group:

```
[repertoire, 7] =
ucd.content &=
  element repertoire { (code-point | group) + }?
```

### 4.1 Sets of code points

It is often the case that successive code points have the same property values, for a given set of properties. The most striking example is that of an unallocated plane, where all but the last two code points are reserved and have the same property values. Another example is the URO (U+4E00 .. U+9FA5) where all the code points have the same property values if we ignore their name and their Unihan properties.

```
[Set of code points, 8] =
set-of-code-points =
  attribute cp { single-code-point }
  | ( attribute first-cp { single-code-point },
      attribute last-cp { single-code-point } )
```

This observation suggests that it is profitable to represent sets of code points which share the same properties, rather than individual code points. To make the representation of the sets simple, we restrict them to be segments in the code point space, that is a set is defined by the first and last code point it contains. Those are captured by the attributes `first-cp` and `last-cp`. The attribute `cp` is a shorthand notation for the case where the set has a single code point.

In the `repertoire`, there must be at most one `code-point` element for a given code point.

### 4.2 Code point types

When thinking about Unicode code points, it is useful to split them into four types:

those assigned to abstract characters (PUA or not) the noncharacters the surrogate code points the reserved code points

This leads to four elements to describe sets of code points:

```
[Code points, 9] =
code-point |=
  element reserved {
    set-of-code-points,
    code-point-attributes }

code-point |=
  element noncharacter {
    set-of-code-points,
    code-point-attributes }

code-point |=
  element surrogate {
    set-of-code-points,
    code-point-attributes }

code-point |=
  element char {
    set-of-code-points,
    code-point-attributes }
```

### 4.3 Group

While we already recognized the situation where a set of code points have exactly the same set of property values, another common situation is that of code points which have almost all the same property values.

For example, the characters U+1740 BUHID LETTER A .. U+1753 BUHID VOWEL SIGN U all have the age "3.2", and all have the script "Buhd". On the one hand, it is convenient to support data files in which those properties are explicitly listed with every code point, at this makes answering questions like "what is the

age of U+1749?" easier, because that data is expressed right there. On the other hand, this leads to rather large data files, and it also tends to obscure the differences between similar characters.

Our representation accounts for this situation with the notion of groups. A `group` element is simply a container of code points that also holds default values for the properties. If a code point inside a `group` does not list explicitly a property but the `group` lists it, then the code point inherits that property from its `group`. For example, the fragment with explicit properties:

```
<char cp="1740" age="3.2" na="BUHID LETTER A" gc="Lo" sc="Buhd"/>
<char cp="1741" age="3.2" na="BUHID LETTER I" gc="Lo" sc="Buhd"/>
<char cp="1752" age="3.2" na="BUHID VOWEL SIGN I" gc="Mn" sc="Buhd"/>
<char cp="1820" age="3.0" na="MONGOLIAN LETTER A" gc="Lo" sc="Mong"/>
```

is equivalent to this fragment which uses a `group`:

```
<group age="3.2" gc="Lo" sc="Buhd">
  <char cp="1740" na="BUHID LETTER A"/>
  <char cp="1741" na="BUHID LETTER I"/>
  <char cp="1752" na="BUHID VOWEL SIGN I" gc="Mn"/>
  <char cp="1820" age="3.0" na="MONGOLIAN LETTER A" sc="Mong"/>
</group>
```

The element for U+1740 does not have the `age` attribute, and it therefore inherits it from its enclosing `group` element, that is "3.2". On the other hand, the element for U+1820 does have this attribute, so the value is "3.0".

As this example illustrates, the notion of `group` does not necessarily align with the notion of Unicode block. It is entirely defined and limited to our representation. In particular, the value of a property for a code point can always be determined from the XML document alone, assuming that this property and this code point are expressed at all. Of course, one may create an XML representation where the groups happen to coincide with the Unicode blocks.

Groups cannot be nested. The motivation for this limitation is to make the life of consumers easier: either a property is defined by the element for a code point, or it is defined by the immediately enclosing `group` element.

For UCDXML versions prior to 17.0, only non-Unihan attributes are applied to the `group` elements. Starting with 17.0, Unihan attributes are also applied to the `group` elements.

```
[groups, 10] =
group =
  element group {
    code-point-attributes,
    code-point* }
```

#### 4.4 Properties

Each property, except for the `Special_Case_Condition` and `Name_Alias` properties, is represented by an attribute. In an XML data file, the absence of an attribute (may be only on some `code-points`) means that the document does not express the value of the corresponding property. Conversely, the presence of an attribute is an expression of the corresponding property value; the implied null value is represented by the empty string.

The `Name_Alias` property is represented by zero or more `name-alias` child elements. Unlike the situation for properties represented by attributes, it is not possible to determine whether all the aliases have been represented in a data file by inspecting that data file.

The name of an attribute is the abbreviated name of the property as given in the file `PropertyAliases.txt` in the corresponding version of the UCD. For the Unihan properties, the name is that given in the various versions of the Unihan database.

For catalog and enumerated properties, the values are those listed in the file `PropertyValueAliases.txt` in the corresponding version of the UCD; if there is an abbreviated name, it is used, otherwise the long name is used.

Note that the set of possible values for a property captured in this schema may change from one version to the next.

##### 4.4.1 Age property

The `age` attribute captures the version of Unicode in which a code point was assigned to an abstract character, or made a surrogate or non-character.

```
[age attribute, 11] =
code-point-attributes &=
  attribute age { "1.1"
    | "2.0" | "2.1"
    | "3.0" | "3.1" | "3.2"
    | "4.0" | "4.1"
    | "5.0" | "5.1" | "5.2"
    | "6.0" | "6.1" | "6.2" | "6.3"
    | "7.0"
    | "8.0"
    | "9.0"
    | "10.0"
    | "11.0"
    | "12.0" | "12.1"
    | "13.0"
    | "14.0"
    | "15.0" | "15.1"
    | "16.0"
    | "17.0"
    | "unassigned"
  }?
```

##### 4.4.2 Name properties

There are two name properties: the name given by the current version of the standard (`na`), and possibly the name this character had in version 1.0 of the standard (`na1`).

```
[na attribute, 12] =
code-point-attributes &=
  attribute na { "" |
    "CJK UNIFIED IDEOGRAPH-#" |
    "CJK COMPATIBILITY IDEOGRAPH-#" |
    "EGYPTIAN HIEROGLYPH-#" |
    "TANGUT IDEOGRAPH-#" |
    "KHITAN SMALL SCRIPT CHARACTER-#" |
    "NUSHU CHARACTER-#" |
    xsd:string { pattern="[a-zA-Z0-9]+(( -| |[\_ ])[a-zA-Z0-9]+)*" }
  }?
```

```
[na1 attribute, 13] =
code-point-attributes &=
  attribute na1 { "" | xsd:string { pattern="[a-zA-Z0-9]+([\-_ ][a-zA-Z0-9]+)*(\ (.*)?)?" } }?
```

The majority of the characters in Unicode have a name which is of the form CJK UNIFIED IDEOGRAPH-`<code point>`. It also happens that character names cannot contain the character U+0023 # NUMBER SIGN, so we adopted the following convention: if a code point has the attribute `na` (either directly or by inheritance from an enclosing group), then occurrences of the character # in the name are to be interpreted as the value of the code point. For example:

```
<char cp="3400" na="CJK UNIFIED IDEOGRAPH-3400"/>
```

and

```
<char cp="3400" na="CJK UNIFIED IDEOGRAPH-#"/>
```

are equivalent. The # can be in any position in the value of the `na` attribute. The convention also applies just as well to a set of multiple code points:

```
<char cp="3400" na="CJK UNIFIED IDEOGRAPH-3400"/>
<char cp="3401" na="CJK UNIFIED IDEOGRAPH-3401"/>
```

is equivalent to

```
<char cp="3400" na="CJK UNIFIED IDEOGRAPH-#"/>
<char cp="3401" na="CJK UNIFIED IDEOGRAPH-#"/>
```

which in turn is equivalent to:

```
<char first-cp="3400" last-cp="3401" na="CJK UNIFIED IDEOGRAPH-#"/>
```

#### 4.4.3 Name Alias properties

The `Name_Alias` property is represented by zero or more `name-alias` child elements:

```
[name-alias element, 14] =
code-point-attributes &=
  element name-alias {
    attribute alias { xsd:string { pattern="[a-zA-Z0-9]+(-|_|[\-_ ])[a-zA-Z0-9]+)" } },
    attribute type { "abbreviation" | "alternate"
                   | "control" | "correction"
                   | "figment"
                   }? } *
```

#### 4.4.4 Block property

The `Block` property is represented by the `blk` attribute:

```
[blk attribute, 15] =
code-point-attributes &=
  attribute blk {
    "Adlam"
    "Aegean_Numbers"
    "Ahom"
    "Alchemical"
    "Alphabetic_PF"
    "Anatolian_Hieroglyphs"
    "Ancient_Greek_Music"
    "Ancient_Greek_Numbers"
    "Ancient_Symbols"
    "Arabic"
    "Arabic_Ext_A"
    "Arabic_Ext_B"
    "Arabic_Ext_C"
    "Arabic_Math"
    "Arabic_PF_A"
    "Arabic_PF_B"
    "Arabic_Sup"
    "Armenian"
    "Arrows"
    "ASCII"
    "Avestan"
    "Balinese"
    "Bamum"
    "Bamum_Sup"
    "Bassa_Vah"
    "Batak"
    "Bengali"
    "Beria_Erfe"
    "Bhaiksuki"
    "Block_Elements"
    "Bopomofo"
    "Bopomofo_Ext"
    "Box_Drawing"
    "Brahmi"
    "Braille"
    "Buginese"
    "Buhid"
    "Byzantine_Music"
    "Carian"
    "Caucasian_Albanian"
    "Chakma"
    "Cham"
    "Cherokee"
    "Cherokee_Sup"
    "Chess_Symbols"
    "Chorasmanian"
    "CJK"
    "CJK_Compat"
    "CJK_Compat_Forms"
    "CJK_Compat_Ideographs"
    "CJK_Compat_Ideographs_Sup"
    "CJK_Ext_A"
    "CJK_Ext_B"
    "CJK_Ext_C"
    "CJK_Ext_D"
    "CJK_Ext_E"
    "CJK_Ext_F"
```

```

"CJK_Ext_G"
"CJK_Ext_H"
"CJK_Ext_I"
"CJK_Ext_J"
"CJK_Radicals_Sup"
"CJK_Strokes"
"CJK_Symbols"
"Compat_Jamo"
"Control_Pictures"
"Coptic"
"Coptic_Epact_Numbers"
"Counting_Rod"
"Cuneiform"
"Cuneiform_Numbers"
"Currency_Symbols"
"Cypriot_Syllabary"
"Cypro_Minoan"
"Cyrillic"
"Cyrillic_Ext_A"
"Cyrillic_Ext_B"
"Cyrillic_Ext_C"
"Cyrillic_Ext_D"
"Cyrillic_Sup"
"Deseret"
"Devanagari"
"Devanagari_Ext"
"Devanagari_Ext_A"
"Diacriticals"
"Diacriticals_Ext"
"Diacriticals_For_Symbols"
"Diacriticals_Sup"
"Dingbats"
"Dives_Akuru"
"Dogra"
"Domino"
"Duployan"
"Early_Dynastic_Cuneiform"
"Egyptian_Hieroglyph_Format_Controls"
"Egyptian_Hieroglyphs"
"Egyptian_Hieroglyphs_Ext_A"
"Elbasan"
"Elymaic"
"Emoticons"
"Enclosed_Alphanum"
"Enclosed_Alphanum_Sup"
"Enclosed_CJK"
"Enclosed_Ideographic_Sup"
"Ethiopic"
"Ethiopic_Ext"
"Ethiopic_Ext_A"
"Ethiopic_Ext_B"
"Ethiopic_Sup"
"Garay"
"Geometric_Shapes"
"Geometric_Shapes_Ext"
"Georgian"
"Georgian_Ext"
"Georgian_Sup"
"Glagolitic"
"Glagolitic_Sup"
"Gothic"
"Grantha"
"Greek"
"Greek_Ext"
"Gujarati"
"Gunjala_Gondi"
"Gurmukhi"
"Gurung_Khema"
"Half_And_Full_Forms"
"Half_Marks"
"Hangul"
"Hanifi_Rohingya"
"Hanunoo"
"Hatran"
"Hebrew"
"High_PU_Surrogates"
"High_Surrogates"
"Hiragana"
"IDC"
"Ideographic_Symbols"
"Imperial_Aramaic"
"Indic_Number_Forms"
"Indic_Siyag_Numbers"
"Inscriptional_Pahlavi"
"Inscriptional_Parthian"
"IPA_Ext"
"Jamo"
"Jamo_Ext_A"
"Jamo_Ext_B"
"Javanese"
"Kaithi"
"Kaktovik_Numerals"
"Kana_Ext_A"
"Kana_Ext_B"
"Kana_Sup"
"Kanbun"
"Kangxi"
"Kannada"
"Katakana"
"Katakana_Ext"
"Kawi"
"Kayah_Li"
"Kharoshthi"
"Khitani_Small_Script"
"Khmer"
"Khmer_Symbols"
"Khojki"
"Khudawadi"
"Kirat_Rai"

```

```

"Lao"
"Latin_1_Sup"
"Latin_Ext_A"
"Latin_Ext_Additional"
"Latin_Ext_B"
"Latin_Ext_C"
"Latin_Ext_D"
"Latin_Ext_E"
"Latin_Ext_F"
"Latin_Ext_G"
"Lepcha"
"Letterlike_Symbols"
"Limbu"
"Linear_A"
"Linear_B_Ideograms"
"Linear_B_Syllabary"
"Lisu"
"Lisu_Sup"
"Low_Surrogates"
"Lycian"
"Lydian"
"Mahajani"
"Mahjong"
"Makasar"
"Malayalam"
"Mandaic"
"Manichaean"
"Marchen"
"Masaram_Gondi"
"Math_Alphanum"
"Math_Operators"
"Mayan_Numerals"
"Medefaidrin"
"Meetei_Mayek"
"Meetei_Mayek_Ext"
"Mende_Kikakui"
"Merotic_Cursive"
"Merotic_Hieroglyphs"
"Miao"
"Misc_Arrows"
"Misc_Math_Symbols_A"
"Misc_Math_Symbols_B"
"Misc_Pictographs"
"Misc_Symbols"
"Misc_Symbols_Sup"
"Misc_Technical"
"Modi"
"Modifier_Letters"
"Modifier_Tone_Letters"
"Mongolian"
"Mongolian_Sup"
"Mro"
"Multani"
"Music"
"Myanmar"
"Myanmar_Ext_A"
"Myanmar_Ext_B"
"Myanmar_Ext_C"
"Nabataean"
"Nag_Mundari"
"Nandinagari"
"NB"
"New_Tai_Lue"
"Newa"
"Nko"
"Number_Forms"
"Nushu"
"Nyiakeng_Puachue_Hmong"
"OCR"
"Ogham"
"Ol_Chiki"
"Ol_Onal"
"Old_Hungarian"
"Old_Italic"
"Old_North_Arabian"
"Old_Permic"
"Old_Persian"
"Old_Sogdian"
"Old_South_Arabian"
"Old_Turkic"
"Old_Uyghur"
"Oriya"
"Ornamental_Dingbats"
"Osage"
"Osmanya"
"Ottoman_Siyah_Numbers"
"Pahawh_Hmong"
"Palmyrene"
"Pau_Cin_Hau"
"Phags_Pa"
"Phaistos"
"Phoenician"
"Phonetic_Ext"
"Phonetic_Ext_Sup"
"Playing_Cards"
"Psalter_Pahlavi"
"PUA"
"Punctuation"
"Rejang"
"Rumi"
"Runic"
"Samaritan"
"Saurashtra"
"Sharada"
"Sharada_Sup"
"Shavian"
"Shorthand_Format_Controls"
"Siddham"
"Sidetic"

```

```

| "Sinhala"
| "Sinhala_Archaic_Numbers"
| "Small_Forms"
| "Small_Kana_Ext"
| "Sogdian"
| "Sora_Sompeng"
| "Soyombo"
| "Specials"
| "Sundanese"
| "Sundanese_Sup"
| "Sunuwar"
| "Sup_Arrows_A"
| "Sup_Arrows_B"
| "Sup_Arrows_C"
| "Sup_Math_Operators"
| "Sup_PUA_A"
| "Sup_PUA_B"
| "Sup_Punctuation"
| "Sup_Symbols_And_Pictographs"
| "Super_And_Sub"
| "Sutton_SignWriting"
| "Syloti_Nagri"
| "Symbols_And_Pictographs_Ext_A"
| "Symbols_For_Legacy_Computing"
| "Symbols_For_Legacy_Computing_Sup"
| "Syriac"
| "Syriac_Sup"
| "Tagalog"
| "Tagbanwa"
| "Tags"
| "Tai_Le"
| "Tai_Tham"
| "Tai_Viet"
| "Tai_Xuan_Jing"
| "Tai_Yo"
| "Takri"
| "Tamil"
| "Tamil_Sup"
| "Tangsa"
| "Tangsa"
| "Tangut"
| "Tangut_Components"
| "Tangut_Components_Sup"
| "Tangut_Sup"
| "Telugu"
| "Thaana"
| "Thai"
| "Tibetan"
| "Tifinagh"
| "Tirhuta"
| "Todhri"
| "Tolong_Siki"
| "Toto"
| "Transport_And_Map"
| "Tulu_Tigalari"
| "UCAS"
| "UCAS_Ext"
| "UCAS_Ext_A"
| "Ugaritic"
| "Vai"
| "Vedic_Ext"
| "Vertical_Forms"
| "Vithkuqi"
| "VS"
| "VS_Sup"
| "Wancho"
| "Warang_Citi"
| "Yezidi"
| "Yi_Radicals"
| "Yi_Syllables"
| "Yijing"
| "Zanabazar_Square"
| "Znamenny_Music"
| }?

```

#### 4.4.5 General Category

The general category is represented by the `gc` attribute.

```

[gc attribute, 16] =
code-point-attributes &=
  attribute gc {
    "Cc" | "Cf" | "Cn" | "Co" | "Cs"
    "Ll" | "Lm" | "Lo" | "Lt" | "Lu"
    "Mc" | "Me" | "Mn"
    "Nd" | "Nl" | "No"
    "Pc" | "Pd" | "Pe" | "Pf" | "Pi" | "Po" | "Ps"
    "Sc" | "Sk" | "Sm" | "So"
    "Zl" | "Zp" | "Zs"
  }?

```

#### 4.4.6 Combining properties

The combining class is represented by the `ccc` attribute, which holds the decimal representation of the combining class.

Because the set of values that this property has taken across the various versions of the UCD is rather large, our schema does not restrict the possible values to those actually used.

```

[ccc attribute, 17] =
code-point-attributes &=
  attribute ccc { xsd:integer { minInclusive="0" maxInclusive="254" } }?

```

#### 4.4.7 Bidirectionality properties

The bidirectional class is represented by the `bc` attribute.

```
[bc attribute, 18] =
code-point-attributes &=
  attribute bc {
    "AL" | "AN"
    "B" | "BN"
    "CS"
    "EN" | "ES" | "ET"
    "FSI"
    "L" | "LRE" | "LRI" | "LRO"
    "NSM"
    "ON"
    "PDF" | "PDI"
    "R" | "RLE" | "RLI" | "RLO"
    "S"
    "WS"
  }?
```

The mirrored property is represented by the `bidirectional_mirrored` attribute, which takes a boolean value.

```
[Bidirectional_Mirrored attribute, 19] =
code-point-attributes &=
  attribute Bidirectional_Mirrored { boolean }?
```

The `bidirectional_mirror_glyph` attribute is the code point of a character whose glyph is typically a mirrored image of the glyph for the current character.

```
[bidirectional_mirror_glyph attribute, 20] =
code-point-attributes &=
  attribute bidirectional_mirror_glyph { "" | single-code-point }?
```

Note that we do not express the "Best Fit" element recorded in `BidiMirroring.txt`. For one thing, it is not meant to be machine readable. More importantly, the idea underlying the mirrored glyph is delicate to use, since it makes assumptions about the design of the fonts, and the best fit goes even farther.

The `Bidi_Control` property is represented by the `bidirectional_control` attribute.

```
[Bidirectional_Control attribute, 21] =
code-point-attributes &=
  attribute Bidirectional_Control { boolean }?
```

The `bidirectional_paired_bracket_type` and `bidirectional_paired_bracket` properties are represented by the `bidirectional_paired_bracket_type` and `bidirectional_paired_bracket` attributes respectively.

```
[bidirectional_paired_bracket_type attribute, 22] =
code-point-attributes &=
  attribute bidirectional_paired_bracket_type { "o" | "c" | "n" }?
```

```
[bidirectional_paired_bracket attribute, 23] =
code-point-attributes &=
  attribute bidirectional_paired_bracket { "#" | single-code-point }?
```

#### 4.4.8 Decomposition properties

The decomposition type and decomposition mapping properties are represented by the `decomposition_type` and `decomposition_mapping` attributes.

Most characters have a decomposition mapping to themselves. This is very similar to the situation we encountered with names, and we adopted a similar convention: if the value of a decomposition mapping is the character itself, we use the attribute value `#` (U+0023 # NUMBER SIGN) as a shorthand notation; this enables those attributes to be captured in groups.

```
[decomposition_type attribute, 24] =
code-point-attributes &=
  attribute dt {
    "can" | "com" | "enc" | "fin" | "font" | "fra"
    "init" | "iso" | "med" | "nar" | "nb" | "sml"
    "sqr" | "sub" | "sup" | "vert" | "wide" | "none"
  }?

code-point-attributes &=
  attribute dm { "#" | zero-or-more-code-points }?
```

The properties `Composition_Exclusion` and `Full_Composition_Exclusion` are represented by the attributes `composition_exclusion` and `full_composition_exclusion`:

```
[composition_exclusion attribute, 25] =
code-point-attributes &=
  attribute composition_exclusion { boolean }?

code-point-attributes &=
  attribute full_composition_exclusion { boolean }?
```

The properties `NFC_Quick_Check`, `NFD_Quick_Check`, `NFKC_Quick_Check`, `NFKD_Quick_Check`, `Expands_On_NFC`, `Expands_On_NFD`, `Expands_On_NFKC`, `Expands_On_NFKD`, `FC`, `NFKC_Closure` have corresponding attributes.

```
[quick_check attribute, 26] =
code-point-attributes &=
  attribute NFC_Quick_Check { "Y" | "N" | "M" }?

code-point-attributes &=
  attribute NFD_Quick_Check { "Y" | "N" }?

code-point-attributes &=
  attribute NFKC_Quick_Check { "Y" | "N" | "M" }?

code-point-attributes &=
  attribute NFKD_Quick_Check { "Y" | "N" }?
```

```
code-point-attributes &=
  attribute Expands_On_NFC { boolean }?
```

```
code-point-attributes &=
  attribute Expands_On_NFD { boolean }?
```

```
code-point-attributes &=
  attribute Expands_On_NFKC { boolean }?
```

```
code-point-attributes &=
  attribute Expands_On_NFKD { boolean }?
```

```
code-point-attributes &=
  attribute FC_NFKC ( "#" | one-or-more-code-points );
```

#### 4.4.9 Numeric Properties

The numeric type is represented by the `nt` attribute.

The numeric value is represented by the `nv` attribute, represented as a whole number or a fraction.

[numeric properties, 27] =

```
code-point-attributes &=
  attribute nt { "De" | "Di" | "Nu" | "None" }?

code-point-attributes &=
  attribute nv { "NaN" | xsd:string { pattern="-?[0-9]+(/[0-9]+)?" } }?
```

#### 4.4.10 Joining properties

The joining class of a character is represented by the `jt` attribute.

The `jg` attribute is the joining group of the character.

[joining properties, 28] =

```
code-point-attributes &=
  attribute jt { "C" | "D" | "L" | "R" | "T" | "U" }?

code-point-attributes &=
  attribute jg { "African_Feh" | "African_Noon" | "African_Qaf"
    | "Ain" | "Alaph" | "Alef"
    | "Beh" | "Beth" | "Burushaski_Yeh_Barree"
    | "Dal" | "Dalath_Rish"
    | "E"
    | "Farsi_Yeh" | "Fe" | "Feh" | "Final_Semkath"
    | "Gaf" | "Gamal"
    | "Hah" | "Hanifi_Rohingya_Kinna_Ya"
    | "Hanifi_Rohingya_Pa" | "He" | "Heh" | "Heh_Goal"
    | "Heth"
    | "Kaf" | "Kaph" | "Kashmiri_Yeh" | "Khaph"
    | "Knotted_Heh"
    | "Lam" | "Lamadh"
    | "Malayalam_Bha" | "Malayalam_Ja" | "Malayalam_Lla"
    | "Malayalam_Llla" | "Malayalam_Nga"
    | "Malayalam_Nna" | "Malayalam_Nnna"
    | "Malayalam_Nya" | "Malayalam_Ra" | "Malayalam_Ssa"
    | "Malayalam_Tta" | "Manichaeen_Aleph"
    | "Manichaeen_Ayin" | "Manichaeen_Beth"
    | "Manichaeen_Daleth" | "Manichaeen_Dhamedh"
    | "Manichaeen_Five" | "Manichaeen_Gimel"
    | "Manichaeen_Heth" | "Manichaeen_Hundred"
    | "Manichaeen_Kaph" | "Manichaeen_Lamedh"
    | "Manichaeen_Mem" | "Manichaeen_Nun"
    | "Manichaeen_One" | "Manichaeen_Pe"
    | "Manichaeen_Qoph" | "Manichaeen_Resh"
    | "Manichaeen_Sadhe" | "Manichaeen_Samekh"
    | "Manichaeen_Taw" | "Manichaeen_Ten"
    | "Manichaeen_Teth" | "Manichaeen_Thamedh"
    | "Manichaeen_Twenty" | "Manichaeen_Waw"
    | "Manichaeen_Yodh" | "Manichaeen_Zayin" | "Meem"
    | "Mim"
    | "No_Joining_Group" | "Noon" | "Nun" | "Nya"
    | "Pe"
    | "Qaf" | "Qaph"
    | "Reh" | "Reversed_Pe" | "Rohingya_Yeh"
    | "Sad" | "Sadhe" | "Seen" | "Semkath" | "Shin"
    | "Straight_Waw" | "Swash_Kaf" | "Syriac_Waw"
    | "Tah" | "Taw" | "Teh_Marbuta" | "Teh_Marbuta_Goal"
    | "Teth" | "Thin_Noon" | "Thin_Yeh"
    | "Vertical_Tail"
    | "Waw"
    | "Yeh" | "Yeh_Barree" | "Yeh_With_Tail" | "Yudh"
    | "Yudh_He"
    | "Zain" | "Zhain"
    | "BAA"
    | "FA"
    | "HAA" | "HA_GOAL" | "HA"
    | "CAF"
    | "KNOTTED_HA"
    | "RA"
    | "SWASH_CAF"
    | "HAMZAH_ON_HA_GOAL"
    | "TAA_MARBUTAH"
    | "YA_BARREE" | "YA"
    | "ALEF_MAQSURAH"
  }?
```

The `Join_Control` property is represented by the `Join_c` attribute.

[joining properties, 29] =

```
code-point-attributes &=
  attribute Join_C { boolean }?
```

#### 4.4.11 Linebreak properties

The `Line_Break` property is represented by the `lb` attribute.

[lb attribute, 30] =

```
code-point-attributes &=
  attribute lb { "AI" | "AK" | "AL" | "AP" | "AS"
    | "B2" | "BA" | "BB" | "BK"
    | "CB" | "CJ" | "CL" | "CM" | "CP" | "CR"
    | "EB" | "EM" | "EX" }
```

```

| "GL"
| "H2" | "H3" | "HH" | "HL" | "HY"
| "ID" | "IN" | "IS"
| "JL" | "JT" | "JV"
| "LF"
| "NL" | "NS" | "NU"
| "OP"
| "PO" | "PR"
| "QU"
| "RI"
| "SA" | "SG" | "SP" | "SY"
| "VF" | "VI"
| "WJ"
| "XX"
| "ZW" | "ZWJ"
| }?

```

#### 4.4.12 East Asian Width property

The East Asian width property is represented by the `ea` attribute.

```

[ea attribute, 31] =
code-point-attributes &=
  attribute ea { "A" | "F" | "H" | "N" | "Na" | "W" }?

```

#### 4.4.13 Case properties

The Uppercase, Lowercase, Other\_Uppercase and Other\_Lowercase properties are represented by corresponding attributes.

```

[ casing properties, 32 ] =
code-point-attributes &=
  attribute Upper { boolean }?

code-point-attributes &=
  attribute Lower { boolean }?

code-point-attributes &=
  attribute OUpper { boolean }?

code-point-attributes &=
  attribute OLower { boolean }?

```

Most characters have a case mapping and case folding properties that simply map or fold to themselves. This is very similar to the situation we encountered with names, and we adopted a similar convention: if the value of a case mapping or case folding property is the character itself, we use the attribute value `#` (U+0023 # NUMBER SIGN) as a shorthand notation; this enables those attributes to be captured in groups.

The simple case mappings are recorded in the `suc`, `s1c`, `stc` attributes.

```

[ casing properties, 33 ] =
code-point-attributes &=
  attribute suc { "#" | single-code-point }?

code-point-attributes &=
  attribute s1c { "#" | single-code-point }?

code-point-attributes &=
  attribute stc { "#" | single-code-point }?

```

The non-simple casing are recorded in the `uc`, `lc` and `tc` attributes.

```

[ casing properties, 34 ] =
code-point-attributes &=
  attribute uc { "#" | one-or-more-code-points }?

code-point-attributes &=
  attribute lc { "#" | one-or-more-code-points }?

code-point-attributes &=
  attribute tc { "#" | one-or-more-code-points }?

```

The Simple\_Case\_Folding and Case\_Folding properties are recorded in the `scf` and `cf` attributes respectively.

```

[ casing properties, 35 ] =
code-point-attributes &=
  attribute scf { "#" | single-code-point }?

code-point-attributes &=
  attribute cf { "#" | one-or-more-code-points }?

```

The Case\_Ignorable, Cased, Changes\_When\_Casefolded, Changes\_When\_Casemapped, Changes\_When\_Lowercased, Changes\_When\_NFKC\_Casefolded, Changes\_When\_Titlecased, Changes\_When\_Uppercased, NFKC\_Casefold, and NFKC\_Simple\_Casefold properties are recorded in these attributes:

```

[ casing properties, 36 ] =
code-point-attributes &=
  attribute CI { boolean }?

code-point-attributes &=
  attribute Cased { boolean }?

code-point-attributes &=
  attribute CWCF { boolean }?

code-point-attributes &=
  attribute CWCM { boolean }?

code-point-attributes &=
  attribute CWL { boolean }?

code-point-attributes &=
  attribute CWKCF { boolean }?

code-point-attributes &=
  attribute CWT { boolean }?

```

```
code-point-attributes &=
  attribute CWU { boolean }?

code-point-attributes &=
  attribute NFKC_CF { "#" | zero-or-more-code-points }?

code-point-attributes &=
  attribute NFKC_SCF { "#" | zero-or-more-code-points }?
```

Note that the UCD records more information about case folding than is expressed in the properties, specifically the entries in CaseFolding.txt with status T.

#### 4.4.14 Script properties

The script and script extension properties are represented by the `sc` and `scx` attributes respectively.

[script properties, 37] =

```
script = "Adlm" | "Aghb" | "Ahom" | "Arab" | "Armi" | "Armn"
| "Avst"
| "Bali" | "Bamu" | "Bass" | "Batk" | "Beng" | "Berf"
| "Bhks" | "Bopo" | "Brah" | "Brai" | "Bugi" | "Buhd"
| "Cakm" | "Cans" | "Cari" | "Cham" | "Cher" | "Chrs"
| "Copt" | "Cpmn" | "Cprt" | "Cyr1"
| "Deva" | "Diak" | "Dogr" | "Dsrt" | "Dupl"
| "Egyp" | "Elba" | "Elym" | "Ethi"
| "Gara" | "Geor" | "Glag" | "Gong" | "Gomr" | "Goth"
| "Gran" | "Grek" | "Gujr" | "Gukh" | "Guru"
| "Hang" | "Hani" | "Hano" | "Hatr" | "Hebr" | "Hira"
| "Hluw" | "Hmgg" | "Hmnp" | "Hrkt" | "Hung"
| "Ital"
| "Java"
| "Kali" | "Kana" | "Kawi" | "Khar" | "Khmr" | "Khoj"
| "Kits" | "Knda" | "Krai" | "Kthi"
| "Lana" | "Laoo" | "Latn" | "Lepc" | "Limb" | "Lina"
| "Linb" | "Lisu" | "Lyci" | "Lydi"
| "Mahj" | "Maka" | "Mand" | "Mani" | "Marc" | "Medf"
| "Mend" | "Merc" | "Mero" | "Mlym" | "Modi" | "Mong"
| "Mroo" | "Mtei" | "Mult" | "Mymr"
| "Nagm" | "Nand" | "Narb" | "Nbat" | "Newa" | "Nkoo"
| "Nshu"
| "Ogam" | "Olck" | "Onao" | "Orkh" | "Orya" | "Osge"
| "Osma" | "Ougr"
| "PalM" | "Pauc" | "Perm" | "Phag" | "Phli" | "Phlp"
| "Phnx" | "Plrd" | "Prti"
| "Rjng" | "Rohg" | "Runr"
| "Samr" | "Sarb" | "Saur" | "Sgnw" | "Shaw" | "Shrd"
| "Sidd" | "Sidd" | "Sind" | "Sinh" | "Sogd" | "Sogo"
| "Sora" | "Soyo" | "Sund" | "Sunu" | "Sylo" | "Sync"
| "Tagb" | "Takr" | "Tale" | "Talu" | "Taml" | "Tang"
| "Tavt" | "Tavo" | "Telu" | "Tfng" | "Tglg" | "Thaa"
| "Thai" | "Tibt" | "Tirh" | "Tnsa" | "Todr" | "Tols"
| "Toto" | "Tutg"
| "Ugar"
| "Vaii" | "Vith"
| "Wara" | "Wcho"
| "Xpeo" | "Xsux"
| "Yezi" | "Yiii"
| "Zanb" | "Zinh" | "Zyyy" | "Zzzz"
```

```
code-point-attributes &=
  attribute sc { script }?

code-point-attributes &=
  attribute scx { list { script + } }?
```

#### 4.4.15 ISO Comment properties

The ISO 10646 comment field is represented by the `isc` attribute.

[isc attribute, 38] =

```
code-point-attributes &=
  attribute isc { text }
```

#### 4.4.16 Hangul properties

The property Hangul\_Syllable\_Type is represented by the `hst` attribute.

[hst attribute, 39] =

```
code-point-attributes &=
  attribute hst { "L" | "LV" | "LVT" | "NA" | "T" | "V" }?
```

The property Jamo\_Short\_Name is represented by the `jsn` attribute:

[JSN attribute, 40] =

```
code-point-attributes &=
  attribute JSN { xsd:string { pattern="[A-Z]{0,3}" } }?
```

#### 4.4.17 Indic properties

The property Indic\_Syllabic\_Category is represented by the `inSC` attribute.

[InSC attribute, 41] =

```
code-point-attributes &=
  attribute InSC { "Avagraha"
| "Bindu"
| "Brahmi_Joining_Number"
| "Cantillation_Mark"
| "Consonant"
| "Consonant_Dead"
| "Consonant_Final"
| "Consonant_Head_Letter"
| "Consonant_Initial_Postfixed"
```

```

| "Consonant_Killer"
| "Consonant_Medial"
| "Consonant_Placeholder"
| "Consonant_Preceding_Repha"
| "Consonant_Prefixed"
| "Consonant_Repha"
| "Consonant_Subjoined"
| "Consonant_Succeeding_Repha"
| "Consonant_With_Stacker"
| "Gemination_Mark"
| "Invisible_Stacker"
| "Joiner"
| "Modifying_Letter"
| "Non_Joiner"
| "Nukta"
| "Number"
| "Number_Joiner"
| "Other"
| "Pure_Killer"
| "Register_Shifter"
| "Reordering_Killer"
| "Syllable_Modifier"
| "Tone_Letter"
| "Tone_Mark"
| "Virama"
| "Visarga"
| "Vowel"
| "Vowel_Dependent"
| "Vowel_Independent"
}?)

```

The property Indic\_Positional\_Category is represented by the `InPC` attribute:

```

[InPC attribute, 42] =
code-point-attributes &=
  attribute InPC {
    "Bottom"
    "Bottom_And_Left"
    "Bottom_And_Right"
    "Invisible"
    "Left"
    "Left_And_Right"
    "NA"
    "Overstruck"
    "Right"
    "Top"
    "Top_And_Bottom"
    "Top_And_Bottom_And_Left"
    "Top_And_Bottom_And_Right"
    "Top_And_Left"
    "Top_And_Left_And_Right"
    "Top_And_Right"
    "Visual_Order_Left"
  }?)

```

The property Indic\_Conjunct\_Break is represented by the `InCB` attribute:

```

[InCB attribute, 43] =
code-point-attributes &=
  attribute InCB {
    "Consonant"
    "Extend"
    "Linker"
    "None"
  }?)

```

#### 4.4.18 Identifier and Pattern and programming language properties

The properties `ID_Start`, `Other_ID_Start`, `XID_Start`, `ID_Continue`, `Other_ID_Continue`, `XID_Continue`, `ID_Compat_Math_Start`, and `ID_Compat_Math_Continue` are represented by corresponding attributes:

```

[Identifier properties, 44] =
code-point-attributes &=
  attribute IDS { boolean }?

code-point-attributes &=
  attribute OIDS { boolean }?

code-point-attributes &=
  attribute XIDS { boolean }?

code-point-attributes &=
  attribute IDC { boolean }?

code-point-attributes &=
  attribute OIDC { boolean }?

code-point-attributes &=
  attribute XIDC { boolean }?

code-point-attributes &=
  attribute ID_Compat_Math_Start { boolean }?

code-point-attributes &=
  attribute ID_Compat_Math_Continue { boolean }?

```

The properties `Pattern_Syntax` and `Pattern_White_Space` are represented by corresponding attributes:

```

[pattern properties, 45] =
code-point-attributes &=
  attribute Pat_Syn { boolean }?

code-point-attributes &=
  attribute Pat_WS { boolean }?

```

#### 4.4.19 Properties related to function and graphic characteristics

The properties Dash, Hyphen, Quotation\_Mark, Terminal\_Punctuation, Sentence\_Terminal, Diacritic, Extender, Soft\_Dotted, Alphabetic, Other\_Alphabetic, Math, Other\_Math, Hex\_Digit, ASCII\_Hex\_Digit, Default\_Ignorable\_Code\_Point, Other\_Default\_Ignorable\_Code\_Point, Logical\_Order\_Exception, Prepend\_Concatenation\_Mark, Modifier\_Combining\_Mark, White\_Space, Vertical\_Orientation, and Regional\_Indicator describe the function or graphic characteristic of a character, and have each a corresponding attribute.

*[properties related to function and graphic characteristics, 46] =*

```
code-point-attributes &=
  attribute Dash { boolean }?

code-point-attributes &=
  attribute Hyphen { boolean }?

code-point-attributes &=
  attribute QMark { boolean }?

code-point-attributes &=
  attribute Term { boolean }?

code-point-attributes &=
  attribute STerm { boolean }?

code-point-attributes &=
  attribute Dia { boolean }?

code-point-attributes &=
  attribute Ext { boolean }?

code-point-attributes &=
  attribute SD { boolean }?

code-point-attributes &=
  attribute Alpha { boolean }?

code-point-attributes &=
  attribute OAlpha { boolean }?

code-point-attributes &=
  attribute Math { boolean }?

code-point-attributes &=
  attribute OMath { boolean }?

code-point-attributes &=
  attribute Hex { boolean }?

code-point-attributes &=
  attribute AHex { boolean }?

code-point-attributes &=
  attribute DI { boolean }?

code-point-attributes &=
  attribute ODI { boolean }?

code-point-attributes &=
  attribute LOE { boolean }?

code-point-attributes &=
  attribute PCM { boolean }?

code-point-attributes &=
  attribute MCM { boolean }?

code-point-attributes &=
  attribute WSpace { boolean }?

code-point-attributes &=
  attribute vo { "R" | "Tr" | "Tu" | "U" }?

code-point-attributes &=
  attribute RI { boolean }?
```

#### 4.4.20 Properties related to boundaries

The properties Grapheme\_Base, Grapheme\_Extend, Other\_Grapheme\_Extend, Grapheme\_Link, Grapheme\_Cluster\_Break, Word\_Break, and Sentence\_Break each have a corresponding attribute:

*[properties related to boundaries, 47] =*

```
code-point-attributes &=
  attribute Gr_Base { boolean }?

code-point-attributes &=
  attribute Gr_Ext { boolean }?

code-point-attributes &=
  attribute OGr_Ext { boolean }?

code-point-attributes &=
  attribute Gr_Link { boolean }?

code-point-attributes &=
  attribute GCB { "CN" | "CR"
    | "EB" | "EBG" | "EM" | "EX"
    | "GAZ"
    | "L" | "LF" | "LV" | "LVT"
    | "PP"
    | "RI"
    | "SM"
    | "T"
    | "V"
    | "XX"
    | "ZWJ"
  }?

code-point-attributes &=
  attribute WB { "CR"
```

```

| "DQ"
| "EB" | "EBG" | "EM" | "EX" | "Extend"
| "FO"
| "GAZ"
| "HL"
| "KA"
| "LE" | "LF"
| "MB" | "ML" | "MN"
| "NL" | "NU"
| "RI"
| "SQ"
| "WSegSpace"
| "XX"
| "ZWJ"
}|?

```

```

code-point-attributes &=
  attribute SB {
    | "AT"
    | "CL" | "CR"
    | "EX"
    | "FO"
    | "LE" | "LF" | "LO"
    | "NU"
    | "SC" | "SE" | "SP" | "ST"
    | "UP"
    | "XX"
  }?

```

#### 4.4.21 Properties related to ideographs

The properties Ideographic, Unified\_Ideograph, Equivalent\_Unified\_Ideograph, IDS\_Binary\_Operator, IDS\_Tertiary\_Operator, IDS\_Unary\_Operator, and Radical have corresponding attributes:

[properties related to ideographs, 48] =

```

code-point-attributes &=
  attribute Ideo { boolean }?

code-point-attributes &=
  attribute UIdeo { boolean }?

code-point-attributes &=
  attribute EqUIdeo { single-code-point }?

code-point-attributes &=
  attribute IDSB { boolean }?

code-point-attributes &=
  attribute IDST { boolean }?

code-point-attributes &=
  attribute IDSU { boolean }?

code-point-attributes &=
  attribute Radical { boolean }?

```

#### 4.4.22 Miscellaneous properties

The properties Deprecated, Variation\_Selector, and Noncharacter\_Code\_Point have corresponding attributes:

[miscellaneous properties, 49] =

```

code-point-attributes &=
  attribute Dep { boolean }?

code-point-attributes &=
  attribute VS { boolean }?

code-point-attributes &=
  attribute NChar { boolean }?

```

#### 4.4.23 Unihan properties

The Unihan properties (from the Unihan database) are represented as attributes.

[Unihan properties, 50] =

```

code-point-attributes &= attribute kAccountingNumeric
  { xsd:string { pattern="\d+" } }?

code-point-attributes &= attribute kAlternateTotalStrokes
  { list { xsd:string { pattern="(\d+:[BGHJKMPSTUV]+)-" }+ } }?

code-point-attributes &= attribute kBigFive
  { xsd:string { pattern="[0-9A-F]{4}?" } }?

code-point-attributes &= attribute kCangjie
  { xsd:string { pattern="[A-Z]+" } }?

code-point-attributes &= attribute kCantonese
  { list { xsd:string { pattern="[a-z]{1,6}[1-6]" }+ } }?

code-point-attributes &= attribute kCCCII
  { list { xsd:string { pattern="[0-9A-F]{6}" }+ } }?

code-point-attributes &= attribute kCheungBauer
  { list { xsd:string { pattern="\d{3}/\d{2};[A-Z]*;[a-z1-6[/\,]+ }+ } }?

code-point-attributes &= attribute kCheungBauerIndex
  { list { xsd:string { pattern="\d{3}\.[01]\d" }+ } }?

code-point-attributes &= attribute kCihaiT
  { list { xsd:string { pattern="[1-9]\d{0,3}\.\d{3}" }+ } }?

code-point-attributes &= attribute kCNS1986
  { xsd:string { pattern="[12E]-[0-9A-F]{4}" } }?

code-point-attributes &= attribute kCNS1992

```

```

    { xsd:string { pattern="[1-9]-[0-9A-F]{4}" } }?
code-point-attributes &= attribute kCompatibilityVariant
  { "" | xsd:string { pattern="U+[23]?[0-9A-F]{4}" } }?
code-point-attributes &= attribute kWowles
  { list { xsd:string { pattern="\d{1,4}(\.\d{1,2})?" }+ } }?
code-point-attributes &= attribute kDaeJaweon
  { xsd:string { pattern="\d{4}\.\d{2}[01]" } }?
code-point-attributes &= attribute kDefinition
  { xsd:string { pattern="[\t]" } }?
code-point-attributes &= attribute kEACC
  { xsd:string { pattern="[0-9A-F]{6}" } }?
code-point-attributes &= attribute kFanqie
  { list { xsd:string { pattern="[\x{3400}-\x{4DBF}\x{4E00}-\x{9FFF}\x{20000}-\x{2A6DF}]{2}" }+ } }?
code-point-attributes &= attribute kFenn
  { list { xsd:string { pattern="\d+a?[A-KP*]" }+ } }?
code-point-attributes &= attribute kFennIndex
  { list { xsd:string { pattern="\d{1,3}\.[01]\d" }+ } }?
code-point-attributes &= attribute kFourCornerCode
  { list { xsd:string { pattern="\d{4}(\.\d)?" }+ } }?
code-point-attributes &= attribute kGB0
  { xsd:string { pattern="\d{4}" } }?
code-point-attributes &= attribute kGB1
  { xsd:string { pattern="\d{4}" } }?
code-point-attributes &= attribute kGB3
  { xsd:string { pattern="\d{4}" } }?
code-point-attributes &= attribute kGB5
  { xsd:string { pattern="\d{4}" } }?
code-point-attributes &= attribute kGB7
  { xsd:string { pattern="[0-9A-F]{4}" } }?
code-point-attributes &= attribute kGB8
  { xsd:string { pattern="\d{4}" } }?
code-point-attributes &= attribute kGradeLevel
  { xsd:string { pattern="[1-6]" } }?
code-point-attributes &= attribute kGSR
  { list { xsd:string { pattern="\d{4}[a-vx-z]'" }+ } }?
code-point-attributes &= attribute kHangul
  { list { xsd:string { pattern="[\x{1100}-\x{1112}][\x{1161}-\x{1175}][\x{11A8}-\x{11C2}]?:[01ENX]{1,3}" }+ } }?
code-point-attributes &= attribute kHanYu
  { list { xsd:string { pattern="[1-8]\d{4}\.[0-3]\d{0-3}" }+ } }?
code-point-attributes &= attribute kHanyuPinlu
  { list { xsd:string { pattern="[a-z]\x{300}-\x{302}\x{304}\x{308}\x{30C}+(\d+)" }+ } }?
code-point-attributes &= attribute kHanyuPinyin
  { list { xsd:string { pattern="(\d{5}\.\d{2}0)*\d{5}\.\d{2}0:([a-z]\x{300}-\x{302}\x{304}\x{308}\x{30C})+,*[a-z]\x{300}-\x{302}\x{304}\x{308}\x{30C})+" }+ } }?
code-point-attributes &= attribute kHZRadBreak
  { xsd:string { pattern="[\x{2F00}-\x{2FD5}][U+2F[0-9A-D][0-9A-F]\:[1-8]\d{4}\.[0-3]\d0" } }?
code-point-attributes &= attribute kHKGlyph
  { list { xsd:string { pattern="\d{4}" }+ } }?
code-point-attributes &= attribute kIBMJapan
  { list { xsd:string { pattern="F[ABC][0-9A-F]{2}" }+ } }?
code-point-attributes &= attribute kIICore
  { list { xsd:string { pattern="ABC[GHJKMPT]{1,7}" }+ } }?
code-point-attributes &= attribute kIRG_GSource
  { "" | xsd:string { pattern="G[013578EKS]-[0-9A-F]{4}" }
    | xsd:string { pattern="G(DZ|GH|RM|WZ|XC|XH|ZH)-\d{4}\.\d{2}" }
    | xsd:string { pattern="GKX-\d{4}\.\d{2,3}" }
    | xsd:string { pattern="G(HZ|HZR)-\d{5}\.\d{2}" }
    | xsd:string { pattern="G(CE|FC|IDC23|OCD|XHZ)-\d{3}" }
    | xsd:string { pattern="G(H|HF|LGVJ|PGLG|T|ZHSJ)-\d{4}" }
    | xsd:string { pattern="G(4K|CESI|CVY|DM|GT|JZ|KJ|XM|WY|ZFY|ZJW|ZYS)-\d{5}" }
    | xsd:string { pattern="G(FZ|IDC)-[0-9A-F]{4}" }
    | xsd:string { pattern="GCA-[A-Z]\d{4}" }
    | xsd:string { pattern="GGFZ-\d{6}" }
    | xsd:string { pattern="G(BK|LK|Z)-\d{7}" }
    | xsd:string { pattern="G(CH|CY|HC|U)-[023][0-9A-F]{4}" }
    | xsd:string { pattern="GZA-[123467]\d{5}" }
  }?
code-point-attributes &= attribute kIRG_HSource
  { "" | xsd:string { pattern="H-[0-9A-F]{4}" }
    | xsd:string { pattern="H(B[012])- [0-9A-F]{4}" }
    | xsd:string { pattern="HD-[23]?[0-9A-F]{4}" }
    | xsd:string { pattern="HU-[023][0-9A-F]{4}" }
  }?
code-point-attributes &= attribute kIRG_JSource
  { "" | xsd:string { pattern="J[014]-[0-9A-F]{4}" }
    | xsd:string { pattern="J3A?-[0-9A-F]{4}" }
    | xsd:string { pattern="J13A?-[0-9A-F]{4}" }
    | xsd:string { pattern="J14-[0-9A-F]{4}" }
    | xsd:string { pattern="JA[34]?-[0-9A-F]{4}" }
    | xsd:string { pattern="JARIB-[0-9A-F]{4}" }
    | xsd:string { pattern="JH-(JT[ABC][0-9A-F]{3}S?|IB\d{4}|\d{6})" }
    | xsd:string { pattern="JK-\d{5}" }
  }

```

```

    | xsd:string { pattern="JM[-\d{6}]" }
  }?
code-point-attributes &= attribute kIRG_KPSource
{ "" | xsd:string { pattern="KP([01]-[0-9A-F]{4})U-[023][0-9A-F]{4}" } }?
code-point-attributes &= attribute kIRG_KSource
{ "" | xsd:string { pattern="K[0-6]-[0-9A-F]{4}" }
  | xsd:string { pattern="KC-\d{5}" }
  | xsd:string { pattern="KU-[023][0-9A-F]{4}" }
}
code-point-attributes &= attribute kIRG_MSource
{ "" | xsd:string { pattern="MA-[0-9A-F]{4}" }
  | xsd:string { pattern="MB[12]-[0-9A-F]{4}" }
  | xsd:string { pattern="MC-\d{5}" }
  | xsd:string { pattern="MDH?-[23]?[0-9A-F]{4}" }
}
code-point-attributes &= attribute kIRG_SSource
{ "" | xsd:string { pattern="SATM?-\d{5}" } }?
code-point-attributes &= attribute kIRG_TSource
{ "" | xsd:string { pattern="T([1-79A-F]|1[1-3])-[0-9A-F]{4}" }
  | xsd:string { pattern="TU-[023][0-9A-F]{4}" }
}
code-point-attributes &= attribute kIRG_UKSource
{ "" | xsd:string { pattern="UK-\d{5}" } }?
code-point-attributes &= attribute kIRG_USource
{ "" | xsd:string { pattern="UTC-\d{5}" } }?
code-point-attributes &= attribute kIRG_VSource
{ "" | xsd:string { pattern="V[0-4]-[0-9A-F]{4}" }
  | xsd:string { pattern="VN-[023F][0-9A-F]{4}" }
}
code-point-attributes &= attribute kIRGDaeJaweon
{ list { xsd:string { pattern="\d{4}\.\d{2}[01]" } }+ } }?
code-point-attributes &= attribute kIRGHanyuDaZidian
{ list { xsd:string { pattern="[1-8]\d{4}\.[0-3]\d[01]" } }+ } }?
code-point-attributes &= attribute kIRGKangXi
{ list { xsd:string { pattern="[01]\d{3}\.[0-7]\d[01]" } }+ } }?
code-point-attributes &= attribute kJp
{ list { xsd:string { pattern="[0-9A-F]{4}5?" } }+ } }?
code-point-attributes &= attribute kJapanese
{ list { xsd:string { pattern="[x{3041}-\x{3096}]\x{3099}\x{309A}\x{30A1}-\x{30FA}\x{30FC}]" }+ } }?
code-point-attributes &= attribute kJapaneseKun
{ list { xsd:string { pattern="[A-Z]" } }+ } }?
code-point-attributes &= attribute kJapaneseOn
{ list { xsd:string { pattern="[A-Z]" } }+ } }?
code-point-attributes &= attribute kJinmeiyKanji
{ list { xsd:string { pattern="(20\d{2})(:U+[23]?[0-9A-F]{4})?" }+ } }?
code-point-attributes &= attribute kJis0
{ list { xsd:string { pattern="\d{4}" } }+ } }?
code-point-attributes &= attribute kJis1
{ list { xsd:string { pattern="\d{4}" } }+ } }?
code-point-attributes &= attribute kJIS0213
{ list { xsd:string { pattern="[12],\d{2},\d{1,2}" } }+ } }?
code-point-attributes &= attribute kJoyoKanji
{ list { xsd:string { pattern="(20\d{2})|(U+[23]?[0-9A-F]{4})" }+ } }?
code-point-attributes &= attribute kKangXi
{ list { xsd:string { pattern="\d{4}\.\d{2}[01]" } }+ } }?
code-point-attributes &= attribute kKarlGren
{ list { xsd:string { pattern="[1-9]\d{0,3}[A*]" } }+ } }?
code-point-attributes &= attribute kKorean
{ list { xsd:string { pattern="[A-Z]" } }+ } }?
code-point-attributes &= attribute kKoreanEducationHanja
{ list { xsd:string { pattern="20\d{2}" } }+ } }?
code-point-attributes &= attribute kKoreanName
{ list { xsd:string { pattern="20\d{2}" } }+ } }?
code-point-attributes &= attribute kLau
{ list { xsd:string { pattern="[1-9]\d{0,3}" } }+ } }?
code-point-attributes &= attribute kMainlandTelegraph
{ list { xsd:string { pattern="\d{4}" } }+ } }?
code-point-attributes &= attribute kMandarin
{ list { xsd:string { pattern="[a-z]\x{300}-\x{302}\x{304}\x{308}\x{30C}]" }+ } }?
code-point-attributes &= attribute kMatthews
{ list { xsd:string { pattern="[1-9]\d{0,3}(a|\.)5?" } }+ } }?
code-point-attributes &= attribute kMeyerWempe
{ list { xsd:string { pattern="[1-9]\d{0,3}[a-t*]" } }+ } }?
code-point-attributes &= attribute kMojiJoho
{ list { xsd:string { pattern="MJ\d{6}:(FE0[01]|E01[01][0-9A-F])?" } }+ } }?
code-point-attributes &= attribute kMorohashi
{ list { xsd:string { pattern="(\d{5}'{0,2})H\d{3}:(FE0[01]|E010[0-9A-F])?" }+ } }?

```

```

code-point-attributes &= attribute kNelson
  { list { xsd:string { pattern="\d{4}" }+ } }?

code-point-attributes &= attribute kOtherNumeric
  { list { xsd:string { pattern="\d+" }+ } }?

code-point-attributes &= attribute kPhonetic
  { list { xsd:string { pattern="[1-9]\d{0,3}[A-D]?\" }+ } }?

code-point-attributes &= attribute kPrimaryNumeric
  { list { xsd:string { pattern="\d+" }+ } }?

code-point-attributes &= attribute kPseudoGB1
  { xsd:string { pattern="\d{4}" } }?

code-point-attributes &= attribute kRSAdobe_Japan1_6
  { list { xsd:string { pattern="[CV]\+\d{1,5}\+[1-9]\d{0,2}\.[1-9]\d?\.\d{1,2}" }+ } }?

code-point-attributes &= attribute kRSUnicode
  { list { xsd:string { pattern="[1-9]\d{0,2}'{0,3}\.-?\d{1,2}" }+ } }?

code-point-attributes &= attribute kSBGY
  { list { xsd:string { pattern="\d{3}\.[0-7]\d" }+ } }?

code-point-attributes &= attribute kSemanticVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}<([ks][A-Za-z0-9_]+(:[TBZFJ]+)?(,[ks][A-Za-z0-9_]+(:[TBZFJ]+)?)*))\" }+ } }?

code-point-attributes &= attribute kSimplifiedVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}" }+ } }?

code-point-attributes &= attribute kSMSZD2003Index
  { list { xsd:string { pattern="\d{1,3}\.\d{2}" }+ } }?

code-point-attributes &= attribute kSMSZD2003Readings
  { list { xsd:string { pattern="[a-z]\x{300}\x{301}\x{302}\x{304}\x{308}\x{30C}+([a-z]\x{300}\x{301}\x{302}\x{304}\x{308}\x{30C})+\x{7CB5}[a-z]+[1-6]([a-z]+[1-6]

code-point-attributes &= attribute kSpecializedSemanticVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}<([ks][A-Za-z0-9_]+(:[TBZFJ]+)?(,[ks][A-Za-z0-9_]+(:[TBZFJ]+)?)*))\" }+ } }?

code-point-attributes &= attribute kSpoofingVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}" }+ } }?

code-point-attributes &= attribute kStrange
  { list { ( xsd:string { pattern="[ACU]" }
    | xsd:string { pattern="B:U\+31[0-2AB][0-9A-F]" }
    | xsd:string { pattern="[MORV](:\U\+[23]?[0-9A-F]{4})?" }
    | xsd:string { pattern="H:(:\U\+31[3-8][0-9A-F])+" }
    | xsd:string { pattern="I:(:\U\+[23]?[0-9A-F]{4})*" }
    | xsd:string { pattern="K:(:\U\+30[A-F][0-9A-F])+" }
    | xsd:string { pattern="S:[4-9]\d" }
  )+ } }?

code-point-attributes &= attribute kTaiwanTelegraph
  { list { xsd:string { pattern="\d{4}" }+ } }?

code-point-attributes &= attribute kTang
  { list { xsd:string { pattern="*\*[A-Za-z()]\x{E6}\x{251}\x{259}\x{25B}\x{300}\x{30C}+" }+ } }?

code-point-attributes &= attribute kTayNumeric
  { list { xsd:string { pattern="\d+" }+ } }?

code-point-attributes &= attribute kTGH
  { list { xsd:string { pattern="20\d{2}:[1-9]\d{0,3}" }+ } }?

code-point-attributes &= attribute kTGHZ2013
  { list { xsd:string { pattern="\d{3}\.\d{3}(,\d{3}\.\d{3})*:[a-z]\x{300}-\x{302}\x{304}\x{308}\x{30C}+" }+ } }?

code-point-attributes &= attribute kTotalStrokes
  { xsd:string { pattern="[1-9]\d{0,2}" } }?

code-point-attributes &= attribute kTraditionalVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}" }+ } }?

code-point-attributes &= attribute kUnihanCore2020
  { xsd:string { pattern="[GHJKMPT]{1,7}" } }?

code-point-attributes &= attribute kVietnamese
  { list { xsd:string { pattern="[A-Za-z]\x{110}\x{111}\x{300}-\x{303}\x{306}\x{309}\x{31B}\x{323}+" }+ } }?

code-point-attributes &= attribute kVietnameseNumeric
  { list { xsd:string { pattern="\d+" }+ } }?

code-point-attributes &= attribute kXerox
  { list { xsd:string { pattern="\d{3}:\d{3}" }+ } }?

code-point-attributes &= attribute kXHC1983
  { list { xsd:string { pattern="\d{4}\.\d{3}*?(\,\d{4}\.\d{3})*?":[a-z]\x{300}\x{301}\x{304}\x{308}\x{30C}+" }+ } }?

code-point-attributes &= attribute kZhuang
  { list { xsd:string { pattern="[a-z]+\?" }+ } }?

code-point-attributes &= attribute kZhuangNumeric
  { list { xsd:string { pattern="\d+" }+ } }?

code-point-attributes &= attribute kZVariant
  { list { xsd:string { pattern="U\+[23]?[0-9A-F]{4}<([ks][A-Za-z0-9_]+(:[TBZ]+)?(,[ks][A-Za-z0-9_]+(:[TBZ]+)?)*))\" }+ } }?

```

#### 4.4.24 Tangut data

The Tangut data are represented as attributes. The attribute `kRSUnicode` represents the radical stroke index. The attribute `kTGT_MergedSrc` indicates the source reference for the character.

[Tangut data, 51] =

```

code-point-attributes &=
  attribute
    kRSUnicode
    kTGT_RSUnicode { xsd:string { pattern="[0-9]+\.[0-9]+" } }?

```

```

code-point-attributes &=
  attribute kTGT_MergedSrc
  {
    xsd:string { pattern="H2004-[AB]-\d{4}" }
    | xsd:string { pattern="H2021-\d{6}" }
    | xsd:string { pattern="L(19(86|97)|20(06|12))-\d{4}" }
    | xsd:string { pattern="L2008-\d{4}([AB])-\d{4}" }
    | xsd:string { pattern="N1966-\d{3}-\d{2}[0-9A-Z]{1,2}" }
    | xsd:string { pattern="N5217-\d{2}" }
    | xsd:string { pattern="S1968-\d{4}" }
    | xsd:string { pattern="UTN42-\d{3}" }
  }?

```

#### 4.4.25 Nushu data

The Nushu data are represented as attributes. The attribute `kSrc_NushuDuben` indicates the page number and order of the item from the NushuDuben reference source. Nushu common reading is represented as `kReading`.

[Nushu data, 52] =

```

code-point-attributes &=
  attribute
    kSrc_NushuDuben
    kNSHU_DubenSrc { xsd:string { pattern="[0-9]+\.[0-9]+" }}?

code-point-attributes &=
  attribute
    kReading
    kNSHU_Reading { xsd:string }?

```

#### 4.4.26 Emoji properties

The properties Emoji, EPres, EMod, EBase, EComp, and ExtPict have corresponding attributes:

[Emoji properties, 53] =

```

code-point-attributes &=
  attribute Emoji { boolean }?

code-point-attributes &=
  attribute EPres { boolean }?

code-point-attributes &=
  attribute EMod { boolean }?

code-point-attributes &=
  attribute EBase { boolean }?

code-point-attributes &=
  attribute EComp { boolean }?

code-point-attributes &=
  attribute ExtPict { boolean }?

```

#### 4.4.27 Unikemet properties

The Unikemet data are represented as attributes. The attribute `kEH_Cat` is a catalog entry corresponding to the IFAO-based taxonomy. The attribute `kEH_Core` determines whether an Egyptian hieroglyph is part of the 'Core' set. Legacy or None. The attribute `kEH_Desc` provides a detailed description of the appearance of the hieroglyph. The attribute `kEH_Func` represents a function type representing a pictogram, a logogram, a phonemogram (or "phonogram"), a classifier (or "determinative"), a phono-repeater (sub-category of classifier), a radicogram or interpretant. The attribute `kEH_FVal` expresses the function type using the Gardiner 1957 convention for Egyptian hieroglyph transliteration. The attribute `kEH_HG` indicates the Hieroglyphica source. The attribute `kEH_IFAO` indicates the IFAO source value defined as page number and order in that page. The attribute `kEH_JSesh` indicates the JSesh source as specified in JSesh Documentation. The attribute `kEH_NoMirror` determines whether the hieroglyph does not mirror. The attribute `kEH_NoRotate` determines whether the hieroglyph does not rotate. The attribute `kEH_Unik` represent the original Unikemet catalog index used by the Egyptian Hieroglyph block.

[Unikemet data, 54] =

```

code-point-attributes &= attribute kEH_Cat
  { xsd:string { pattern="([A-IK-Z][AA]-\d{2})-\d{3}" }}?

code-point-attributes &=
  attribute kEH_Core { "C" | "L" | "N" }?

code-point-attributes &= attribute kEH_Desc
  { xsd:string { pattern="([\t]+)" }}?

code-point-attributes &= attribute kEH_Func
  { list { "/" } xsd:string { pattern="([\t]+)" }}?

code-point-attributes &= attribute kEH_FVal
  { list { "" | xsd:string { pattern="[bdf-hjkmnpr-twy\.,/\-!+=;>&()\{\}\s\x{303}\x{30C}\x{323}\x{32E}\x{331}\x{A723}\x{A725}\x{A7BD}]" }} }?

code-point-attributes &= attribute kEH_Unik
  { xsd:string { pattern="([A-IK-Z][AA|NL|NU])\d{3}[A-Z]{0,2}" }
  | xsd:string { pattern="HJ([A-IK-Z][AA])\d{3}[A-Z]{0,2}" }
  }?

code-point-attributes &= attribute kEH_JSesh
  { list { ( xsd:string { pattern="([A-IK-Z][Aa|NL|NU|Ff])\d{1,3}[A-Za-z]{0,5}" }
  | xsd:string { pattern="(US1|US22|US248|US685)([A-IK-Z][Aa|NL|NU])\d{1,3}[A-Za-z]{0,5}" }
  )+ }?

code-point-attributes &= attribute kEH_HG
  { list { xsd:string { pattern="([A-IK-Z][AA])\d{1,3}[A-Za-z]{0,2}" }+ }?

code-point-attributes &= attribute kEH_IFAO
  { list { xsd:string { pattern="\d{1,3},\d{1,2}[ab]?" }+ }?

code-point-attributes &=
  attribute kEH_NoMirror { boolean }?

code-point-attributes &=
  attribute kEH_NoRotate { boolean }?

code-point-attributes &= attribute kEH_AltSeq
  { xsd:string { pattern="[0-9A-F]{5}(\s[0-9A-F]{4,5})*" }}?

```

## 5 Blocks

The `blocks` child of the `ucd` describes the blocks. It has one child `block` element per block, with attributes to describe the extent and name of the block.

```
[blocks, 55] =
ucd.content &=
  element blocks {
    element block {
      attribute first-cp { single-code-point },
      attribute last-cp { single-code-point },
      attribute name { text } }+ }?
```

## 6 Named Sequences

The `named-sequences` child of the `ucd` describes the named sequences. It has one child `named-sequence` element per named sequence, with attributes to describe the name and sequence.

Similarly, the `provisional-named-sequences` child of the `ucd` describes the provisional named sequences.

```
[named sequences, 56] =
ucd.content &=
  element named-sequences {
    element named-sequence {
      attribute cps { one-or-more-code-points },
      attribute name { text } }+ }?

ucd.content &=
  element provisional-named-sequences {
    element named-sequence {
      attribute cps { one-or-more-code-points },
      attribute name { text } }+ }?
```

## 7 Normalization Corrections

The `normalization-corrections` child of the `ucd` describes the normalization corrections. It has one child `normalization-correction` element per correction, with attributes to describe the code point affected, its old normalization, its new normalization and the version of Unicode in which the correction was made.

```
[normalization corrections, 57] =
ucd.content &=
  element normalization-corrections {
    element normalization-correction {
      attribute cp { single-code-point },
      attribute old { one-or-more-code-points },
      attribute new { one-or-more-code-points },
      attribute version { text } }+ }?
```

## 8 Standardized Variants

The `standardized-variants` child of the `ucd` describes the standardized variant. It has one child `standardized-variant` element per variant. The attributes on that last element capture the variation sequence, the description of the desired appearance, and the shaping environment under which the appearance is different.

```
[standardized variants, 58] =
ucd.content &=
  element standardized-variants {
    element standardized-variant {
      attribute cps { two-code-points },
      attribute desc { text },
      attribute when { text } }+ }?
```

## 9 CJK Radicals

The `CJK-radicals` child of the `ucd` describes the CJK radicals. It has one child `CJK-radical` element per radical. The attributes on that last element capture the radical number, the corresponding CJK radical character, and the corresponding CJK unified ideograph.

```
[CJK radicals, 59] =
ucd.content &=
  element CJK-radicals {
    element CJK-radical {
      attribute number { xsd:string { pattern="[0-9]{1,3}'{0,3}" } },
      attribute radical { single-code-point? },
      attribute ideograph { single-code-point } }+ }?
```

## 10 Emoji sources

The `emoji-sources` child of the `ucd` describes the emoji sources.

```
[emoji sources, 60] =
ucd.content &=
  element emoji-sources {
    element emoji-source {
      attribute unicode { one-or-more-code-points },
      attribute docomo { jis-code-point? },
      attribute kddi { jis-code-point? },
      attribute softbank { jis-code-point? } }+ }?
```

```
[datatype for code points, 61] =
jis-code-point xsd:string { pattern="[0-9A-F]{4}" }
```

## 11 Do Not Emit

The `do-not-emit` child of the `ucd` describes the character sequences that should not be emitted or generated in newly authored texts.

```
[do-not-emit, 62] =
ucd.content &=
  element do-not-emit {
    element instead {
      attribute of { one-or-more-code-points },
```

```

attribute use { one-or-more-code-points },
attribute because {
  "Arabic_Tashkil"
  "Bengali_Khanda_Ta"
  "Deprecated"
  "Discouraged"
  "Dotless_Form"
  "Hamza_Form"
  "Indic_Atomic_Consonant"
  "Indic_Consonant_Conjunct"
  "Indic_Vowel_Letter"
  "Malayalam_Chillu"
  "None"
  "Precomposed_Form"
  "Precomposed_Hieroglyph"
  "Preferred_Spelling"
  "Tamil_Shrii"
} }+ }?

```

## 12 The full schema

Our schema is just the accumulation of the pieces we have described so far:

```

[UCD RelaxNG schema] =
[namespace declaration: 1]
[datatypes: 2, 3, 61]
[schema start: 4]
[boolean: 5]
[description: 6]
[repertoire: 7, 8, 9, 10]
[attributes: 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50]
[Tangut data: 51]
[Nushu data: 52]
[Unikemet data: 54]
[blocks: 55]
[named sequences: 56]
[normalization corrections: 57]
[standardized variants: 58]
[cjk radicals: 59]
[emoji sources: 60]
[emoji properties: 53]
[do-not-emit: 62]

```

An expanded version is linked from the top of this document.

## 13 Examples

Here is a fragment of the UCD for a few representative characters (only some of the properties are represented):

```

<ucd xmlns="http://www.unicode.org/ns/2003/ucd/1.0">
  <repertoire>
    <char cp="001F" age="1.1" na="&lt;control&gt;" na1="UNIT SEPARATOR"
      gc="Cc" bc="S" lb="CM"/>

    <char cp="0020" age="1.1" na="SPACE" gc="Zs" bc="WS" ea="Na" lb="SP"/>

    <char cp="0026" age="1.1" na="AMPERSAND" gc="Po" bc="ON" ea="Na"/>

    <char cp="0028" age="1.1" na="LEFT PARENTHESIS" na1="OPENING PARENTHESIS"
      gc="Ps" bc="ON" Bidl_M="y" bmg="0029" ea="Na" lb="OP"/>

    <char cp="0041" age="1.1" na="LATIN CAPITAL LETTER A"
      gc="Lu" slc="0061" ea="Na" sc="Latn"/>

    <char cp="AC00" age="2.0" na="HANGUL SYLLABLE GA" gc="Lo"
      dt="can" dm="1100 1161" ea="W" lb="ID" sc="Hang"/>

    <char cp="20094" age="3.1" na="CJK UNIFIED IDEOGRAPH-20094"
      gc="Lo" ea="W" lb="ID" sc="Hani" kIRG_GSource="KX"
      kIRGHanyuDaZidian="10036.060" kIRG_TSource="5-214E"
      kRSUnicode="4.3" kIRGKangXi="0082.090"/>

    <group age="3.2" gc="Lo" sc="Buhd">
      <char cp="1740" na="BUHID LETTER A"/>
      <char cp="1741" na="BUHID LETTER I"/>
      <char cp="1752" na="BUHID VOWEL SIGN I" gc="Mn"/>
      <char cp="1820" age="3.0" na="MONGOLIAN LETTER A" sc="Mong"/>
    </group>
  </repertoire>
</ucd>

```

## Acknowledgments

Thanks to Markus Scherer and Mark Davis for their help developing this XML representation. Thanks to the reviewers: Julie Allen, Ernest van den Boogaard, Daniel Bünzli, John Cowan, Asmus Freytag, Felix Sasaki, Andrew West. Special thanks to Eric Muller and Laurențiu Iancu.

## Modifications

This section indicates the changes introduced by each revision.

### Revision 37

- New value for the age attribute: 17.0.
- New values for the blk attribute: Beria\_Erfe, CJK\_Ext\_J, Misc\_Symbols\_Sup, Sharada\_Sup, Sidetic, Tai\_Yo, Tangut\_Components\_Sup, ToLong\_Siki.
- New value for the do-not-emit attribute: None.
- New values for the jg attribute: Thin\_Noon, BAA, FA, HAA, HA\_GOAL, HA, CAF, KNOTTED\_HA, RA, SWASH\_CAF, HAMZAH\_ON\_HA\_GOAL, TAA\_MARBUTAH, YA\_BARREE, YA, ALEF\_MAQSURAH.

- New value for the `lb` attribute: `HH`.
- New value for the `InPC` attribute: `Invisible`.
- New value for the `InSC` attribute: `Consonant_Repha`.
- New values for the `script` attribute: `Berf`, `Sidt`, `Tayo`, `Tols`.
- New code point attributes for Unikemet: `KEH_AltSeq`, `KEH_Cat`, `KEH_Core`, `KEH_Desc`, `KEH_Func`, `KEH_FVal`, `KEH_HG`, `KEH_IFAO`, `KEH_JSesh`, `KEH_NoMirror`, `KEH_NoRotate`, `KEH_UniK`.
- New attribute for the `kTayNumeric` property.
- Removed attributes for deprecated properties: `Gr_Link`, `Hyphen`, `isc`, `kGB7`, `kJa`, `XO_NFC`, `XO_NFD`, `XO_NFKC`, `XO_NFKD`, `FC_NFKC`.
- Removed elements that only contained historical information: `normalization-corrections`, `emoji-sources`.
- Unihan attributes are applied at the group where applicable, similar to how non-Unihan attributes are applied at the group.

**Revision 36**

- New value for the `age` attribute: `16.0`.
- New values for the `blk` attribute: `Egyptian_Hieroglyphs_Ext_A`, `Garay`, `Gurung_Khema`, `Kirat_Rai`, `Myanmar_Ext_C`, `O1_Onal`, `Sunuwar`, `Symbols_for_Legacy_Computing_Sup`, `Todhri`, `Tulu_Tigalari`.
- New values for the `script` attribute: `Gara`, `Gukh`, `Krai`, `Onao`, `Sunu`, `Todr`, `Tutg`.
- New value for the `lg` attribute: `Kashmiri_Yeh`.
- New value for the `InSC` attribute: `Reordering_Killer`.
- New attributes: `MCM`, `kFanqie`, `kZhuang`.
- Modified patterns for the `ckj-radical/@number`, `kRSUnicode` and `kIRG_GSource` attributes.
- Added the `do-not-emit` element.

Revision 35 being a proposed update, only changes between revisions 34 and 36 are noted here.

**Revision 34**

- New value for the `age` attribute: `15.1`.
- New value for the `blk` attribute: `CJK_Ext_I`.
- New values for the `lb` attribute: `AK`, `AP`, `A5`, `VF`, `VI`.
- Modified values for the `number`, `radical` attributes of the `ckj-radical` element.
- Changed single value into list for the `nv` code point attribute.
- New code point attributes: `ID_Compat_Math_Continue`, `ID_Compat_Math_Start`, `IDSU`, `NFKC_SCF`, `InCB`.
- Modified patterns for the `kBigFive`, `kIRG_GSource`, `kMorohashi`, `kRSUnicode` attributes.
- Changed single values into lists for the `kMorohashi`, `kPrimaryNumeric` Unihan attributes.
- New Unihan attributes: `kJapanese`, `kMojjiJoho`, `kSMSZD2003Index`, `kSMSZD2003Readings`, `kVietnameseNumeric`, `kZhuangNumeric`.

Revision 33 being a proposed update, only changes between revisions 32 and 34 are noted here.

**Revision 32**

- New value for the `age` attribute: `15.0`.
- New values for the `blk` attribute: `Arabic_Ext_C`, `CJK_Ext_H`, `Cyrillic_Ext_D`, `Devanagari_Ext_A`, `Kaktovik_Numerals`, `Kawi`, `Nag_Mundari`.
- New values for the `script` attribute: `Kawi`, `Nagm`.
- New Unihan attribute: `kAlternateTotalStrokes`.
- Modified patterns for the `kIRG_GSource`, `kIRG_HSource`, `kIRG_TSource`, `kSemanticVariant`, `kSpecializedSemanticVariant`, `kZVariant` attributes.

Revision 31 being a proposed update, only changes between revisions 30 and 32 are noted here.

**Revision 30**

- New value for the `age` attribute: `14.0`.
- New values for the `blk` attribute: `Arabic_Ext_B`, `Cypro_Minoan`, `Ethiopic_Ext_B`, `Kana_Ext_B`, `Latin_Ext_F`, `Latin_Ext_G`, `Old_Uyghur`, `Tangsa`, `Toto`, `UCAS_Ext_A`, `Vithkuqi`, `Znamenny_Music`.
- New values for the `script` attribute: `Cpmn`, `Ougr`, `Tnsa`, `Toto`, `Vith`.
- New values for the `lg` attribute: `Thin_Yeh`, `Vertical_Tail`.
- New Unihan attribute: `kStrange`.
- Modified patterns for the `kIRG_GSource`, `kIRG_MSource`, `kIRG_VSource`, `kPhonetic`, `kSpoofingVariant` attributes.
- Removal of the `kwubi` attribute, which has never been present in released versions of the UCD.

Revision 29 being a proposed update, only changes between revisions 28 and 30 are noted here.

**Revision 28**

- New value for the `age` attribute: `13.0`.
- New values for the `blk` attribute: `Chorasmian`, `CJK_Ext_G`, `Dives_Akuru`, `Khitan_Small_Script`, `Lisu_Sup`, `Symbols_For_Legacy_Computing`, `Tangut_Sup`, `Yezidi`.
- New values for the `script` attribute: `Chrs`, `Diak`, `Kits`, `Yezi`.
- New value for the `InPC` attribute: `Top_And_Bottom_And_Left`.
- New Unihan attributes `kSpoofingVariant`, `kUnihanCore2020`, `kIRG_SSource`, `kIRG_UKSource`, `kTGHZ2013`.
- New Emoji attributes `Emoji`, `EPres`, `EMod`, `EBase`, `EComp`, `ExtPict`.
- Modified patterns for the `kIRG_GSource`, `kIRG_HSource`, `kIRG_KPSource`, `kIRG_KSource`, `kIRG_TSource`, `kKangXi`, `kSemanticVariant`, `kSimplifiedVariant`, `kSpecializedSemanticVariant`, `kTraditionalVariant` attributes.

Revision 27 being a proposed update, only changes between revisions 26 and 28 are noted here.

**Revision 26**

- New value for the `age` attribute: `12.1`.

**Revision 25**

- New value for the `age` attribute: `12.0`.
- New values for the `script` attribute: `Elym`, `Hmnp`, `Nand`, `Wcho`.

- New values for the `blk` attribute: Egyptian\_Hieroglyph\_Format\_Controls, Elymaic, Nandinagari, Nyiakeng\_Puachue\_Hmong, Ottoman\_Siyaq\_Numbers, Small\_Kana\_Ext, Symbols\_And\_Pictographs\_Ext\_A, Tamil\_Sup, Wancho.
- Modified patterns for the `kIRG_GSource`, `kIRG_KSource`, `kIRG_TSource`, `kTaiwanTelegraph` attributes.

Revision 24 being a proposed update, only changes between revisions 23 and 25 are noted here.

#### Revision 23

- New value for the `age` attribute: 11.0.
- New values for the `blk` attribute: Chess\_Symbols, Dogra, Georgian\_Ext, Gunjala\_Gondi, Hanifi\_Rohingya, Indic\_Siyaq\_Numbers, Makasar, Mayan\_Numerals, Medefaidrin, Old\_Sogdian, Sogdian.
- New values for the `script` attribute: Dogr, Gong, Maka, Medf, Rohg, Sogd, Sogo.
- New values for the `kg` attribute: Hanifi\_Rohingya\_Kinna\_Ya, Hanifi\_Rohingya\_Pa.
- New value for the `wb` attribute: wSegSpace.
- New values for the `InSC` attribute: Consonant\_Initial\_Postfixed.
- New attributes: EqUIdeo, kJimeiyokanji, kJoyoKanji, kKoreanEducationHanja, kKoreanName, kTGH.
- Modified patterns for the `KTGT_MergedSrc` attribute.
- Modified patterns for the `kIRG_GSource`, `kIRG_HSource` and `kIRG_VSource` attributes.

Revision 22 being a proposed update, only changes between revisions 21 and 23 are noted here.

#### Revision 21

- New value for the `age` attribute: 10.0.
- New values for the `blk` attribute: CJK\_Ext\_F, Kana\_Ext\_A, Masaram\_Gondi, Nushu, Soyombo, Syriac\_Sup, Zanabazar\_Square.
- New values for the `sc` attribute: Gonn, Nshu, Soyo, Zamb.
- New values for the `kg` attribute: Malayalam\_Nga, Malayalam\_Ja, Malayalam\_Nya, Malayalam\_Tta, Malayalam\_Nna, Malayalam\_Nnna, Malayalam\_Bha, Malayalam\_Ra, Malayalam\_Lla, Malayalam\_Llla, Malayalam\_Ssa.
- New value for the `InPC` attribute: Bottom\_And\_Left.
- Modified patterns for the `kIRG_GSource`, `kIRG_JSource`, `kIRG_KSource` attributes.
- New code point attributes: vo, RI
- New code point attributes for Nushu data: kSrc\_NushuDuben and kReading.

Revision 20 being a proposed update, only changes between revisions 19 and 21 are noted here.

#### Revision 19

- New value for the `age` attribute: 9.0.
- New values for the `sc` attribute: Adlm, Bhks, Marc, Nawa, Osge, Tang.
- New values for the `blk` attribute: Adlam, Bhaiksuki, Cyrillic\_Ext\_C, Glagolitic\_Sup, Ideographic\_Symbols, Marchen, Mongolian\_Sup, Nawa, Osage, Tangut, Tangut\_Components.
- New values for the `gcb` attribute: EB, EBG, EM, GAZ, ZWJ.
- New values for the `wb` attribute: EB, EBG, EM, GAZ, ZWJ.
- New values for the `lb` attribute: EB, EM, ZWJ.
- New values for the `kg` attribute: African\_Feh, African\_Noon, African\_Qaf.
- New code point attributes: PCM, kRSTUnicode and kTGT\_MergedSrc.
- Modified patterns for the `kRSUnicode`, `kRSKangXi`, `kMandarin`, `kIRG_JSource`, `kIRG_USource` and `kFennIndex` attributes.

Revision 18 being a proposed update, only changes between revisions 17 and 19 are noted here.

#### Revision 17

- New value for the `age` attribute: 8.0.
- New values for the `sc` attribute: Ahom, Hatr, Hluw, Hung, Mult, Sgnw.
- New values for the `blk` attribute: Ahom, Anatolian\_Hieroglyphs, Cherokee\_Sup, CJK\_Ext\_E, Early\_Dynastic\_Cuneiform, Hatran, Multani, Old\_Hungarian, Sup\_Symbols\_And\_Pictographs, Sutton\_SignWriting.
- New values for the `InSC` attribute: Consonant\_Killer, Consonant\_Prefixed, Consonant\_With\_Stacker, Syllable\_Modifier.
- New code point attributes: InPC, kJa.
- New patterns for the `kIRG_GSource` attribute: GFC-, GGFZ-.
- Switched the reference to ISO 19757 from :2003 and :2003 Amd1 to :2008.

Revision 16 being a proposed update, only changes between revisions 15 and 17 are noted here.

#### Revision 15

- New value for the `age` attribute: 7.0.
- New values for the `kg` attribute.
- New values for the `sc` attribute.
- New values for the `blk` attribute.
- New values for the `InSC` attribute.
- New values for the `kIICore` attribute.
- New values for the `kIRG_GSource` attribute.

Revision 14 being a proposed update, only changes between revisions 13 and 15 are noted here.

#### Revision 13

- New value for the `age` attribute: 6.3.
- New values `DQ`, `HL`, `SQ` for the `wb` attribute(forUnicode6.3).
- New code point attributes `bpt` and `bpb` (for Unicode 6.3).
- New values for the `bc` attribute: LRI, RLI, FSI, PDI (for Unicode 6.3).
- Updated the patterns for `kHanyuPinlu` and `kTotalStrokes` (for Unicode6.3).
- Updated the patterns for `kIRG_HSource` and `kIRG_VSource` (for Unicode6.2).

- Clarified that the child elements list-like elements are in no particular order.

Revision 12 being a proposed update, only changes between revisions 11 and 13 are noted here.

#### Revision 11

- New value for the `age` attribute: 6.2.
- New value for the `gcb`, `wb` and `lb` attributes: RI (for Unicode 6.2).
- Updated the patterns for `kIRG_GSource` and `kIRG_HSource` (for Unicode 6.2).

Revision 10 being a proposed update, only changes between revisions 9 and 11 are noted here.

#### Revision 9

- Clarified the default values.
- Indicate that property values may change from one release to the next.
- Introduced the `blk` attributes, for the Block property.
- Introduced the `scx` attribute, for the ScriptExtensions property.
- Introduced the `name-alias` element, for the Name\_Alias property.
- New value for the `age` attribute: 6.1.
- New values for the `script` attribute: Cakm, Merc, Mero, Plrd, Shrd, Sora, Takr.
- New values for the `lb` attribute: HL and CJ.
- New value for the `jg` attribute: Rohingya\_Yeh.
- The value of the `fc_nfkC` attribute must now be either # or one-or-more-code-points.
- For the `nv` attribute, the absence of a numeric value is now represented by NaN rather than by the empty string.
- The values of the `ccc` are now restricted to 0..254, instead of 0..255.
- Updated the patterns for `kSemanticVariant`, `kSpecializedSemanticVariant`, `kIRG_USource`, and `kMandarin`.

Revision 8 being a proposed update, only changes between revisions 7 and 9 are noted here.

#### Revision 7

- New value for the `age` attribute: 6.0.
- New value for the `jg` attribute: Teh\_Marbuta\_Goal
- New values for the `script` attribute: Batk, Brah, Mand.
- Updated the patterns for `kIRG_GSource`, `kIRG_HSource`, `kIRG_JSource`, `kIRG_KSource`, `kIRG_MSource`, `kIRG_TSource`, `kIRG_VSource`.
- Added the `InSc` and `InMc` elements.
- Added the `emoji-sources` element.

Revision 6 being a proposed update, only changes between revisions 5 and 7 are noted here.

#### Revision 5

- Changed the type of `block/@first-cp`, `block/@last-cp` and `normalization-corrections/@cp` from text to single-code-point
- Changed the type of `named-sequence/@cps`, `provisional-named-sequences/@cps`, `normalization-correction/@old` and `normalization-correction/@new` from text to one-or-more-code-points.
- Changed the type of `standardized-variants/@cps` from text to two-code-points.
- New values for the `jg` attribute: Farsi\_Yeh and Nya.
- New value for the `age` attribute: 5.2.
- New values for the `sc` attribute: Lana, Tavt, Avst, Egyp, Samr, Lisu, Bamu, Java, Mtei, Armi, Sarb, Prti, Phli, Orkh, Kthi.
- New value for the `lb` attribute: CP.
- New value for the `sc` attribute: Zinh.
- New code point attributes CI, Cased, CWCF, CWCM, CWL, CWKCF, CWT, CWU, NFKC\_CF.
- New attributes `kHanyuPinyin` and `kIRG_MSource`.
- New element `cjk-radicals`
- Updated the patterns for `kIRG_GSource`, `kIRG_JSource`, `kIRG_KPSource`, `kIRG_KSource`, `kIRG_TSource`, `kIRG_VSource`, `kHanyuPinlu`, `kMandarin`, `kSemanticVariant`, `kSpecializedSemanticVariant`, `kVietnamese`, `kZVariant`.
- Point out that Relax NG schemas do not modify or augment the infoset, and that it is possible to convert mechanically our schema to other schema languages.

Revision 4 being a proposed update, only changes between revisions 3 and 5 are noted here.

#### Revision 3

- First approved version, for Unicode 5.1.0.
- For optional elements which acts as collections, such as `repertoire` and `named-sequences`, impose that there be at least one element in the collection.
- Remove the constraint that the value `jg` is limited when `jt` has certain values; similarly for `bmg / Bidi_M` and for `nv / nt`.
- Value NL added to the `wb` attribute (for Unicode 5.1).
- Value PP added to the `gcb` attribute (for Unicode 5.1).
- Corrected the `vai` script value to `vaii`.
- Removed the discussion of elements or attributes in different namespaces.
- Removed the `code-point` element.

#### Revision 2

- Promoted to Draft UAX.
- Changed the title from "An XML representation of the UCD"
- Value 5.1 added to the `age` attribute (for Unicode 5.1).
- Value SM added to the `gcb` attribute (for Unicode 5.1).
- Values CR, Extend, LF, MB added to the `wb` attribute (for Unicode 5.1).
- Values CR, EX, LF, SC added to the `sb` attribute (for Unicode 5.1).
- Value Burushaski\_Yeh\_Barree added to the `jg` attribute (for Unicode 5.1).

- Value `Alef_Maqsurah` added to the `jg` attribute (for Unicode 2.x).
- Values `Cari`, `Cham`, `Kali`, `Lepc`, `Lyci`, `Lydi`, `Oick`, `Rjng`, `Saur`, `Sund` and `vai` added to the `sc` attribute (for Unicode 5.0).
- `jamo` attribute renamed to `JSN`
- `sfc` attribute renamed to `scf`
- Attribute `kXHC1983` added (for Unicode 5.1.0).
- Pattern for attribute `kIRG_USource` extended (for Unicode 5.1.0).
- Element `provisional-named-sequences` added (for Unicode 5.0)

#### Revision 1

- First working draft.

---

© 2008–2025 Unicode, Inc. This publication is protected by copyright, and permission must be obtained from Unicode, Inc. prior to any reproduction, modification, or other use not permitted by the [Terms of Use](#). Specifically, you may make copies of this publication and may annotate and translate it solely for personal or internal business purposes and not for public distribution, provided that any such permitted copies and modifications fully reproduce all copyright and other legal notices contained in the original. You may not make copies of or modifications to this publication for public distribution, or incorporate it in whole or in part into any product or publication without the express written permission of Unicode.

Use of all Unicode Products, including this publication, is governed by the Unicode [Terms of Use](#). The authors, contributors, and publishers have taken care in the preparation of this publication, but make no express or implied representation or warranty of any kind and assume no responsibility or liability for errors or omissions or for consequential or incidental damages that may arise therefrom. This publication is provided "AS-IS" without charge as a convenience to users.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc., in the United States and other countries.