Citation Style Language Syntax Specification

version 0.9

July, 2009

Editors

- Bruce D'Arcus

Contributors

- Frank Bennett
- Simon Kornblith
- Julian Onions
- Elena Razlogova
- Andrea Rossato
- Rintze Zelle

Introduction

The Citation Style Language (CSL) is an open XML format to describe citation and bibiographic formatting. It is designed to be:

- language-and-application-independent
- easy-to-use and compact
- feature rich
- international-friendly
- easy-to-distribute and update styles

Style Layout

The CSL element structure is namespaced:

namespace
: `http://purl.org/net/xbiblio/csl`

recommended prefix
: `cs`

All CSL styles share the same basic structure: only five different XML elements can be nested directly in the `cs:style` root element: `cs:info`, `cs:citation`, `cs:bibliography`, `cs:macro` and `cs:terms`. The roles of each of these elements (described in more detail below) are:

`cs:info`
: contains metadata describing the style (name of the style, authors of the style, etc)

`cs:citation`
: describes how in-**text** citations should be formatted

`cs:bibliography`
: describes how bibliographies should be formatted

`cs:macro`
: allows for reuse of formatting instructions, allowing for more compact styles

`cs:terms`
: allows for the modification of locale-specific strings (e.g. "edited by" can be changed in "ed. by")

Independent and Dependent Styles

Two main types of CSL styles exist: independent and dependent styles.

An independent style contains a full style description, and includes at least the `cs:info` and `cs:citation` element. Unless it is a note-based style that lacks a bibliography, it also includes the `cs:bibliography` element. The `cs:terms` element and one or more `cs:macro` elements are optional in independent styles.

A dependent style, on the other hand, merely refers to an independent style, like an alias or shortcut. It only includes the `cs:info` element. Dependent styles are used if multiple publications share a single style format. Each publication can thus have its own dependent style (with the info section describing the journal's metadata, e.g. the journal's name or **ISSNs**) with a corresponding entry in (for instance) a public style repository, while only a single independent master style has to be maintained.

Note that dependent styles cannot be used to indicate changes compared to the master style. If there is any difference in formatting between two styles, however small, two separate CSL styles have to be created.

Preamble

Before the style element, each CSL style should include the XML declaration element, specifying the version of XML used as well as the character encoding. The "style" element itself carries a number of arguments:

`xmlns`
: the namespace declaration that binds the elements in the style to the given namespace URI

`class`
: with two possible **values**, "in-**text**" or "note", this specifies whether the style is note-based or uses in-**text** citations coupled to a bibliography

`xml:lang` (optional)
: specifies the locale used for argument **values** within the style

`default-locale` (optional)
: sets the localization of the style output

An example of a preamble is shown below. For most styles only the **value** of "class" and "default-locale" will differ.

```
<?xml version="1.0" encoding="UTF-8"?>
<style xmlns="http://purl.org/net/xbiblio/csl" class="in-text"
xml:lang="en" default-locale="fr-FR">
```

Info

The `cs:info` section of a CSL style contains the style metadata, which does not affect the formatting of citations. Instead, the metadata makes it possible to host styles in style repositories, to allow users to subscribe to field-specific style collections, and to automatically update styles. An example of a filled-in "info" section is shown below, and is followed by a description of all possible elements.

```
<info>
 <title>My first style</title>
 <id>http://www.zotero.org/styles/my-style-name</id>
 <author>
  <name>My name</name>
  <email>[hidden email]</email>
  <uri>http://wherever.com/</uri>
 </author>
 <category term="author-date"/>
 <category term="zoology"/>
 <updated>2008-10-29T21:01:24+00:00</updated>
 <summary>My great new style format.</summary>
 <rights>This work is licensed under a Creative Commons
         Attribution-Share Alike 3.0 Unported License
         http://creativecommons.org/licenses/by-sa/3.0/</rights>
</info>
```

Many elements available in the info section are borrowed from the Atom Syndication Format:

`cs:id`
: this required field should preferably be a valid, stable, and unique URL if the style is to be made publicly available. This identifier establishes the identity of the style, and so can be used by applications to handle automatic updating, and so forth..

`cs:title`
: name of the style (required). The title is shown in the Zotero Style Repository.

`cs:author/cs:contributor/cs:translator`
: people who write a new style, or make significant changes can claim authorship (see example above). For smaller changes the contributor role can be used. In both cases one should supply a name. An email-address and URI are optional.

`cs:updated`
: the contents of this required element is used to assess whether the style has changed since the last time it has been accessed or cached. The syntax of the timestamp is described here.

`cs:published`
: similar to updated, this element contains a timestamp, in this case the timepoint when the style was initially created or first made available.

`cs:category`
: styles can be divided in a number of categories. This information can be used to ease browsing of large style repositories and could allow users to subscribe to styles within particular content areas. The different types of categories are:

- the style's class, which describes how in-**text** citations are rendered:
    - `author-date`: e.g. "... (Doe, 1999)"
    - `numeric`: e.g. "... [1]"
    - `label`: " ... [doe99]."
    - `note`: the citation appears as a footnote or endnote
    - `in-`**`text`**: the full citation appears in-line
- the field(s) the style applies to (the "generic-base" category is meant for generic styles like Harvard and APA): "anthropology", "astronomy", "biology", "botany", "chemistry", "communications", "engineering", "generic-base", "geography", "geology", "history", "humanities", "law", "literature", "math", "medicine", "philosophy", "physics", "psychology", "sociology", "science", "political_science", "social_science", "theology", "zoology"

`cs:rights`
: a license dictating how the style file may be modified and distributed by others. See, e.g. the http://creativecommons.org/license/

`cs:`**`issn`**
: a style written for a specific journal can include the journal's **ISSN** (International Standard Serial Number). N.B. currently Zotero only supports a single **ISSN**. There are plans to add a "**issnl**" element to allow for inclusion of the **ISSN**-L, and also to allow for multiple **ISSNs** (as many journals have both a print and online **ISSN**).

Citation

The `cs:citation` construct is a key part of the style, and describes how in-line citations should be formatted. Sometimes a citation will only be a simple number, in other cases a more elaborate citation is desired, as is the case for author-date styles. The basic structure of the `cs:citation` construct is as follows:

```
<citation>
  <option />
  <sort>
    sort keys
  </sort>
  <layout>
    rendering elements
  </layout>
</citation>
```

The `cs:layout` specifies what information should be included in the citation. Additional control is possible with a range of options, and by setting sorting behavior (both will be discussed later).

Bibliography

This is the second of the key parts, where the bibliography is formatted. It is very similar to the citations section.

```
<bibliography>
  <option .../>
  <layout>
    ...
  </layout>
</bibliography>
```

Again, a set of options to control some of the layout, then the layout itself.

Macros

A list of macro definitions is usually included between the `cs:info` and `cs:citation` sections. These are sort of like subroutines that can be called later in the description to make similar styles for parts. **Effective use of macros is a key to making good styles**. Ideally, in fact, the main layout sections for the citation and bibliography should be quite simple, and simply call a series of macros.

An example macro is

```
<macro name="editor-translator">
  <names variable="editor translator" prefix="(" suffix=")"
delimiter=", ">
    <name and="symbol" initialize-with=". " delimiter=", "/>
    <label form="short" prefix=", " text-transform="capitalize"
suffix="."/>
  </names>
</macro>
```

It is particularly crucial in author-date styles that rely on author names for sorting that one create a macro that can handle a wide variety of cases, including resources that do no include listed authors. Example:

```
<macro name="author">
  <names variable="author">
    <name name-as-sort-order="all"
          and="symbol"
          sort-separator=", "
          initialize-with=". "
          delimiter=", "
          delimiter-precedes-last="always"/>
    <label form="short" prefix=" (" suffix=".)" text-
transform="capitalize"/>
    <substitute>
      <names variable="editor"/>
      <names variable="translator"/>
      <text macro="title"/>
    </substitute>
  </names>
</macro>
```

This example includes the logic that allows the formatter to gracefully adapt to a wide-range of resource types. Likewise, one could create a macro for titles like so:

```
  <macro name="title">
    <choose>
      <if type="book">
        <text variable="title" text-case="sentence" font-
style="italic"/>
      </if>
      <else>
        <text variable="title" text-case="sentence"/>
      </else>
    </choose>
  </macro>
```

Because of the **value** of macros and the potential to reuse them in different styles and automated software tools, it is recommended that you try to adapt common macro names, such as:

- title
- author
- author-short
- editor-translator
- publisher
- access (for URLs and archival locations)
- event (for conference, hearings, etc.)
- issued
- issued-year
- pages
- citation-locator (for cited pages and such)
- locators (volume and issue, for example)
- container-prefix (for the "In" and such that often preceded container info)
- edition (for edition or version info)

Locale

locales-xx-XX.xml

To support style localization, CSL offers preset translations of terms, localized date layouts (RZ:not yet implemented) and localized punctuation (RZ:idem). A list of available locale files (e.g. locales-en-US.xml), can be found in the [[http://xbiblio.svn.sourceforge.net/viewvc/xbiblio/csl/locales/|xbiblio code repository]].

Example of the (reduced) contents of a locale filee:

```
<?xml version="1.0" encoding="UTF-8"?>
<locale xml:lang="en" xmlns="http://purl.org/net/xbiblio/csl">
  <terms>
    <term name="no date">n.d.</term>
    <term name="et-al">et al.</term>
    <term name="page">
      <single>page</single>
      <multiple>pages</multiple>
    </term>
    <term name="page" form="short">
      <single>p</single>
      <multiple>pp</multiple>
    </term>
  </terms>
</locale>
```

Overriding locales-xx-XX.xml

Localization as provided by the locales-xx-XX.xml files can be overridden using the `cs:locale` element, using the same structure used in the locale files:

```
<locale xml:lang="en">
  <terms>
    <term name="editortranslator" form="short">
      <single>ed. &amp; trans.</single>
      <multiple>eds. &amp; trans.</multiple>
    </term>
  </terms>
</locale>
```

Rendering Elements

Layout

All desired rendering elements (with the exception of `cs:layout` itself) in `cs:citation` and `cs:bibliography` should be nested inside the `cs:layout` element. This element offers similar functionally to the `cs:group` element in specifying a delimiter (RZ: the delimiter doesn't do anything when used in cs:bibliography, right?), affixes and other types of formatting for all enclosed contents. When used in the `cs:citation` element, the delimiter will be used to delimite multiple items in a single citation, e.g.:

```
<layout prefix="(" suffix=")" delimiter=", ">
  <text variable="citation-number"/>
</layout>
```

... would result in citations formatted as "(1, 2)".

Text

The `cs:text` element can be used to output **text** from a number of sources:

- variable - the contents of one or more variable. The `form` attribute can be set for variables that have both "long" (default) and "short" forms. Multiple variables can be separated in the output with a delimiter.
- macro - the results of evaluating a macro
- term - a specific term which is subject to localisation. The `include-period` attribute can be set to "true" (default is "false) to append a period to short-form ("short" or "short-verb") terms. The `plural` attribute can be used to set pluralization behavior (RZ: why is this, really? Is this just the only way to choose between the "single" and "multiple" forms of a term? Context dependency doesn't seem to make sense here), with the possible **values**:
  - "always" - always use the plural form, e.g. "page 1, pages 1-3"
  - "never" - always use the singular form, e.g. "page 1, page 1-3"
  - "contextual" (default) - based on (RZ:???) In addition the `form` attribute can be used to specify the term-form, with **values**:
  - "long" - e.g. "editor"/"editors" for the term editor
  - "verb" - e.g. "edited by" for the term editor
  - "short" - e.g. "ed"/"eds" for the term editor
  - "verb-short" - e.g. "ed" for the term editor
  - "symbol" - e.g. "�" for the term section
- **value** - use for verbatim **text**
- point-locator - a descriptor for locating sub-item content within a cited resource (e.g. used in some styles to indicate specific page numbers for excerpted content)

In addition, formatting attributes can be used (optionally).
For example:

```
<text variable="title" prefix=" Title: " form="short" font-
style="italic"/>
```

Date

The `cs:date` element can be used to output one or more of the following date variables (assuming they hold **values** that can be parsed as dates, which can be tested with the `is-date` conditional): (RZ: is the ability to display multiple variables really useful (e.g.)? I've never come across styles that use this)

- issued
- event
- accessed
- container
- original-date

To allow dates to be displayed in any format, the `cs:date` itself only acts as a wrapper for one or more `cs:date-part` elements. However, any of the formatting attributes can be set for the `cs:date` element. For `cs:date` elements specifying multiple date variables, a delimiter can be set to separate the variables.

Date-part

The `cs:date-part` elements specify the different date parts of the date(s) specified in `cs:date`. The following `cs:date-part` names are available:

- month - the `form` attribute can be set to:
  - "long" (default) - e.g. "January"
  - "short" - e.g. "Jan"

    - "numeric" - e.g. "1"
    - "numeric-leading-zeros" - e.g. "01" The `include-period` attribute can be set to "true" (default is "false) to append a period to the "short" (abbreviated) month date-part.
- day - the `form` attribute can be set to:
    - numeric (default) - e.g. "1"
    - numeric-leading-zeros - e.g. "01"
    - ordinal - e.g. "1st"
- year - the `form` attribute can be set to:
    - long (default) - e.g. "2005"
    - short - e.g. "05"
- other - Other represents any non-month/day/year date part, also in short/long form. (RZ: if you don't know the format of 'other', how can you ever parse it into either short or long form?)

In addition, formatting attributes can be used (optionally).
For example:

```
<date variable="issued" prefix="(" suffix=")">
   <date-part name="year" suffix=" "/>
   <date-part name="month" form="short" suffix=" "/>
   <date-part name="day"/>
</date>
```

Number

The `cs:number` element can be used to output any of the following variables (assuming they hold **values** that can be parsed as numbers, which can be tested with the `is-numeric` conditional): (RZ: should is-numeric fail or pass on "12th edition", schema isn't very clear on that)

- edition
- volume
- issue
- number
- number-of-volumes

The `cs:number` element tries to extract the first number found in the variable field. If no number is detected, no output is generated.
The `form` attribute of 'cs:number' can be set to: * numeric (default) - e.g. "1", "2", "3" * ordinal - e.g. "1st", "2nd", "3rd" * long-ordinal - e.g. "first", "second", "third" * roman - "i", "ii", "iii"
In addition, formatting attributes can be used (optionally). E.g. the **text**-case can be applied to capitalize the roman numbers.
When used in a conditional, number tests if there is a number present, allowing conditional formatting. (RZ: This just refers to is-numeric, right?)

Names

Name

These are the types of contributors that can be used in the layout. They can be displayed with the `cs:names` element. They map to various things in the zotero entries. Some of them are available in both short and long form.

- author
- editor
- translator
- publisher
- original-author
- original-publisher
- recipient
- interviewer
- series-editor
- composer

Contributor markup is done using the `cs:names` and `cs:name` elements. The names wraps the whole cotributor list, and the name how to format an individual. The names also allows a `cs:substitute` block to fill in with other syntax. For the `cs:name` block, there are a number of options that can be specified, besides the generic formatting:

- form - long or short.
- and - set to either //symbol// to use & or //**text**// to use the word "and" to combine authors.
- delimiter - set to something like "," to separate names.
- delimiter-precedes-last - //always// uses the delimiter even for the last author, //never// doesn't.

- name-as-sort-order - //first// sorts by the first author, //all// doesn't.
- sort-separator - some **text** to separate the first and last names.
- initialize-with - the **text** to follow each initial and a directive to use initials.

e.g.,
```
<names variable="author">
 <name form="short" and="symbol" delimiter=", " initialize-with=". "/>
</names>
```
The `cs:substitute` element comes into play if the named author variable is missing. It allows other things to be substituted. For instance
```
<names variable="author">
 <name name-as-sort-order="all" and="symbol" sort-separator=", "
initialize-with=". "
    delimiter=", " delimiter-precedes-last="always"/>
 <label form="short" prefix=" (" suffix=".)" text-
transform="capitalize"/>
 <substitute>
   <names variable="editor"/>
   <names variable="translator"/>
   <text macro="title"/>
 </substitute>
</names>
```
would fill in with the editor, translator or the title in that order.

Et-al

Although the `cs:et-al` element does not have to be declared to enable et-al substitution (setting the et-al options is sufficient), `cs:et-al` can be declared explicitly within `cs:names` to allow formatting attributes to be attached to the output of `cs:et-al`. In addition, the substitution-term may be set to either "et-al" (the default) or "and others", allowing for different et-al substitution strings between in-**text** citations and the bibliography.
```
<names variable="author">
 <name/>
 <et-al term="and others" font-style="italic"/>
</names>
```

Label

The `cs:label` element is used to print **text** terms that depend on document content for pluralization. An example is the label for pages, which can be either singular or plural (p. or pp.).
```
<group prefix=" (" suffix=")">
  <label variable="page" form="short" suffix=". "/>
  <text variable="page"/>
</group>
```
When specified within a layout or group element, the variables for which `cs:label` can be used are `page` and `locator`. Alternatively, `cs:label` can be called in `cs:names`, in which case the variable specified in `cs:names` is passed to `cs:label`. In both cases any of the formatting attributes can be used. Other attributes that can be set are: * `include-period` - set to "true" (default is "false) to append a period to short-form ("short" or "short-verb") terms. * `plural` - sets pluralization behavior, with the possible **values**: - "always" - always use the plural form, e.g. "page 1, pages 1-3" - "never" - always use the singular form, e.g. "page 1, page 1-3" - "contextual" (default) - based on (RZ:???) * `form` - the form of the label-term, with **values** (RZ: except for short/long these only make sense for labels specified in cs:names) - "long" - e.g. "editor"/"editors" for the term editor - "verb" - e.g. "edited by" for the term editor - "short" - e.g. "ed"/"eds" for the term editor - "verb-short" - e.g. "ed" for the term editor - "symbol" - e.g. "�" for the term section

Group

The group element is used to set a delimiter and common formatting attributes to collections of rendering elements. It also acts as an conditional: if none of the enclosed child variables and macro call results produce, 'decorating' output such as is ignored.
```
<group delimiter=": ">
 <text variable="publisher-place"/>
 <text variable="publisher"/>
</group>
```
A group can (optionally) represent semantic document components, as in:
```
<group class="container" prefix=". ">
```

Style Behavior

Most of following syntax applies to the `cs:macro`, `cs:citation` and `cs:bibliography` sections.

Options

Styles are partially configured by setting a number of options. Some of these options are available in both the `cs:citation` and `cs:bibliography` sections, while others are specific to one of the two sections. Below a description and example is given for each option.

Common options

Common options can be set (separately) in both `cs:citation` and `cs:bibliography`.

- et-al-min - the minimum number of contributors (e.g. authors, editors, etc.) for et-al abbreviation to kick in.

```
<option name="et-al-min" value="6"/>
```

- et-al-use-first - the number of contributor names to display when et-al abbreviation is used.

```
<option name="et-al-use-first" value="6"/>
```

Citation only options

- et-al-subsequent-min - as et-al-min, but for subsequent references.

```
<option name="et-al-subsequent-min" value="6"/>
```

- et-al-subsequent-use-first - as et-al-use-first, but for subsequent references.

```
<option name="et-al-subsequent-use-first" value="1"/>
```

- disambiguate-add-year-suffix - disambiguate in-**text** references that are otherwise the same, by adding year-suffixes, e.g. "Doe 2007a, 2007b".

```
<option name="disambiguate-add-year-suffix" value="true"/>
```

- disambiguate-add-names - disambiguate in-**text** references that are otherwise the same by adding additional contributor names, disregarding the "et-al" setting.

```
<option name="disambiguate-add-names" value="true"/>
```

- disambiguate-add-givenname - disambiguate in-**text** references that are otherwise the same by adding given names, e.g. "John Doe, 2005; Mary Doe, 2005" instead of "Doe, 2005; Doe 2005".

```
<option name="disambiguate-add-givenname" value="true"/>
```

- collapse - can be set the the **values**:
  - citation-number - collapses numeric citations from [1, 2, 3] to [1-3]. For correct results citations should also be sorted by citation-number.
  - year - collapses subsequent citations with the same author, e.g. "(Doe 2000, 2001)" instead of "(Doe 2000, Doe 2001)".
  - year-suffix - collapses as for the year **value**, but also collapses identical years, e.g. "(Doe 2000a, b)" instead of "(Doe 2000a, Doe 2000b)". This setting is ignored if disambiguate-add-year-suffix is not set to true.

Bibliography only options

- hanging-indent - formats each bibliography entry with a hanging indent.

```
<option name="hanging-indent" value="true"/>
```

- second-field-align - if set to true, subsequent lines of each bibliography entry are aligned with the beginning of the second field. For example, if the first field is **<text** variable="citation-number" suffix=". "/>:
  1. Adams, D. (2002). The Ultimate Hitchhiker's Guide to the
     Galaxy (1st ed.).

If set to margin, the first field is put in the margin and all subsequent lines are aligned with the margin (as is the case for the IEEE style).

```
<option name="second-field-align" value="margin"/>
```

- subsequent-author-substitute - substitutes subsequent recurrences of an author for a given string, e.g.:
  Asimov. Foundation, 1951.
  ---. Foundation and Empire, 1952.
  ---. Second Foundation, 1953.

```
<option subsequent-author-substitute="---"/>
```

- line-spacing - defines spacing between lines in units of lines (default **value** is 1)

```
<option name="line-spacing" value="2"/>
```

- entry-spacing - defines spacing between entries in units of line-spacing (default **value** is 1)

```
<option name="entry-spacing" value="2"/>
```

Sorting

The sorting order for in-**text** citation clusters [e.g. (Doe 2001; Johnson 2003)], and for the bibliography can be set with the `cs:sort` element, in which one or more sort keys can be specified. Sort keys can specify either variables or macros. Sort order ("ascending" (default) or "descending") can be specified with the `sort` attribute. An example:

```
<citation>
  <sort>
    <key macro="author"/>
    <key variable="issued" sort="descending"/>
  </sort>
</citation>
```

In this example, citations are first sorted by the output of the author macro. Entries that share the same author macro output are further sorted in reverse order by date of issue. Using macros instead of variables as sort keys is especially useful in case of substitutions (e.g. in many styles the editor variable substitutes for an empty author variable), or when the sort key should be the year instead of the complete date of issue.

Conditionals

Conditional statements can be expressed with the `cs:if` element, which can be extended with one or more `cs:else-if` elements and a `cs:else` element to allow for multi-way choices. Conditionals should always be embedded in a `cs:choose` parent element. Conditionals based on the variable type or presence of a variable are common in bibliographies and macros, as in

```
<choose>
  <if type="book">
    ...
  </if>
  <else-if type="chapter">
    ...
  </else-if>
  <else>
    ...
  </else>
</choose>
```

The following tests are available in `cs:if` and `cs:else-if` elements: * `type` - tests the item type * `variable` - tests whether a variable has a **value** * `is-numeric` - tests whether a variable has a numeric **value** * `is-date` - tests whether a variable has a date **value** * `position` - tests the position of the item citation in the **text**. Possible positions are `first`, `subsequent`, `ibid`, `ibid-with-locator`. The first time an item is cited, the position of the citation will be `first`. If the next citation again references that item, the position becomes `ibid`, or, if a locator is added to the second cite, `ibid-with-locator`. Finally, if the same item is again referenced after another item has been cited, the position becomes `subsequent`. Whenever position="ibid-with-locator" is true, position="ibid" is also true, and whenever position="ibid" is true, position="subsequent" is also true * `disambiguate` which can be tested against true/false. If disambiguate is is tested against "true", the **text** inside the conditional will be used if it will differentiate two otherwise identical citations. If the citations remain identical after its addition, it will not be added. (RZ: don't understand how this works. Is this test available in citation clusters as well as in bibliographies? Does the "two" make sense, or are all citations compared?) * `locator` - tests the locator type (`page`, `chapter`, `verse`, etc.) * `match` - an extension of the conditional to include AND/OR/NOT like testing behaviour by setting against `all`/`any`/`none`, respectively. E.g. a test which is true if the item type is either "chapter" or "book":

```
<if type="chapter book" match="any">
```

Formatting Attributes

The following formatting parameters, all of which are optional, can be used for any of the rendering elements (`names`, `name`, `et-al`, `date`, `date-part`, **`text`**, `number`, `label`, `group` and `layout`). With the exception of the `layout` element, formatting parameters do not affect the affixes specified in the rendering element. If affixes should receive formatting, affixes can be transferred to standalone '**text**' elements, and a 'group' element can be used to specify formatting of all enclosed rendering elements.

- **`prefix`**: **text** to insert before main output
- **`suffix`**: **text** to insert after main output
- **`font-style`**:
  - "normal" (default)
  - "italic"
  - "oblique" (slanted)

- **`font-variant`**:
    - "normal" (default)
    - "small-caps"
- **`font-weight`**:
    - "normal" (default)
    - "bold"
    - "light"
- **`text-decoration`**:
    - "none" (default)
    - "underline"
- **`text-case`**: changes **text** case
    - "lowercase" - display as lowercase
    - "uppercase" - display as uppercase
    - "capitalize-first" - capitalize first character; other characters displayed as is
    - "capitalize-all" - capitalize first character of every word; other characters displayed lowercase
    - "title" - display as title case (the Chicago Manual of Style calls this "headline style")
    - "sentence" - display as sentence case/sentence style
- **`vertical-align`**:
    - "baseline" (default)
    - "sup" - superscript
    - "sub" - subscript
- **`display`**:
    - "block" - outputs the **text** in a block
    - "inline-block" - (RZ: waiting for a sensible description)
- **`quotes`**: wrappes main output in quotes if set to true (default **value** is false)

Delimiter

In addition to the formatting attributes listed above, a `delimiter` may be specified for the `names`, `name`, `date`, **`text`**, `group` and `layout` rendering elements.
An example:

```
<group delimiter=", ">
  <text variable="title" font-style="italic"/>
  <text variable="publisher" prefix="(" suffix=")"/>
</group>
```

Appendices

Appendix I - Variables

Source Variables

Source variables contain the properties of the cited items, and have no relation to the **text** in which the items are cited.

- title
- container-title
- collection-title
- original-title
- publisher
- publisher-place
- event
- event-place
- page
- locator
- version
- volume
- number-of-volumes
- issue
- medium
- status
- edition
- genre
- note

- annote
- abstract
- keyword
- number
- archive
- archive_location
- archive-place
- URL
- DOI
- ISBN

Date variables

- issued
- event
- accessed
- container
- original-date

Name variables

- author
- editor
- translator
- recipient
- interviewer
- publisher
- composer
- original-publisher
- original-author
- container-author (to be used when citing a section of a book, for example, to distinguish the author proper from the author of the containing work)
- collection-editor (use for series editor)

Citation Variables

Citation variables are assigned by the CSL processor. Their **value** can be dependent on the position of the cited items (for citation-number), or on the disambiguation logic and formatting of the selected CSL style (for citation-label) (RZ: not sure this is true).

- citation-number (RZ: this is the citation-number without its affixes, right?)
- citation-label (RZ: what does the citation-label represent? Is it the entire citation output, e.g. "(Doe 1999)"? If so, what happens with citation-label in (collapsed) citation clusters (e.g. (Doe 1999, 2000;Johson 1929)))?

Appendix II - Types

These are the different item types available within CSL:

- article
- article-magazine
- article-newspaper
- article-journal
- bill
- book
- broadcast
- chapter
- entry
- entry-dictionary
- entry-encyclopedia
- figure
- graphic
- interview
- legislation
- legal_case
- manuscript
- map
- motion_picture
- musical_score

- pamphlet
- paper-conference
- patent
- post
- post-weblog
- personal_communication
- report
- review
- review-book
- song
- speech
- thesis
- treaty
- webpage

Appendix III - Terms

These are the different terms available within CSL:

- Miscellaneous Terms
  - accessed
  - anonymous
  - and
  - and others
  - at
  - et-al
  - forthcoming
  - from
  - in press
  - ibid
  - in
  - no date
  - references
  - retrieved
  - letter
  - interview
  - online
  - cited
  - edition
  - internet
  - presented at
- Roles
  - editor
  - translator
  - interviewer
  - recipient
- Months
  - month-01
  - month-02
  - month-03
  - month-04
  - month-05
  - month-06
  - month-07
  - month-08
  - month-09
  - month-10
  - month-11
  - month-12
- Other
  - cs-terms.locator (locators)
  - book

- chapter
- column
- figure
- folio
- issue
- line
- note
- opus
- page
- page-range (a synonym for "page", to be deprecated)
- page-first
- paragraph
- part
- section
- sub verbo
- volume
- verse
- cs-terms.extension
- info-fields