Hereinafter: A Legal Citation Program

Charles Duan[*]

February 28, 2026
Version 1.0.1

Contents

Part I
Introduction

You've done the research, your stack of cases and articles is overflowing, the ideas are swimming in your head, and are ready to start writing. But as you open your document, a troubling question creeps in.

Will you format citations as you write? That will interrupt your train of thought every few minutes as you pull out the style guide. And as you edit and rearrange, your citations will need to be redone many times over. All those *supra* and *id.* citations will quickly become out of place and perhaps unintelligible as you polish the text.

Or will you put off citation formatting for later? In that case your coauthors and draft reviewers will see a mess of an unfinished paper. And you'll have to reserve hours or days before submitting for publication, to slog through an unrewarding round of citation formatting.

Hereinafter offers a better way. With it, you can type citations in your writing concisely and efficiently, but with all the features of a rich legal citation system. You can edit and reorganize with almost no need to manually update existing citations in your document. And you can produce correctly formatted citations at any stage of the writing process, so that when you and others are reviewing drafts, you aren't distracted by incomplete formatting.

I have been developing *Hereinafter* since I started law school in 2004. In the years since, I have used it to format tens of thousands of citations in hundreds of legal briefs, administrative comments, and journal articles. It produces tables of authorities without hesitation and handles lengthy law review pieces without difficulty. I can confidently polish my pieces right up to the deadline, knowing that my *supra* and *id.* citations will still be perfect after a recompile.

Most importantly, when the computer handles the appearance of my writing, I get to focus on the substance. This has made the writing process far more rewarding and enjoyable. Have you ever read through one of your own drafts and thought, "Part II.A really belongs after II.B, but it'll be too much trouble to fix the footnotes"? I don't. *Hereinafter* lets me strive toward my best work, rather than settling for the easy but imperfect.

To me, *Hereinafter* is an indispensable tool for legal writing. I hope it will also become your choice hereinafter.

1 Quick Start Guide

This introductory guide will show the basic concepts you'll need to use *Hereinafter* to add citations to your document. As with other automated citation systems, you will need to do two things:

1. Define your references (cases, books, articles, and so on) by entering bibliographic information for them.

2. Cite references by entering citation commands into your document.

Importantly, these two steps do not need to be done in order. In my personal workflow, I will often first write, entering citation commands for references that I've read but have not defined. I find that this order preserves my train of thought best. *Hereinafter* displays a list of undefined references and other missing items at the end of document compilation, conveniently letting you know what needs to be fixed.

1.1 Prerequisites

At this stage, I assume that you have general knowledge both of legal citation and of the LaTeX typesetting system. *Hereinafter* operates purely within LaTeX, and is controlled by commands described below in this manual.

To begin, you should include the package in the preamble of your document:

```
\usepackage[⟨options⟩]{hicite}
```

The ⟨*options*⟩ are described in section 5. The most important of these are the options review and memo, which determine whether the package will follow the rules for law review articles or for legal memoranda, respectively.

1.2 Defining References

A *reference* is a single work of authority that you want to cite. In order to cite a reference in *Hereinafter*, you must "define" it by entering key information about the reference.

To define a reference, you will need to (1) choose a nickname for the reference, (2) determine the type of reference, and (3) identify the relevant bibliographic information for the type. Part VI lists the available reference types; common ones are book, case, website, and jrnart (journal articles). The bibliographic information is entered as parameter-value pairs. For example:

```
\defcase{wheaton}{
    p=Wheaton,
    d=Peters,
    vol=33,
    rep=U.S.,
    page=591,
    year=1834,
}
```

defines a reference named `wheaton` for the famous lawsuit between two Supreme Court reporters. For a more detailed overview of how to define references, see section 6.2 and the sections that follow.

Hereinafter permits for various shortcuts to simplify the definition of references. The following are equivalent to the above definition:

```
\defcase{wheaton}{
    parties=Wheaton v. Peters,
    cite=33 U.S. 591,
    year=1834,
}
\defcase{wheaton}{
    Wheaton v. Peters, 33 U.S. 591 (1834)
}
```

Shortcuts for reference definition are described in section 7.

A wide range of types of references can be defined in *Hereinafter*. The full list, with many example definitions, is given in Part VI. For example, here is how one would define a reference for this manual:

```
\defbook{hereinafter}{
    author=Charles Duan,
    title=Hereinafter: A Legal Citation Program,
    year=2026,
}
```

A key difficulty with legal citation is the wide range of possible parameters to a reference definition—different types of authors, date formats, optional parentheticals, and more. *Hereinafter* provides structure to that range of possibilities, thereby offering as much flexibility as possible while retaining consistency. Consider, for example, the problem of a work written by a person employed by an institution. You may consider both the person and the institution to be the "author." *Hereinafter* allows for this:

```
\defbook{ftc-pharma}{
    author=Markus H. Meier et al.,
    instauth=Federal Trade Commission,
    title=Overview of FTC Actions [...],
    date=june 2019,
}
```

All parameters are described in Part VII. Authors and dates are the most complex parameters; they are described in section 8 and section 9, respectively.

1.3 Entering Citations

To cite references in a document, the key commands are `\sentence` and `\clause`, which produce sentence-style and clause-style citations. For law review articles, these commands should be used in footnotes, and the `\note` command creates a footnote with a sentence-style citation.

All of these commands receive a single argument of a citation string following a syntax described in section 15. By design, the syntax looks like a simplified legal citation. The following would generate a two-element string-cite to *Wheaton v. Peters*, describing a (hypothetical) tension between two parts of the decision:

```
\sentence {see, e.g., wheaton at 598 note 3 (Story, J.)
(describing holding); but see wheaton at 600-602 (noting
alternative views)}.
```

Produces:

See, e.g., *Wheaton v. Peters*, 33 U.S. 591, 598 n.3 (1834) (Story, J.) (describing holding); *but see id.* at 600–02 (noting alternative views).

This example shows a number of notable features of *Hereinafter*. First, references are always cited by identifier; any short or *id.* form is selected automatically without user intervention. This means that citations are portable: They largely can be moved around the document unchanged, simplifying reorganizations of one's writing.

Second, there is rich support for signals, pin cites, and parentheticals in citations. In particular, pin cites can involve page numbers, section numbers, footnotes, and more, as described in section 17. They are abbreviated automatically, as shown above. Parentheticals and signals are also ordered, arranged, and formatted appropriately.

Third and perhaps most importantly, nearly all the formatting work is hidden from the writer. Abbreviations, fonts, and keywords are automatically chosen to conform to a given citation system. Importantly, this makes it easy to conform an already-written document to another citation system. If one journal prefers small caps but another does not, it is simple to produce different documents for each.

1.3(a) Inline Citations In addition to sentence and clause citations, *Hereinafter* supports citing references as parts of the text: naming a case as the subject of a sentence, for example. The relevant commands are `\inline`, `\Inline`, `\adjective`, and `\Adjective` depending on whether the reference is being cited at the beginning of a sentence and the part of speech the reference is taking on:

```
In \inline {wheaton}, the Court....  \sentence {wheaton at
603}.
```

Produces:

In *Wheaton v. Peters*, the Court.... *See* 33 U.S. 591, 603 (1834).

Note how the `\sentence` citation automatically omits the case name because it was displayed in with the previous `\inline` command.

For more on the parts and syntax of sentence, clause, and inline citations, see Part III.

1.4 Cross-References

Hereinafter also provides extensive support for cross-references. This requires not only commands for inserting cross-reference citations, but also commands for marking portions of the document to be referenced.

The `\hilabel` command marks a part of a document for cross-referencing. Optionally, a range of a document can be marked using `\hiendlabel` as follows:

```
\hilabel{keypart}
This is the key part!
\hiendlabel{keypart}
```

The command automatically chooses the best way to refer to the marked section depending on where it is used:

- When called in a footnote, the cross-reference is to the footnote number.
- When used in a figure, the cross-reference is to the figure number.
- When used in main body text, *Hereinafter* chooses a cross-reference form depending on the document type: page numbers for a memo-style document, and nearest footnotes for a review-style document ("text accompanying notes x–y").

Additional commands give further options for cross-references. The `\hisectlabel` command can label sections by number, and `\hitanlabel` (terminated with `\hiendtanlabel`) produces a cross-reference citation to both text and footnotes (written as "notes x–y and accompanying text").

To insert a cross-reference, simply use a citation to the reference this with the cross-reference label as the pincite:

```
\sentence{see this at keypart}.
```

Again, the value of this cross-referencing system is that the cited numbers automatically update as the document is edited, minimizing the amount of manual updating.

The full cross-referencing system is described in section 10.

1.5 Additional Features

In addition to the above, *Hereinafter* provides additional features of interest to many authors of legal documents:

- It can produce tables of authorities, as described in section 20.
- It can switch footnotes into endnotes, as described in section 21.
- It can manage abbreviations in text, as described in for reference type abbrev, properly inserting the full version and an explanatory parenthetical the first time the term is used.
- It properly line-breaks URLs, as described in section 11, avoiding unsightly gaps; it also provides several alternate means of presenting URLs in documments.

These, of course, are in addition to the many features of *Hereinafter* that you *don't have to think about anymore*: abbreviations, short citations, font selection, *supra* note numbers, and more.

2 Structure of This Manual

Besides simply describing how *Hereinafter* works, this manual is intended to explain my conceptual framework for legal citation that is embedded in the system. With that framework in mind, this manual on a more practical level then teaches writers how to write citation input commands that will produce formatted citations meeting their needs, and explains how those inputs are formatted. As a result, it has three main objectives:

- Describing the data model behind *Hereinafter*'s legal citations
- Specifying the input formats that writers must use
- To a lesser extent, explaining the algorithms for producing formatted citations

The manual is organized topically for different elements of legal citations, with each section covering these three objectives of the data model, input formats, and formatting algorithms (to the extent that each objective is relevant). There are five parts:

1. How references are defined, including common parameters of references such as authors and dates.
2. How citations are entered into documents.
3. General formatting algorithms used, such as keeping track of previous citations for short forms.
4. All reference definition parameters.
5. All reference types.

As currently written, this manual is primarily for writers of legal documents who already have a baseline knowledge of legal citation, and secondarily for software developers who wish to learn how the system works so that they can modify it to produce different citation formats. It is not as useful for learning how to format legal citations, as most citation manuals are. First, it explains the formatting of citations in a manner that is useful to programmers but probably confusing to human cite-checkers.[1] Second, this manual does not discuss the meanings of citation elements: It does not explain what citation signals mean, when parentheticals are appropriate, or which edition of a book to cite, for example.

It would not be difficult to revise this manual to address these deficiencies and turn it into a guide on manual citation formatting, by augmenting the data model sections to describe semantics and by rewriting the formatting algorithm sections. Were that work to be done, the hope is that presenting students with a data model and principles in tandem with formats would offer them a clearer, enriched view of the seemingly arcane world of legal citation.

[1] In particular, a computer reads and formats citations sequentially, starting from the beginning and never looking backwards, so at any given point it must keep track of a great deal of information in memory about prior citations. Computers do this easily, but a person would have to write copious and error-prone notes to check the citations purely in forward order. A more reasonable approach for a person would be to read the citations backwards to find necessary information about previous citations.

2(a) Grammatical structures In many places, this manual describes expected inputs in terms of grammatical structures. These are shown in blue boxes and use a version of Backus-Naur form.[2] For example:

> ⟨*address*⟩ := ⟨*number*⟩ ⟨*street*⟩ [⟨*apartment*⟩] ‖ `PO Box` ⟨*number*⟩

For those unfamiliar with this notation:

- The items between angle brackets represent text values, akin to entry spaces in a form.
- := means that the item on the left is defined to consist of the specified items on the right.
- ‖ means to choose either of the items to the left or right.
- Square brackets [ ] mean that the material inside is optional.
- Underlined text needs to be included literally.
- (⟨*item*⟩)* means that whatever is inside the parentheses can be repeated (or used once, or omitted entirely).

So to explain the above statement, an ⟨*address*⟩ can take on two forms. It can be a ⟨*number*⟩ followed by a ⟨*street*⟩ and optionally an ⟨*apartment*⟩ number, such as "123 Elm Lane" or "15 Broadway #5. Or it can be a post office box like "PO Box 123". In general, grammatical definitions will be followed by examples to help demonstrate what input is expected.

In other cases, the manual will show expected inputs using a LaTeX syntax:

> `\command[`⟨*option*⟩`]{`⟨*required*⟩`}`

This means that the macro `\command` accepts an optional argument ⟨*option*⟩, surrounded by square brackets if present, and needs a mandatory argument ⟨*required*⟩, surrounded by curly braces.

2(b) Documentation files This manual is one of two parts of documentation for *Hereinafter*. The package source code is contained in multiple module files compiled into a unified main package, and most chapters of this manual correspond to module files. The module files are written as literate programming LaTeX `.dtx` files, and contain further documentation of the internal macros and implementation. For each chapter having a corresponding module file, the filename is listed in the margin so that the expanded documentation can be identified.

The manual alone should be sufficient for writers of legal documents to understand how to use *Hereinafter*. The module file documentation is intended to help package authors who wish to correct, modify, or extend the citation program itself, perhaps to adapt it to other citation formatting regimes.

[2] Spaces between items are not specified rigorously in these grammars.

3 Installation and Usage

Hereinafter comprises a suite of files:

- The package file itself, `hicite.sty`.
- Several auxiliary package files, described in Part VIII.
- Tables of abbreviations and other information.

All of these files need to be installed in a TeX-accessible location.

To use the package, simply include it in a document:

```
\usepackage[⟨options⟩]{hicite}
```

Then compile it using LaTeX or an equivalent. No auxiliary programs are needed, even for table indexing. However, multiple compilation runs may be necessary in order to get cross-references and short citations to be correct.

If you have an existing Biblatex file, see section 49 for possible compatibility options.

4 Intellectual Property Concerns

Over the years, there have been multiple disputes over intellectual property in legal citation systems and manuals. It is both necessary to address both what rights I claim as the author of *Hereinafter*, and what rights of others it implicates.

The source code of *Hereinafter* is released under the GNU General Public License (GPL) version 3.0. That license applies to any executable code. The text of this manual itself and all other documentation is released under a Creative Commons license as described on the first page of the manual. For purposes of clarity, the Creative Commons license applies to any text found in this user manual and any text found in comments in the package source code; the GPL license applies otherwise.

With respect to other legal citation manuals and systems, this software and manual are not endorsed by, affiliated with, or authorized by any other legal citation system, such as the Bluebook, the University of Chicago Manual of Legal Citation (the "Maroonbook"), the ALWD Guide to Legal Citation, or the Indigo Book. It does not claim to be compatible with or identical to any of those systems; indeed a design principle of *Hereinafter* is to be more flexible than any of these existing systems. To the extent that material from existing citation systems is used (primarily for tables of abbreviations), it has been taken from the Indigo Book, which has been placed in the public domain.

5 Package Options

The following options are accepted by this package.

review Law review formatting: selects law review fonts, expects footnotes, formats *supra* cites with note numbers, and so on.

journal Alias for review.

journalfonts Selects law review fonts. Same as `\UseFontPack{review}`. Other formatting aspects are not changed.

memofonts Selects legal memorandum fonts. Same as `\UseFontPack{memo}`. Other formatting aspects are not changed.

memo Legal memorandum formatting: selects memorandum fonts, expects citations in text, and so on.

brief Alias for memo.

endnote Uses endnotes rather than footnotes for formatting.

endnotes Same as endnote.

fullurl URLs are displayed with their full text.

toaurl URLs are displayed with their full text in the table of authorities and a shortcut otherwise, as described in section 11.2.

linkurl URLs are displayed as a linked word, as described in section 11.2.

italcase Case names are italicized regardless of font packs previously chosen.

toastar Stars are added to common references in the table of authorities, as described in section 20.5(a).

somenotename In general, if a case is named inline in text and a citation to the case is made immediately thereafter, then the case name is suppressed in the citation so as to avoid redundancy. This option makes an exception where the citation to the case is the first citation in a footnote. In such cases, the inline use was in text potentially distant from the footnote, so naming the case again in the footnote is less redundant and potentially helpful to readers.

nonotename Suppresses case names even in citations at the start of footnotes. This is typical of most legal journals.

useopturl Redefines the opturl parameter to be the same as url, as described in the documentation for the former parameter.

noterefs Cross-references to text will refer to the closest footnote numbers. This is typical of law reviews and is the standard when the review option is selected.

pagerefs Cross-references to text will refer to the page numbers on which the relevant text is found. This is typical of legal memoranda and is the standard when the memo option is selected.

noauthwarn No warning will be issued if three or more authors are given without the noetal parameter.

Part II

References

6 Defining References

A reference represents a single citable work of authority. What constitutes a single work may be uncertain, particularly for statutes and multi-authored works, as described previously. The general test for whether two items are two references or subdivisions of a single reference is whether, if they were cited immediately in sequence, *id.* should be used for the second item. In some cases this is a matter of the writer's taste, so this package generally gives writers the flexibility to define references however they see fit.

If a reference is used in a citation string but not defined, this package will issue an error, or it will keep track of the missing citation. See section 23 for more information on missing references.

6.1 Data Model

Each reference consists of:

- A reference type, such as a book, case, or statute. The complete list of reference types is given in Part VI. The reference type determines how the reference is formatted and what parameters are necessary to define the reference.
- An identifier nickname that will be used to identify this reference in citations (and, on occasion, inside other references). The nickname must be unique and generally should not contain spaces.
- Parameters providing information about the reference. For example, a book uses parameters for the author, title, and publication year. The reference type determines which parameters are required and optional; this is documented in Part VI. There are also parameters of general applicability to all reference types, described in section 40. The complete list of parameters is given in Part VII.
- A Table of Authorities category, described below.

6.2 Input Syntax

The package provides its own syntax for writers to use when defining references. As an alternative, section 49 describes a compatibility layer that accepts Biblatex reference definitions, so long as the parameter names correspond with those of this package.

Generally, references are defined as follows:

```
\def⟨reference type⟩ {⟨name⟩} {
    ⟨param1⟩ = ⟨value1⟩,
    ⟨param2⟩ = ⟨value2⟩,
    ...
}
```

For example, to define the historic case on copyrightability:

```
\defcase{baker}{
    parties=Baker v. Selden,
    cite=101 U.S. 99,
    year=1880,
}
```

The input follows TeX syntax and uses the `keyval` package, so if there are equal signs or commas in the parameter values, then the parameter should be surrounded with braces.

Entering the parameter names for every reference can be tedious, so for common reference types there are shortcuts described in section 7.

6.3 Citation Groups

Often there will be several references that follow a standard conventional form, so it would be convenient to define all of them in one go. The `\defcitegroup` macro enables this to an extent. It takes four arguments:

- (optional) A prefix for the reference nicknames to be defined, default blank
- A reference definer macro (`\def`⟨*type*⟩)
- Initial text for the reference definition body
- A comma-separated list of texts used both as the nickname and as a suffix to the reference definition

For example, say you want to define references for several sections of the Patent Act. That could be achieved with the following three definitions:

```
\defstatcode{101}{35 U.S.C. S 101}
\defstatcode{102}{35 U.S.C. S 102}
\defstatcode{103}{35 U.S.C. S 103}
```

This works (using the reference parser for statutes) but is repetitive. Instead, you can write:

```
\defcitegroup\defstatcode{35 U.S.C. S }{101, 102, 103}
```

Each of the items in the comma-separated list (e.g., `101`) becomes both a reference nickname and the end of the reference definition.

With the optional argument, the reference names can be given a prefix:

```
\defcitegroup[17usc]\defstatcode{17 U.S.C. S }{101, 102,
103}
```

would define `17usc101` as the statute “17 U.S.C. § 101” and so on.

7 Simplified Input

While all references can be defined using the parameter-value input described previously, that input is fairly verbose. To simplify the entry process, common reference types accept an alternate form of input, in which the elements of the reference are given in a citation-like string. The package parses that string to extract parameters, thereby defining the reference.

To use this simplified input, ensure that the reference type is one of the types listed in this section, and then enter the simplified input string in lieu of the parameter-value pairs as follows:

```
\defcase{baker}{
    Baker v. Selden, 101 U.S. 99 (1880)
}
```

Please ensure that the simplified input has no equal signs. If there are additional parameters for the reference definition, they may be entered after a semicolon as follows:

```
\defcase{baker}{
    Baker v. Selden, 101 U.S. 99 (1880);
    docket=No. 95
}
```

7.1 Common Elements

The simplified input formats use several common elements, described below.

7.1(a) Parentheticals Text at the end of a formatted reference input surrounded by parentheses is detected and interpreted as a parenthetical. In the specifications for reference type parsers below, text matching this form will be identified as content inside parentheses.

7.1(b) Volume/Reporter/Page Many of the reference type parsers will detect this pattern commonly found in reported case citations and consecutively-paginated journal citations:

⟨*volume*⟩ ⟨*reporter*⟩ ⟨*page*⟩ (⟨*date*⟩)

Example:

```
123 F.3d 456 (2000)
```

Text matching this format can be detected and separated into parts, according to the following algorithm:

1. If there is a parenthetical, then remove it and use it as the date.
2. The first word of the input is the volume.
3. If the second-to-last word is a division name (see section 17), then the last two words are the page. Otherwise, just the last word is the page.
4. Any words remaining are the reporter/journal name.

Alternatively, the volume, reporter, and page elements can be separated with slashes (i.e., ⟨*volume*⟩/⟨*reporter*⟩/⟨*page*⟩).

In explaining reference inputs below, text matching this form will be identified as ⟨*vol-rep-page*⟩.

7.1(c) Journal-Like Titles Many reference types can accept parseable input of the form:

⟨*journal-text*⟩ := ⟨*authors*⟩ `, ` ⟨*title*⟩ `, ` ⟨*journal-id*⟩
⟨*authors*⟩ := ⟨*author*⟩ [` & ` ⟨*author*⟩]

Example:

```
John Doe & Jane Q. Public, A Theory of Law, ...
```

The author and title parameters are automatically set from this, and ⟨*journal-id*⟩ is passed back to the reference type parser for further analysis. In parsing this text, ⟨*authors*⟩ is taken to be anything before the first comma in the text, and ⟨*journal-id*⟩ is text following the last comma. (Thus, ⟨*journal-text*⟩ must contain at least two commas.)

Text matching this form will be denoted as ⟨*journal-text*⟩ in explaining the parsers below, and those explanations will further describe how ⟨*journal-id*⟩ is used.

7.1(d) URLs Any parseable reference input can have a URL appended to it, set off with a comma. (The URL must start with `http://` or `https://`.) The URL will be extracted from the input text and automatically set as parameter url. Because every parseable reference type will accept this, it is not shown in the explanations below. The URL must be the last item in the parseable reference input.

7.2 Cases

The case reference type can parse the following input syntax:

⟨*case*⟩ := ⟨*case-name*⟩ `, ` ⟨*case-cite*⟩ `(` ⟨*court*⟩ ⟨*date*⟩ `)`
⟨*case-name*⟩ := ⟨*p*⟩ ` v. ` ⟨*d*⟩ ‖ ⟨*name*⟩
⟨*case-cite*⟩ := ⟨*vol-rep-page*⟩ ‖ `No. ` ⟨*docket*⟩

Example:

```
\defcase{wheaton}{Wheaton v. Peters, 33 U.S. 591 (1834)}
```

This sets parameters p and d or name as the case party name(s); vol, rep, and page or docket from the locator information (detected as the last pre-parenthetical text following a comma), and court and date from the parenthetical.

7.3 Codified Statutes

The statcode reference type can parse input conforming to ⟨*vol-rep-page*⟩. For example:

```
\defstatcode{patent-eligibility}{35 U.S.C. S 101}
```

7.4 Journal Articles

The jrnart reference type can parse the following input syntax:

⟨*jrnart*⟩ := ⟨*journal-text*⟩ (⟨*date*⟩)
where ⟨*journal-id*⟩ := ⟨*cite*⟩

Example:

```
\defjrnart{doe}{
    John Doe, Article, 12 Law Review 345 (1950)
}
```

In other words, the start of the parseable input is used for parameters author and name, as explained for ⟨*journal-text*⟩ above. The text after the last comma before the parenthetical is passed as the cite parameter, and the parenthetical is interpreted as year.

7.5 Websites and Magazines

The website reference type can parse the following input syntax:

⟨*website*⟩ := ⟨*journal-text*⟩ (⟨*date*⟩)
where ⟨*journal-id*⟩ := ⟨*rep*⟩

Example:

```
\defmagart{doe}{
    John Doe, Latest Memes, Slate (June 5 2000)
}
```

In other words, a parseable website reference input is just like a journal article except without a volume or page number; anything following the title is passed to the parameter rep to be used as a journal title.

Magazines (magart) are identical except a page number is given after the magazine title:

⟨*journal-id*⟩ := ⟨*rep*⟩ ⟨*page*⟩

Example:

```
\defmagart{doe}{
    John Doe, Politics, Newsweek 16 (June 5 2000)
}
```

As a result, the parseable reference input for magazines does *not* look like conventional magazine citations, which typically set the date with a comma after the magazine title, rather than putting the date in a parenthetical.

7.6 Books

The book reference type can parse the following input syntax:

⟨*book*⟩ := ⟨*authors*⟩ , ⟨*name*⟩ (⟨*date*⟩)

Example:

```
\defbook{blackstone}{
    William Blackstone, Commentaries on the Laws of England
(1765)
}
```

where ⟨*authors*⟩ is as defined above with respect to ⟨*journal-text*⟩ (i.e., one author or two separated with an ampersand).

7.7 Court Documents

The courtdoc reference type can parse the following input syntax:

⟨*courtdoc*⟩ := ⟨*name*⟩ ` (` ⟨*date*⟩ `)`

Example:

```
\defcourtdoc{motion}{Motion for Leave (June 5 2000)}
```

7.8 Abbreviations

The abbrev reference type can parse the following input syntax:

⟨*abbrev*⟩ := ⟨*name*⟩ ` (` ⟨*inline*⟩ `)`

Example:

```
\defabbrev{fda}{
    the Food and Drug Administration (the FDA)
}
```

In other words, the full text is before the parentheses and the abbreviation inside them.

7.9 Aliases

The alias reference type can parse the following input syntax:

⟨*alias*⟩ := ⟨*citation*⟩ [` at ` ⟨*page*⟩]

Example:

```
\defalias{b18}{
    brown at 18
}
```

This will set the citation and page parameters.

7.10 The `cite` Parameter

The parameter cite accepts as input a string conforming to ⟨*vol-rep-page*⟩ described above. It sets the parameters vol, rep, page, and possibly year.

8 Authors and Other Names

Hereinafter offers comprehensive support for lists of names in references, such as authors and editors. Among other things, it provides for both personal and institutional authors, optional list truncation with “et al.,” and formation of last-name lists for short citations.

To enter a list of names for a reference, include the appropriate parameter (e.g., author or instauth) once for each name in the list. The names will automatically be compiled into a formatted list. Every name entered will be listed, although a warning will be issued unless the noetal parameter is included or the noauthwarn option is given for the package. To truncate a list, enter only the initial name with “et al.” attached to the end. For example:

```
\defbook{nimmer}{
    author=Melville B. Nimmer, author=David Nimmer, ...
}
\defjrnart{als}{
   noetal, author=John R. Allison, author=Mark A. Lemley,
   author=David L. Schwartz, ...
}
\defjrnart{als}{author=John R. Allison et al., ...}
```

The remainder of this section provides details on how *Hereinafter* parses and uses name lists, and how to fine-tune those processes.

8.1 Data Model

A name list is an ordered sequence of one or more names, each of which is either a personal name and an institutional name.[3] An institutional name is simple text, although it is abbreviated in citations. A personal name, by contrast, consists of five parts:

- The given name, which includes the middle initials or names. This is the only mandatory part.
- The family name. If only one name is given (e.g., Aristotle), it is treated as a given name, and the short-form name instead uses the given name.
- Suffix to the family name, such as Jr. or III. It should not be included in the last-name-only forms.
- The institution.
- An “et al.” indicator for abbreviating the author list.

Separating these parts is necessary, for example, to enable short citations using just the author’s last name.

Institutional affiliations attached to personal names are distinct from institutional names included in a name list. Consider, for example, a government report written by Al Anderson at the Federal Trade Commission and Bob Baker at the Department of

[3] To avoid questions of what constitutes personhood (e.g., AI-authored works), a “personal name” is one that has a family-name component and would not be abbreviated in citations.

Justice. If the report expresses the views of the agencies, then the agencies could be included as authors:

> Al Anderson & Bob Baker, Fed. Trade Comm'n & Dep't of Justice

By contrast, if the report represents the views of the individuals in their official capacities but not necessarily the views of the agencies, then only the individuals should be authors, with their affiliations attached to their names:

> Al Anderson, Fed. Trade Comm'n & Bob Baker, Dep't of Justice

Providing these options gives writers more flexibility to cite mixed personal and institutional authors to convey accurate and useful information.

8.2 Inputs

For each type of name list, there will be two reference definition parameters, one for personal names and one for institutions:

	Personal	**Institutional**
Authors	author	instauth
Editors	editor	insted
Letter recipients	to	instto

Institutional authors are given simply as an unabbreviated name; the package handles abbreviation. If an "et al." is desired after an institutional author, that may be appended to the name. It is treated as if it were part of the institution's name and not processed further.

Because personal authors have five component parts that need to be analyzed individually, they must be given in a specific format. An institutional affiliation is always any part of a name following a comma. To separate the given name, family name, and suffix, the most explicit way is to surround the family name and name suffix with curly braces:

```
⟨name⟩ := ⟨given-name⟩
    [ {  ⟨family-name⟩ [ { ⟨suffix⟩ } ] } ]
    [ ,  ⟨institution⟩ ]
    [ et al. ]
```

Examples:

```
John {Marshall}
Fiona {Scott Morton}
Nicholson {Price {II}}
Oliver Wendell {Holmes {Jr.}}, Massachusetts Supreme
Judicial Court
```

Regarding the last two items, some style guides prefer a comma before the name suffix. *Hereinafter* does not explicitly support this, but with a preceding comma the name suffix would be treated as an institutional affiliation, which produces desirable results.

Because including the braces around family names is cumbersome, this package can infer name parts with a few heuristics, such that no braces are required.

8.2(a) Family names. When no braces delimit the family name, the package first looks for nobiliary particles that indicate a family name: von, van, de, and so on. If that succeeds, it removes suffixes from the family name, thereby separating all the components. If there is no nobiliary particle, then the package first removes suffixes from the whole name, and takes whatever is the last word in the remaining name as the family name.

8.2(b) Name suffixes. When no braces delimit the suffix, the package looks for common suffixes (Jr., Sr., and roman numerals through X). The first such suffix found is treated as a name suffix.

To see this automatic parsing in action, consider the following name input, which would be parsed as follows:

```
author={Georg von Trapp Jr., Family Singers}
```

`Georg`	Given name
`von Trapp`	Family name, based on the nobiliary particle “von”
`Jr.`	Name suffix, as it is on the list of common suffixes
`Family Singers`	Institutional affiliation, as it follows a comma

8.3 Formatting

Once names have been read and separated into component parts, the package arranges and assembles those names into text amenable to inclusion in a citation. There are two matters to address here: Formatting the text of each name, and arranging and punctuating the list.

In terms of name formatting, the primary task is to abbreviate institutional names and affiliations. As a convenience, the package also checks that personal names “look” correct, in that they should not contain institution-like words such as “and” or “Corp.”

With regard to arrangement and punctuation, the following rules apply. First, as noted above, the list of names is separated into two sublists: a list of personal names and a list of institutional names. For each sublist, the last name is delimited by an ampersand and every other name is delimited by a comma. An “et al.” can apply to either the personal name list or the institutional name list. Finally, if both lists contain entries, they are joined by a comma. For example:

- Author One, Author Two & Author Three (a coauthored work)
- Author One et al. (alternative to above example)
- Author One et al., Inst. (a work written by several people at a single institution)
- Author One, Inst. et al. (a work written by one person on behalf of several institutions)
- Author One & Author Two, Inst. One & Inst. Two (a joint report)

8.3(a) Using Partial Lists. Some citation systems place the institutional names in a different place from the personal names. This package provides infrastructure for separating out these sublists, although it is not used in the package itself.

9 Dates

Nearly every reference cited in a legal document must be dated.[4] *Hereinafter* supports a wide variety of date specifications, in view of the many types of legal documents and the frequency with which historical documents are cited in legal research.

Typical legal citation systems have specific rules about the form of date to use with different reference types. *Hereinafter* does not enforce such rules, instead using a single date syntax in all places where dates are used. As a result, it is left to the discretion of writers to choose, for example, whether to include an exact date or just a year for any reference.

Dates may be entered in a variety of formats, so long as the month is spelled out in letters. The key additional feature of *Hereinafter* dates is support for "qualifier words," namely phrases that precede the date and explain its relevance. Qualifier words are considered part of the date and thus may be included in any place where a date is allowed; indeed, qualifier words are required in certain contexts.

9.1 Data Model

A date contains the following components:

- A year. This is the only required component. The phrase "n.d." can be used for undated references. It may be a range.
- Qualifier words preceding the date and explaining its relevance to the reference. For example, "submitted for ratification Sept. 17, 1787" includes qualifier words "submitted for ratification" that explain the relevance of the given date. Typical qualifier words are verbs ending in "ed," though any text preceding a number or keyword will be considered part of the qualifier. Qualifier words are useful, for example, when citing to cases still in litigation (e.g., "cert. granted, ⟨*date*⟩").
- A month and date number. They may also be ranges. The month can also be a season.
- Modifiers such as "c." for circa, "A.D.," "A.C.E.," "B.C.", and "B.C.E."

9.2 Input Syntax

Dates are entered, generally as parameters to references, as follows:

⟨*date*⟩ := [⟨*qualifier*⟩] (⟨*date-range*⟩ ‖ ⟨*date-word*⟩ ‖ ⟨*modifier*⟩)*
⟨*date-range*⟩ := ⟨*date-word*⟩ - ⟨*date-word*⟩
⟨*date-word*⟩ := ⟨*month*⟩ ‖ ⟨*day*⟩ ‖ ⟨*year*⟩

Examples:

```
2009
sept-oct 1980
adopted july 4 1776
c. 538 BC
```

[4] The main exceptions are for in-force codified statutes, assumed to be dated as of the time of the writer's publication, and well-known timeless works for which a date is immaterial, such as the Bible, Blackstone, Shakespeare, and the *Federalist* papers.

In other words, a date consists of qualifier words followed by one or more tokens that may be either date words or ranges of date words.[5] Date words of letters are interpreted as months or modifier words, while date words of numbers are interpreted as days or years. Punctuation is ignored other than dashes, used for ranges. Modifiers such as "A.C.E." and "n.d." should be entered with no dots (i.e., `ACE`, `ace`, `nd`).

Alternately, a date beginning with `!` will be used unformatted.

9.2(a) Permissible Words The following are permissible words in dates.

Date Word Input	Output
`jan, Jan, January, january`	Jan.
`feb, Feb, February, february`	Feb.
`mar, mar, Mar, March, march`	Mar.
`apr, apr, Apr, April, april`	Apr.
`may, May`	May
`june, June, jun, Jun`	June
`july, July, jul, Jul`	July
`aug, Aug, August, august`	Aug.
`sept, Sept, sep, Sep, september, September`	Sept.
`oct, Oct, october, October`	Oct.
`nov, Nov, november, November`	Nov.
`dec, Dec, december, December`	Dec.
`winter, win, Winter, Win`	Winter
`summer, sum, summ, Summer, Sum, Summ`	Summer
`fall, Fall`	Fall
`spring, spr, Spring, Spr`	Spring
`c, circa`	c. [circa]
`nd`	n.d. [no date]
`ad, AD`	A.D.
`bc, BC`	B.C.
`ace, ACE`	A.C.E.
`bce, BCE`	B.C.E.

The actual text for the latter few abbreviations can be redefined with `\AD`, `\BC`, `\BCE`, and `\ACE`.

[5] Technically by this syntax, a range can consist of two unrelated date words—⟨*month*⟩-⟨*day*⟩ for example. However, the parser may interpret that construction in unexpected ways.

10 Cross-References

Hereinafter offers comprehensive support for cross-references in citations. The model is conceptually similar to LaTeX, in which the writer uses commands to label parts of the document and then references those labels. However, legal citation requires and *Hereinafter* supports much more complex cross-referencing. It is possible to label ranges of a document in addition to specific points, and the package automatically determines the best wording to describe the labeled regions. More importantly, use of cross-references is integrated into the citation system, so cross-references can include signals, parentheticals, and other features of citations.

10.1 Types of Cross-References

A cross-reference citation points to another part of the document, but the writer must specify whether that other part is a page, footnote, section, or other element of the document. Thus, the package must support different types of cross-references. The following types are currently supported.

- Points in or ranges of body text. In legal memoranda, these are cited by the page numbers where the range lies. For law review articles, they are cited with reference to the footnote numbers that the range encompasses.
- Footnotes, cited by the footnote number or range of numbers.
- Sections, cited by the fully qualified section number (e.g., "Section II.A.3").
- Floats (tables or figures), cited by the float type and number (e.g., "Figure 3").

10.2 Labeling Document Parts

The following commands are used to mark parts of a document with labels.

`\hilabel[⟨text⟩]{⟨label⟩}` labels the current point in text. It detects whether it is inside a footnote, table, figure, or body text, and adjusts the label accordingly.

`\hiendlabel{⟨label⟩}` closes a labeled range opened with `\hilabel`. Note that the type of text marked at the beginning and end must be the same (e.g., `\hilabel` called in a footnote cannot be followed by `\hiendlabel` in body text). Currently, only contiguous ranges can be labeled, although a desirable additional feature for the package would be support for non-contiguous cross-reference ranges.

Particularly for law review articles, it is sometimes desirable to cross-reference "notes x–y and accompanying text." This is done with `\hitanlabel` and `\hiendtanlabel`. There are also provided commands `\hiflabel` and `\hinotelabel` for labeling floats and footnotes, respectively.

`\hisectlabel{⟨label⟩}` labels the current section. `\hiendsectlabel{⟨label⟩}` labels an end of a section range.

For law review articles, top-level cross-references will use the word "Part" while other cross-references will use "Section." (The distinction is determined by whether the section number contains a dot.) For legal memoranda, "Section" will always be used. To change this, redefine the internal macro `\hi@xref@sect@choose`.

10.3 Using Cross-References

The ordinary citation commands are used to produce a cross-reference citation, using the pseudo-reference this with the “page number” being the cross-reference label. For example, assuming that `section-i` is defined as a label:

```
\sentence{see this at section-i (providing background)}.
```

As this example suggests, cross-reference citation items can include parentheticals and signals, just like ordinary citation items.

Cross-references to floats and sections can also be used inline, producing text like “Figure *X*.” Again, the syntax mirrors inline citations for other references:

```
Data is shown in \inline{this at section-i}.
```

11 URLs

Every reference type can include a URL. *Hereinafter* formats URLs, makes them clickable links, and inserts line breaks into them. In that last respect, *Hereinafter*'s procedure is superior to other LaTeX URL packages, which fail when a hyperlinked URL in a footnote breaks across a page, a common occurrence for law review articles.

11.1 Input Syntax

The url and opturl parameters add a URL to a reference. The URL should be entered as unformatted text, except that percent signs should be escaped with backslashes to avoid being interpreted as comment symbols.

URLs will allow line breaks after main punctuation, similar to the package `url.sty`.[6] If there is a long string of alphanumeric characters, then this package may not find a suitable breakpoint. Including `\␣` (that is, a backslashed space) in the URL will force a breakpoint to be inserted into the URL, with no effect on appearance.

The command `\url{`⟨*url*⟩`}` can be used to format a URL in text.

11.2 Alternate Presentations of URLs

By default, URLs are shown in text. This format, logical for print publications with no other easy access to hyperlinks, can look lengthy and cumbersome in legal memoranda and other contexts. Accordingly, several other options are provided.

In documents with tables of authorities, URLs can be presented in an alternate format: The URLs are included only in the table listing, and citations in the document include a marker "*available online*." A footnote is added after the first such omitted URL, informing the reader that omitted URLs are in the Table of Authorities. To select this format, use the package option toaurl.

This format has the advantage of producing cleaner-looking briefs, particularly given the large font-to-text-field ratios that most courts require of briefs (such that URLs could take up a large fraction of a page). The main disadvantage is that it actually increases word count, since URLs typically count only as one word.

The text to be shown in place of URLs can be redefined with the macro `\ToaUrlMark`. The text to be shown in the footnote after the first such replaced URL can be redefined with the macro `\ToaUrlText`.

Alternatively, in documents intended to be used only electronically, URLs can be replaced with a word that is hyperlinked. To select this format, use the package option linkurl.

The text to be used in place of the URL is defined in the macro `\LinkText`. No line break protection is performed on the word, so it should be an unbreakable word or the macro should be defined to include an unbreakable `\hbox`.

[6] An exception is for the percent sign, where the line break will be inserted *before* the symbol, so that URL-encoded entities are kept intact.

12 Context-Dependent Parameters

For some references, different parts need different parameters. For example, multivolume treatises often have different authors per volume, and some old multivolume works were published over multiple years. One possibility would be to cite each volume as a different reference, but that would mean that different volumes of the same work would not receive short or *id.* citations.

As a result, the better option is to allow these complex works to be defined as a single reference, but enable an option for changing parameter values depending on the part cited. The struct parameter enables this.

The value passed to struct is a key-value listing where the keys are the determinative document component (typically but not always the volume number), and the values are another key-value listing of replacement parameters. For example:

```
\defbook{treatise}{
  author=A A,
  title=The Treatise,
  year=1800,
  struct={1={author=B B},
          2={title=The Treatise Revised},
          3={year=1805}},
}
```

Here, the default citation would be "A A, The Treatise (1800)." But if volume 1 is cited then the author would be changed, the title changed if volume 2 is cited, and the year changed if volume 3 is cited.

Only certain reference types support parameter structures, and the particular parameters that can be replaced are determined by the reference type.

13 References Within References

Often a shorter work will be included in a larger compilation, and both need to be cited in order to facilitate access to the shorter work. A private letter may be reprinted in a volume of collected letters, for example, or a brief filed in a case may appear within the docket for a case. *Hereinafter* provides a number of ways to support this.

First, some reference parameters accept another reference nickname as input. All references accept the parameter reprinted, and some such as casedoc accept the parameter citation. More generally, the reference type citecontainer provides a comprehensive mechanism for citing a reference contained within another reference.

As an example, consider the following reference definitions:

```
\defletter{madison-letter}{
    author=James Madison,
    to=W.T. Barry,
    date=aug 4 1822,
    reprinted=madison-writings,
}
\defbook{madison-writings}{
    name=Letters and Other Writings of
         James Madison,
    year=1884,
}
```

The parameter reprinted in the letter's reference definition identifies the book in which the letter has been reprinted by its reference nickname, and the book is defined separately as a reference. This approach works perfectly well, but it can be cumbersome to define the book as its own reference with its own nickname. If the book is never used anywhere else, it might be convenient to define the book reference inside the letter reference definition.

"Anonymous references" allow for this. Instead of providing a reference name as the value to reprinted, the book can be included in the letter reference definition as follows:

```
\defletter{madison-letter}{
    author=James Madison,
    to=W.T. Barry,
    date=aug 4 1822,
    reprinted=book: {
        name=Letters and Other Writings
             of James Madison,
        year=1884,
    },
}
```

More generally, anonymous reference definitions consist of the reference type (without the `\def` prefix), a colon and space, and a set of parameter-value pairs in braces. Note that whitespace following the parameter-value pairs can confuse the parser, so it is best to place a comma after the closing brace as shown above.

Part III
Citations

A citation to a reference in a document consists of a *citation command* followed by a *citation string* as the argument to the command. Citation commands are described in section 14, and citation strings are explained generally in section 15. The remainder of this Part then describes individual components of citation strings in more detail.

14 Citation Forms and Commands

Depending on grammatical context, a legal citation in a work will take on one of several forms. *Hereinafter* supports these different grammatical forms of citation by offering several citation commands in this section.[7] This section describes each citation form along with the command used to access the form.

14.1 Sentences

A *sentence*-form citation is set off from the text as a separate sentence:

```
Segregation is unconstitutional. \sentence{see brown at
495}.
```

Produces:
Segregation is unconstitutional. *See Brown v. Bd. of Educ.*, 347 U.S. 483 (1954).

As seen above, `\sentence` produces this style of citation. The trailing period is mandatory; it avoids mistakes and assists word processors' detection of missed capitalization. In legal memoranda, the command may appear anywhere in a document. In law review articles, `\sentence` must appear in a footnote, and if it appears in body text, it will be automatically converted to `\note`. However, a sentence-style citation at the beginning of a paragraph in body text (typically used to cite a block quotation) cannot be automatically converted, so a warning is generated.

14.2 Clauses, Instructions, and Nested Citations

A *clause*-form citation falls within a sentence, preceded by a comma and followed by a comma unless there is other punctuation. Typically clause-style citations are used when different citations are needed for different parts of a single long sentence.

```
Education is important, \clause{see brown at 493}, and
segregation has a ``detrimental effect,'' \clause{brown at
494}.
```

Produces:
Education is important, *see Brown*, 347 U.S. at 493, and segregation has a "detrimental effect," *id.* at 494.

The `\clause` command makes a clause-form citation. It should be surrounded by appropriate punctuation, and in particular there must be punctuation immediately following the command. In law review articles, `\clause` must appear in a footnote, and if it appears in body text, it will be automatically converted to `\note`. However, a clause citation followed by punctuation that is not a comma cannot be automatically converted, so a warning is generated.

14.2(a) Instructional Citations Citations may also appear in instructional sentences that end with a `\clause` citation. Typically these will be imperative sentences with an unitalicized spelled-out signal:

[7] *Hereinafter* also produces a distinct citation form for tables of authorities, but writers do not use this citation form directly.

```
On scientific evidence, see \clause{brown at 494 note 11}.
```

Produces:

On scientific evidence, see *Brown*, 347 U.S. at 494 n.11.

Although this context differs grammatically from other `\clause` citations, the formatting is largely identical so only one command is provided.[8]

14.2(b) Nested Citations Citations may also appear nested in parentheticals within other citations. This can be accomplished by using `\clause` inside the parenthetical:

```
\sentence{see brown at 494-495 (overruling
\clause{plessy})}.
```

Produces:

See Brown, 347 U.S. at 494–95 (overruling *Plessy v. Ferguson*, 163 U.S. 537 (1896)).

More on nested citations, including a more convenient shortcut form, may be found in section 19.

14.3 Inline Citations

An *inline*-form citation acts as a part of speech within a sentence, typically a noun or an adjective. Naming a case in text uses an inline citation:

```
\Inline{brown} followed \inline{sweatt}.
```

Produces:

Brown followed *Sweatt v. Painter*.

The following commands produce inline citations. Any of them may be used in text or footnotes, regardless of the document type.

- `\inline` makes a typical inline citation.
- `\Inline` capitalizes the first letter of the citation.
- `\adjective` removes leading "the" from the citation text, making it amenable to use as an adjective in a sentence.
- `\Adjective` is like `\adjective` but capitalizes the first letter.

Hereinafter's support for inline citations is the most comprehensive for cases and statutes, where there are reasonably well-defined conventions for how to form such citations. For other reference types, such as books and articles, the package uses author names to the extent given. Support for more complete coverage of inline citations is a potential future improvement to the package.

Using inline citation commands is preferable to manually entering inline citations in text for several reasons. First, inline citations often have long and short forms, so programmatic formatting of inline citations ensures that the right one is chosen even

[8] Some style guides use a slightly different format in which citation items are separated with commas and the word "and" rather than semicolons. This is a somewhat rare usage that could be accomplished by issuing separate `\clause` commands for each citation item, so there seems to be no particular need to implement a separate command for this use case.

if paragraphs or sections of text are moved around. Furthermore, inline citations can affect the surrounding non-inline citations. When citing a case for the first time, it is typical to omit the case name from a sentence citation when the case name was used inline:

```
\Inline{brown} involved class actions. \sentence{see brown
at 495}.
```

Produces:
Brown involved class actions. *See* 347 U.S. at 495.

This package detects such situations and removes duplicative names automatically.

14.3(a) Inline Citations and *Id.* One of the most nuanced considerations for inline citations is the effect on *id.* short forms. Consider the following:

```
\sentence{brown at 496}. The Court overruled
\inline{plessy}. \sentence{brown at 494-495}.
```

Produces:
Brown, 347 U.S. at 496. The Court overruled *Plessy*. **???**

Absent the inline citation to *Plessy*, the final citation in the above sentence should be "*Id.* at 494–95." But with the inline citation, it is ambiguous which case the *Id.* refers to. The better option, which *Hereinafter* follows, is to use the named short form.

If this is undesirable, one can cause the *Id.* form to be used by hiding the inline citation with braces:

```
\sentence{brown at 496}. The Court overruled
{\inline{plessy}}. \sentence{brown at 494-495}.
```

Produces:
Brown, 347 U.S. at 496. The Court overruled *Plessy*. *Id.* at 494–95.

By contrast, consider:

```
\sentence{brown at 496}. The Court applied
\inline{14th-amend}. \sentence{brown at 494}.
```

Produces:
Brown, 347 U.S. at 496. The Court applied the Fourteenth Amendment. *Id.* at 494.

Here, it is unambiguous that the *Id.* does not refer to the constitution. *Hereinafter* implements this distinction by designating the inline forms of statutes, constitutions, and a few other references as "invisible" to the *id.* citation algorithm. This behavior can be overridden by using the `\noid` command.

14.4 Note

`\note` makes a sentence-style citation inside a footnote; it is essentially a synonym for:

```
\unskip\footnote{\sentence{⟨string⟩}.}
```

It provides two conveniences for the author. First, the `\unskip` means that a space before `\note` is automatically removed, making TeX code more readable. Second, `\note` can take a second argument of text (which can include sentence and clause citations) appended after the citation:

```
\note{case}{For further information, see \clause{case2}.}
```

14.5 Optional Clause

The `\optclause` command is placed immediately after an `\inline` citation and takes no arguments. Its purpose is to serve a style of citation that some writers prefer, in which a clause-style citation (in text or in a footnote, depending on the type of document) immediately follows the first inline citation to a reference:

```
In \inline{brown}\optclause, the Court held...
```

Produces:
[When case already cited] In *Brown*, the Court held...
[When case not already cited] In *Brown v. Board of Education*, 347 U.S. 483 (1954), the Court held...

In other words, this is identical to `\inline{ref}, \clause{ref at 570}`, if the reference has not been cited so far, but produces nothing if it has been cited already.[9]

14.6 Miscellaneous Commands

A few additional commands are defined here.

Ellipses in legal documents are idiosyncratic, using full spaces between dots. The package redefines ellipses in LaTeX to conform.

The `\sic!` command inserts a properly formatted [*sic*] into text. The exclamation point deals with TeX spacing issues.

Several macros affect memory of citation state to change what forms of citations will be produced.

- The `\noid` command informs the package not to use *id.* for the next citation, by resetting memory of the last reference cited.
- Similarly, the `\fullcite{`⟨*ref*⟩`}` command informs the package to produce a full citation of ⟨*ref*⟩ by resetting memory of that reference's use.
- By contrast, the `\nofullcite{`⟨*ref*⟩`}` effectively cites ⟨*ref*⟩ without displaying anything, so that the next citation to that reference is a short form.
- The `\fulltitle{`⟨*ref*⟩`}` command resets memory of the last statutory title cited.
- The `\addtotoa{`⟨*ref*⟩`}` command adds ⟨*ref*⟩ to the table of authorities without producing any citation text.

14.7 Shorter Commands

The following are synonyms to the citation commands thus described.

[9] Personally, I do not like this style of citation, as it breaks the rule that a citation comes after the proposition being cited. But it is a common enough style that I provide it as an option.

Command	Synonym
`\sentence`	`\Ct`
`\clause`	`\ct`
`\Inline`	`\Ci`
`\inline`	`\ci`
`\note`	`\cn`
`\addtotoa`	`\cx`

15 Citation Strings

To specify the information needed to cite references in a document, this package provides a compact but flexible input syntax intended to capture the many types of information that may be included in a citation. This section first describes a data model for the information that goes into a citation, and then explains the syntax for input text that this package uses to receive that information. The input text syntax is used for all citation commands that this package accepts (`\sentence`, `\clause`, `\inline`, etc.).

15.1 Data Model

A *citation string* is an ordered list of one or more *citation items*.[10] Each citation item includes the following components:

- A signal (see section 16).
- A reference name.
- A volume number.
- A pin cite (see section 17).
- An "optional argument," used by some reference types to fine-tune the citation display.
- A list of parentheticals (see section 19).

All of the components are optional other than the reference name. The syntax and meanings of signals, pin cites, and parentheticals are treated in other documentation sections; the reference name, volume number, and optional argument are freeform text that is used as-is with no processing.

15.2 Input Syntax

Citation strings are given to the package using the following syntax:

```
⟨citation-string⟩ := ⟨citation-item⟩ ( ; ⟨citation-item⟩ )*
⟨citation-item⟩ := [ ⟨signal⟩ ]
      [ ⟨volume⟩ ]
      ⟨reference-name⟩
      [ [ ⟨opt-argument⟩ ] ]
      [ at ⟨pincite⟩ ]
      ( ( ⟨parenthetical⟩ ) )*
```

More concretely, citation strings are one or more semicolon-separated items. The following cites three references:

```
ref-a; ref-b; ref-c
```

A complete citation item with every component might look like this:

[10] Multiple citation items are permitted even for citation command types like `\inline` that ought to accept only one item.

```
see, e.g., 5 treatise[opt] at 35-60 (citing references)
(outdated)
```

This would cite volume 5, pages 35–60 of a reference named `treatise` with the signal `see, e.g.`, an optional argument `opt`, and two parentheticals.

More commonly, only a few components of a citation item will be included. An example citation string in ordinary use might look as follows:

```
statute at S 167; see also supporting-case at 189;
another-case at 465; cf. journal-article at 754 (discussing
related applications of the statute); see generally 7
treatise at S 56
```

Note that if the same reference is cited multiple times in a row, there is no facility for writing *id.* or the like in the input syntax. The reference must be fully identified and pin-cited on each use, and the package will automatically compute whether *id.* should be inserted as described in section 26. While this may seem like unnecessary work, there is a major advantage: The citation string input is fully independent of where in the document the citation string appears. As a result, writers can freely move blocks of text around a document without having to rewrite input citation strings. If *id.* or other shortcuts had been used in the document, then they likely would become unintelligible upon a document reorganization.

16 Signals

Items in a citation string may be preceded by a signal, indicating how the writer of the citation understands the relationship of a cited reference to the writer's argument. A citation preceded with *see* indicates that the reference supports the writer's proposition; *compare* can identify warring sides of a debate; *cf.* can identify cleverly related connections or coyly disguise contradictory authorities; and *see generally* can point to background reading. Signals introduce a great deal of expressiveness into a system of legal citation, enabling writers to compactly distill a complete literature into a single citation string.

16.1 Data Model (List of Signals)

Each signal belongs to one of four classes: support, comparison, contrary, and background. Here are all the signals available, along with their classes.

Signal	Class
see	Support
see, e.g.,	Support
see also	Support
see also, e.g.,	Support
e.g.,	Support
accord	Support
accord, e.g.,	Support
cf.	Support
cf., e.g.,	Support
compare	Comparison
compare, e.g.,	Comparison
and	Comparison
with	Comparison
with, e.g.,	Comparison
contra	Contrary
contra, e.g.,	Contrary
but see	Contrary
but see, e.g.,	Contrary
but cf.	Contrary
but cf., e.g.,	Contrary
see generally	Background
see generally, e.g.,	Background

16.2 Input Syntax

As described in section 15, the signal for a citation item is placed at the beginning of the input text for the citation item, in all lowercase text. As a convenience, duplicative sequential signals may be omitted. For example, the following are equivalent:

```
see ref1; ref2; see also ref3; ref4
see ref1; see ref2; see also ref3; see also ref4
```

In both cases, the *see* signal will apply to `ref1` and `ref2`, and *see also* to `ref3` and `ref4`.

This package does not deal with two aspects of citation signals. First, it does not enforce an order in which signals must be used. This provides slightly more flexibility to writers, although it is expected that most writers will follow conventional ordering of signals. Second, comparison signal citations typically include *and* before the penultimate item of each side of the comparison, for example:

> *Compare* A, B, *and* C, *with* D, *and* F.

For implementation reasons,[11] this package does not add the *and* automatically. Instead, *and* is defined as a comparison signal for the writer to insert in the appropriate place in a citation string.

16.3 Formatting

There are four matters to be handled with respect to formatting of signals.

1. The citation must be formatted with the correct fonts.
2. The punctuation between citation items depends on the surrounding signals. Citation items are generally separated by semicolons, but with two exceptions:
   - Between citation items using comparison-class signals, commas are used.
   - In a sentence-style citation, citation items are separated with a period if the signal class changes. The signal is also capitalized.
3. If the same signal is used two or more times in a row, it is only displayed on the first use.
4. In a series of citation items with contrary-class signals, the word “but” is omitted from all but the first negative signal.

[11] As described in the implementation discussion of citation formatting, each citation item is formatted by executing a group of commands for the citation. To figure out whether an *and* is needed in the middle of a comparison citation string, it would be necessary to read ahead to the next citation item in advance. While this could be done, it would be cumbersome to program, and the workaround described here is not enough of an inconvenience to make the effort seem worthwhile.

17 Page Numbers and Other Pin Cites

The often extraordinary length of legal documents, articles, and treatises necessitates citing specific portions of them in order to inform readers of the relevant parts of a reference that support the writer's claims. Yet the diversity of forms of legal references makes pin-citing the relevant parts a challenge. Some documents are paginated, others divided into sections, others marked with star-pagination reflecting archaic publications. And some forms of citation require identifying multiple pin cites for different published versions of a work—cases reported in several reporters, for example, or for session laws both the statutory section number and the page number of the *Statutes at Large*.

It is not desirable to put the burden on the user to manage formatting of page numbers and subdivisions. A legal citation system must understand users' page number inputs to compose tables of authorities, determine whether to omit page numbers in *id.* citations, and cite subsections of statutes, for example. Furthermore, there are many rules of pin-cite formatting that are inconvenient and burdensome for writers to remember and better suited for automatic processing—abbreviation of words like "paragraph" and italicization of "(*l*)," for example. As a result, a program such as this package must parse and interpret pin cites, and it thus must provide writers with a syntax for entering them.

This section describes the data model for pin cites, the input syntax that this system accepts for them, and the rules for formatting.

17.1 Overall Structure

Because of the diversity of types of legal document numbering systems, the data model and syntax for pin cites are fairly complex. For testing purposes, a writer may access the pin cite formatting algorithm using the command `\PageNumber{`⟨*number*⟩`}`.

17.1(a) Segments A pin cite is made up of several "segments": a default segment and one or more named segments. The named segments are used for parallel citations; their use is defined by specific reference types. For example, session laws use the default segment to hold the subsection number of the statute being cited, and then require a segment named `stat` that identifies the page of the *Statutes at Large* on which the subsection appears. The input syntax and an example are as follows:

⟨*pin-cite*⟩ := ⟨*segment*⟩
    (␣`::` ⟨*segment-name*⟩ `:`␣ ⟨*segment*⟩)*

Example:

```
S 12 ::stat: 783
```

The example would cite § 12 of a statute, at page 783.

17.1(b) Lists of Items A pin cite segment can consist either of a comma-separated list of items, or two comma-separated lists of items delimited by the word `to` (surrounded by spaces). The latter form is used for pin cites to ranges where each side of the range contains multiple items.

⟨*segment*⟩ := ⟨*list*⟩ [_to_ ⟨*list*⟩]
⟨*list*⟩ := ⟨*item*⟩ (_,_ ⟨*item*⟩)*

Example:

```
column 5, line 8 to column 6, line 5
```

Here, the word “to” must be used to separate the ranges to avoid ambiguity.

17.2 Pages and Divisions

Many forms of pin cite items are permitted, and how an item is interpreted may depend on what item preceded it. In general, there are two types of items in pin cites: page numbers and named divisions (like “table 5” or “§§ 15-20”). The number type is important for formatting: Ranges in page numbers are truncated, and named divisions need to be singular or plural and can accept subdivisions.

Generally, page numbers are written as bare numbers, and named divisions are written as the division name, a space, and one or more division numbers. A division number can follow a page number with or without a comma, or with an ampersand (indicating a citation to the page and the division). After a named division is used, any subsequent bare numbers will be interpreted as part of the division. To return to citing page numbers, insert the word `at` (e.g., `section 5, at 143`). This can be useful for helping readers find the location of text divisions. Thus:

Input	Output	Explanation
`157, 160-163`	157, 150–63	Page numbers
`line 403, 450-463`	ll. 403, 450–463	All are line numbers
`line 403, at 450-463`	l. 403, at 450–63	450–463 are pages
`157-158 table 1-2`	157–58 tbls.1–2	Citing just tables
`157-158 & table 1-2`	157–58 & tbls.1–2	Citing pages and tables

As seen in the above examples, the number part of either a page number or a named division can be a hyphenated range, and the hyphen is converted to a proper dash. To include an actual hyphen, use `\-` instead.

Further fine-tuning of the pin cite parsing algorithm can be done by placing special codes or punctuation at the beginning of pin cite items:

Input	Type	Explanation
`S 3`	Division	Produces § 3
`P 3`	Division	Produces ¶ 3
`-34/a`	Division	Interpreted as a division with no name: 34(a)
`!A5`	Number	Uses text after the exclamation point as a bare number, and does not format at all. Useful for newspaper articles
`*16`	Number	Treats whatever follows the star as a page number. Useful for star-paginated works
`?`	Missing	Issues a warning to remind the writer to fill in a pin cite
`/a3`	Division	Interpreted as a subdivision: (a)(3). Must follow a named division

17.3 Subdivisions

Named divisions of text are sometimes organized into top-level divisions and subdivisions. For example, “§ 5(a)” refers to subsection (a) of the fifth section of a reference. Distinguishing between top-level divisions and subdivisions is necessary because it affects pluralization of the division name (“§§ 5–6,” but “§ 5(a)–(b)”). Additionally, writing all the parentheses in conventional subdivision numbers is somewhat cumbersome.

For input to this package, subdivisions are separated from top-level divisions with a slash. Bare subdivisions (without a top-level subdivision) may be used as pin cite items to cite multiple non-contiguous subdivisions, and ranges may include bare subdivisions. Single characters in a subdivision will be automatically parenthesized. The following are examples:

Input	Output
`S 5/a`	§ 5(a)
`S 5/a-6/b`	§§ 5(a)–6(b)
`S 5/a-/b`	§ 5(a)–(b)
`S 5/a, /c`	§ 5(a), (c)
`S 5/a1A`	§ 5(a)(1)(A)
`S 5/a1A(iii)`	§ 5(a)(1)(A)(iii)
`S 5/a1A{note}`	§ 5(a)(1)(A)note

18 Pin Cite Subdivision Joining

For alias and some statute citations, it is possible to give an overall division or page number for the reference and then refer to a specific part of it. For example, one might define the statute 35 U.S.C. § 112 as a reference `sec-112`, and then wish to refer to subsection (b) of that statute. To do so, one may write:

```
\sentence{sec-112 at /b}
Equivalent to: \sentence{sec-112 at S 112/b}
```

Produces:

35 U.S.C. § 112(b)

In other words, the pin cite given as part of the citation item is joined to the pin cite given in the reference definition. The following rules define how a reference pin cite is joined with a citation item pin cite:

- If the citation item pin cite does not start with a slash indicating a subdivision, then the reference pin cite is discarded and only the citation item's is used.
- If the citation item pin cite starts with a slash and the reference pin cite has no slash, then the two are concatenated.
- If the citation item pin cite starts with a slash and the reference pin cite has a slash, then the slash is removed from the citation item pin cite and the two are concatenated.

Thus, the following examples:

Reference	**Citation Item**	**Result**
`S 112`	`S 102/a`	§ 102(a)
`S 112`	`/b`	§ 112(b)
`S 505/j`	`/2A`	§ 505(j)(2)(A)
`S 505/j`	`/2A-/3`	§ 505(j)(2)(A)–(3)

The range joins correctly not because of these rules but because of the ordinary parsing rules for subdivision ranges, given above.

Note that the joining algorithm performs no error checking, and will join subdivisions to page numbers even though subdivisions make no sense in that context. Additionally, this feature is only supported by certain reference types that enable it.

18.1 Formatting Around Pin Cites

Pin cites can affect the text that precedes the pin cite number. In many situations, standard page numbers are often preceded by the word "at" whereas named subdivisions are not. The macro `\hi@atorsect{`⟨*pincite*⟩`}` is provided to insert "at" depending on the nature of ⟨*pincite*⟩, and `\hi@page@atorsect` performs that function on the currently active pin cite in a citation item.

For other citation types such as books, the pin cite is separated from text in the citation with just a space. However, there are exceptions:

- If the preceding text ends with a digit, then a comma is placed, and "at" is placed if the pin cite is a page number as described above.
- If the preceding text is "R." (for "Record"), then "at" is placed regardless of the type of pin cite.[12]

Thus, the following examples:

Preceding Text	**Pin Cite**	**Output**
`Book`	`15`	Book 15
`Treatise`	`S 5`	Treatise § 5
`Annual Report 2015`	`35`	Annual Report 2015, at 35
`Agency Report 2006`	`S 12`	Agency Report 2006, § 12
`R.`	`102`	R. at 102
`R.`	`paragraph 36`	R. at para. 36

Finally, it is sometimes useful to expand initial `\textsection` and `\textparagraph` symbols to words. This is useful for statute citations. The command `\hi@expand@symbols{`⟨*pincite*⟩`}{`⟨*callback*⟩`}` will do this.

[12] For Record cites to named subdivisions (e.g., "R. at para. 5"), it is unclear whether "at" should be included. It seems preferable to do so to ensure that "R." is not interpreted as "Rule."

19 Parentheticals

Parentheticals are information appended to the text of citation items, and are used to convey additional information about the reference being cited. The following example includes two classic examples of parentheticals, one for court and date information and the other for characterizing the holding of a case:

> *See* Plaintiff v. Defendant, 123 F.4th 567 (1st Cir. 2000) (holding that the sky is blue).

Legal citation uses parentheticals extensively, both in highly structured ways to include required bibliographic data about references and more freeform ways that allow authors an additional avenue to converse with readers.

This section describes the nature of parentheticals as understood by this package, how parentheticals are entered for citation items, and how parentheticals are formatted.

19.1 Data Model

19.1(a) Types At the time that a citation item is being formatted, the package begins building up a list of parentheticals. The parentheticals can originate from two places. First, they may be associated with the reference itself, providing bibliographic information or commentary to be generally associated with the reference (e.g., publication dates, judicial opinion types, or alternate sources for finding references). These parentheticals are typically displayed only with the full citation of the reference.

Parentheticals may be provided as part of the citation item input, as described in section 15. These "citation-level" parentheticals are generally meant to explain the writer's use of the reference in a particular citation—characterizing the holding of a case to demonstrate its relevance to the writer's argument, quoting the relevant portion of a reference, or identifying sources that the reference itself quotes or cites.

"Parentheticals" as used in this package are not merely text set inside parentheses. They also include items such as URLs, prior and subsequent case history, and other material typically set off with a comma after the main text of a citation item. Thus, there are three parentheticals in the following example for the date, the website, and a characterization:

> John Doe, A Book Online (2018), http://www.johndoe.xyz/book (providing information).

As this example shows, it is best to treat comma-offset text as parentheticals just like parentheses-enclosed text, because the two are intermingled in presentation. URLs, for example, appear after the date parenthetical to a citation but before parenthetical commentary text.

19.1(b) Content and Ordering Every parenthetical is associated with two items: (1) the text of the parenthetical, and (2) a priority number. The priority number serves two purposes. First, it establishes an ordering of parentheticals. For example, the date parenthetical for a citation always should precede other parentheticals, so it has the lowest priority number. Second, the priority number determines the appearance of the parenthetical. Other than hereinafter parentheticals (which unusually are set inside square brackets), parentheticals with even priority numbers are set inside parentheses while those with odd numbers are set off with a comma. As a result, there is consistency among even citations with many parentheticals.

The priority number for any parenthetical is set automatically based on the context in which the package receives the parenthetical. Generally writers should not need to deal with them, but the mechanism may be useful to know for authors who wish to change the default ordering or appearances.

19.2 Input Syntax

For parentheticals entered as parameters to references, the syntaxes are explained in Part VII. This section describes the syntax for parentheticals entered as components of citation items, expanding on section 15. These are called "citation-level" parentheticals.

Generally parentheticals are entered in citation items as plain text. Citation commands may be used inside parentheticals to produce nested citations, as described in section 14. In addition, several special syntaxes are provided to allow for different types of common parentheticals.

The following input will convert ⟨*cite*⟩ to `\clause{⟨cite⟩}` and remove the colon. This is a convenient shortcut for the common types of parentheticals that start with "quoting," "cited by," or "discussed in" followed by a citation.

```
(⟨verb⟩ing: ⟨cite⟩)
(⟨verb⟩ in: ⟨cite⟩)
(⟨verb⟩ by: ⟨cite⟩)
(⟨verb⟩ at: ⟨cite⟩)
(codified at: ⟨cite⟩)
(codified as amended at: ⟨cite⟩)
```

For example, the following are largely equivalent:

```
\sentence{ref at 3 (quoting: source at 15)}
\sentence{ref at 3 (quoting \clause{source at 15})}
```

In both cases, the citation to `ref` will include a parenthetical with the word "quoting" followed by a formatted citation to `source`.

The main difference between the colon forms and using `\clause` directly is that the package can set priorities of the parentheticals correctly. As currently defined, the codification parentheticals will come first and subsequent commentary (using `in`, `by`, and `at`) comes last, with ⟨*verb*⟩`ing`-type parentheticals and other citation-level parentheticals in the middle.

19.2(a) Deferred Footnotes A parenthetical starting with `footnote:` creates a "deferred footnote" inside a citation item. As its name suggests, the footnote is deferred for placement until after the citation item has been formatted and emitted to the document. If the citation item appears in body text, then the deferred footnote is placed after the separating punctuation at the end of the citation item. If the citation item is already in a footnote, though, the deferred footnote text is appended to the end of the footnote. This mechanism is useful primarily for legal memoranda if there is some material that should be connected to a citation in the middle of a citation string, where that material would be too unwieldy to include in a body-text parenthetical.[13]

[13] The deferred footnote mechanism is also used to deliver a footnote about URLs being in the Table of Authorities when that option is requested, as described in section 11.

19.3 Formatting

The main task in formatting parentheticals is distinguishing among the different types. As discussed above, there are three parenthetical forms:

- Parentheticals surrounded by parentheses. Emitting these is straightforward.
- Parentheticals offset by a comma. The primary additional consideration here is that punctuation at the end of the parenthetical will be the last punctuation in the citation, so if the parenthetical ends with a dot then it is necessary to remove duplicative periods at the ends of sentences and such.
- Hereinafter parentheticals. In addition to using square brackets, we record use of the hereinafter form as described in section 22 and possibly suppress the parenthetical if the hereinafter form was never otherwise used.

Part IV
Additional Features

20 Table of Authorities

Legal briefs and other documents often include a Table of Authorities, listing all of the references cited and the pages where those citations occur. One of the motivations for creating this package was the automatic generation of complete, correct, and properly formatted Tables of Authorities.

The table is generated using LaTeX's auxiliary file feature. Citations generate index lines that are written to a `\jobname.toa` file, and those lines are read, sorted, and formatted to produce the table. Indexing can be disabled with `\disabletoa` and enabled with `\enabletoa`.

A Table of Authorities is a complex structure, because it is typically sorted into sections based on reference types. This section describes how references are sorted into lists, how those lists get their headings, how page numbers in the tables are displayed, and finally how the table is formatted.

20.1 Table of Authorities Categories

Each reference type is associated with a category, used to organize the Table of Authorities. The category can be altered using the reference definition parameter toacat or by the `\SetTOACategory` macro. The categories are used to assemble the headings for the table.

Specifically, every category has three properties:

1. A name for internal use.
2. A "supercategory" that associates the category with others that this category should be joined with in the Table of Authorities. For example, the categories for statutes and regulations should have the same supercategory, so that they are placed in a single section with the Table of Authorities. Supercategories are presented in alphabetical order.
3. A textual description of the category, for use in constructing the Table of Authorities heading. Because both the singular and plural forms are needed, the textual description consists of two to three parts, separated by slashes:

   (a) The base text common to both forms.

   (b) [*Optional*] Text to append to the singular form only.

   (c) Text to append to the plural form only.

A new TOA category may be defined with `\NewTOACategory`. The macro takes four parameters, the first three corresponding to the three properties of categories above, and the last (being "Y" or "N") indicating whether to include the supercategory in the default table of authorities.

Each reference type may have a default category, set with the `\DefaultTOACategory` macro. The parameters are #1 the reference type, and #2 the category name. If the category name is blank, then there will be no default category. This default category will be used unless another category is specified for a particular reference, overriding the default category.

The `\SetTOACategory{`⟨*reference*⟩`}{`⟨*category*⟩`}` macro sets the TOA category for ⟨*reference*⟩. This macro should be called within the reference definer macros to set the category, or it is invoked by the toacat parameter. If it is called multiple times, then the last category will prevail. If it is not called, then the reference will not appear in the table. If ⟨*category*⟩ is empty, then any previously set category will be removed.

The following are the TOA categories, with their supercategories and display texts as described above. Note the choice of supercategory names, which coincidentally sort alphabetically in the correct order. Because the actual supercategory names are never displayed, any new supercategories can be named to fit into the existing ordering.

Category	Supercat.	Text	In TOA?
other	other	Other Source/s	Y
const	const	Constitutional Provision/s	Y
found	const	Foundational Document/s	Y
statute	edict	Statute/s	Y
regulation	edict	Regulation/s	Y
rule	edict	Rule/s	Y
case	case	Case/s	Y
admin	case	Administrative Decision/s	Y
treaty	edict	Treat/y/ies	Y
abbrev	abbrev	Abbreviation/s	N

20.2 Altering Categories in a Document

It may be useful to alter the TOA categories in a document, to create new categories or recategorize certain references or reference types. The macros described above allow for that. The following is a how-to guide on achieving a few such effects.

Place one reference in a different part of the TOA: use the toacat parameter when defining that reference.

Remove a reference from the TOA: use the notoa parameter, or set the toacat parameter to blank.

Place all references of a given type in a different part of the TOA (e.g., treat modelcode references like statutes): Use

```
\DefaultTOACategory{⟨type⟩}{⟨category⟩}
```

Example:

```
\DefaultTOACategory{modelcode}{statute}
```

where ⟨*category*⟩ is chosen from the table above.

Add a new TOA section (e.g., put film references in their own section): Use

```
\NewTOACategory{⟨category⟩}{⟨super⟩}{⟨text⟩}{Y}
\DefaultTOACategory{⟨type⟩}{⟨category⟩}
```

Example:

```
\NewTOACategory{m}{movies}{Motion Picture/s}{Y}
\DefaultTOACategory{film}{m}
```

where ⟨*category*⟩ and ⟨*super*⟩ are new names (not in the above table), and ⟨*text*⟩ is the text of the heading.

Add a new type to an existing TOA section (e.g., add “Model Code” to Statutes and Regulations): Use

```
\NewTOACategory{⟨category⟩}{⟨super⟩}{⟨text⟩}{Y}
\DefaultTOACategory{⟨type⟩}{⟨category⟩}
```

Example:

```
\NewTOACategory{mc}{edict}{Model Code/s}{Y}
\DefaultTOACategory{modelcode}{mc}
```

where ⟨*category*⟩ is a new name (not in the above table), {⟨*super*⟩} is the supercategory name corresponding to the existing TOA section, and ⟨*text*⟩ is the text of the heading.

20.3 Indexing References

References are indexed for the Table of Authorities upon every citation, by having entries added to the `\jobname.toa` file automatically for each citation command. The command `\addtotoa` described in section 14 can also add an entry manually.

An interesting challenge for indexing is that, especially for briefs, references may be cited in parts of a document that use different pagination systems. As a result, it is necessary to store detailed information about where a reference is cited in a document in order to produce a correct listing of pages in the table.

Five pieces of information are written out to the file for each use of a reference:

1. The reference name, and possibly the page for references where the page number is included in table of authorities entries.

2. The page number as formatted.

3. The page number as an arabic number.

4. The prevailing definition of `\thepage`, so that different page counting schemes can be identified.

5. A comma-separated list of "tags" under which the index entry should be categorized. Typically this will be up to two tags, one based on the reference's category and one based on the reference's location. A tag is a colon-separated structure comprising a supercategory name and a textual heading name (corresponding to the TOA Categories elements described in section 20.1).

20.4 Reading the Index

Prior to generating a table of authorities, the index file is read and its entries are sorted to produce three sets of data structures:

- For each supercategory, a list of references falling within that supercategory.

- For each supercategory, a list of textual headings for that supercategory.

- For each reference, a macro identifying the page numbers associated with that reference.

20.4.1 Heading Lists

For each supercategory, the textual description heading may comprise one or more types of references. For example, the "edict" supercategory may have statutes and/or regulations, and the heading should reflect the particular items to be listed. So if one regulation and several statutes have been cited, then the heading should read "Statutes and Regulation," for example.

To implement this, for every supercategory there is a "heading list" associated with the supercategory. The heading list contains every textual description associated with the supercategory, plus a flag for each textual description for whether it should be singular or plural. The first time a textual description is encountered for a heading list, the singular form is used. If the same description is encountered again, then it is updated to be marked as plural.

20.4.1(a) Location-Based Lists It may be desirable to prepare a chapter-by-chapter bibliography, such that references are tagged not by type but by location in the document. The macro `\AuthorityTag{`⟨*tag*⟩`}` will tag all subsequent references cited with the given ⟨*tag*⟩, which can then be used as a list of the Table of Authorities.

20.5 Composing Page Number Lists

If a reference is cited on pages 3, 4, 5, 7, 8, and 10, it is desirable to group these numbers into compact range forms for presentation in the Table of Authorities: "3–5, 7–8, 10." The package does so by reading the page numbers in order, keeping a running compilation of the properly formatted aggregate number listing and revising the last element of that listing as necessary.

20.5(a) Starred References Some courts ask litigants to insert asterisks in the Table of Authorities before references that they rely most heavily on. This can be done automatically by counting the number of times each reference has been used.

The macro `\StarCount{`⟨*num*⟩`}` sets the minimum number of uses of a reference before that reference receives a star. The package option toastar sets the value to 4. By default it is zero, in which case no stars are displayed.

The macro `\StarText` can be defined as text to place at the beginning of the Table of Authorities explaining the stars.

The macro `\StarMark` is the actual text of the star; it will be added before the relevant TOA entries via `\everypar`.

20.6 Showing the Table

The Table of Authorities is produced by calling `\tableofauthorities`. This will create one section for each reference category as described in section 6. Each heading will be created using whatever `\toaheading` is defined as.

Alternately, the macro `\AuthorityTable[`⟨*heading-text*⟩`]{`⟨*list-name*⟩`}` will produce a single Table of Authorities list. This can be used for more fine-grained control over what tables are displayed.

The following affect the appearance of the Table of Authorities:

- `\toaskip` is a glue specification for how much space to put between entries.

- `passimnum` is a counter for how many items an aggregate page listing must have before it is replaced with *passim.*

- `\PassimText` is the text to be used if `passimnum` is met or exceeded.

- `\toaditto` is the text that replaces repeated authors, statutory titles, and other information that can be omitted from the table listings.

- `\toahangindentlen` is defined as a length for the hanging indent. (It is specified in `em` units so it is defined as macro text.)

20.7 Table of Abbreviations

A table of abbreviations is like a simplified table of authorities, and because of *Hereinafter*'s support for abbreviations as the reference type abbrev, such tables can be made automatically. The macro `\tableofabbreviations` takes as an argument a two-element callback, which it executes for every abbreviation definition and full text. Entries are sorted by abbreviations, ignoring any "The".

21 Endnotes

The usual LaTeX endnote option stores the unexecuted text of notes and then executes them all at the end of the document. This does not work for legal citations because the content of the body text interacts with the content of the endnotes: An inline citation in body text can change the appearance of citations in the notes, and vice versa.

Instead, it is necessary to typeset endnotes as they are encountered and store them to a vertical box that can be displayed upon request.

`\EndnoteText{`⟨*text*⟩`}` adds arbitrary text to the endnote list. The text is set in vertical mode.

`\EndnoteTextDelayed` reserves a certain text for future adding to the endnote list. The text is actually added only if other endnote text is added before `\EndnoteTextDelayed` is invoked a second time.

This is useful for adding chapter or section heading markers to and endnote list conditionally. If the chapter contains no endnotes, then the chapter heading will never be inserted into the endnote list.

Again, the text is set in vertical mode.

`\SectionNote` inserts text into the endnote list indicating where a section heading occurred. The section number is inserted automatically, preceded by `\SectionName` from section 10.

`\EndnoteFont` can be defined to set the font for endnotes. It must be defined before endnotes are added, since once they are added the notes have been typeset and cannot be changed.

`\theendnotes` emits the list of endnotes. If the macro is not called, it will be automatically executed at the end of the document with a warning.

22 Short Names

Many types of reference use a short name to identify the reference after the first citation.[14] Cases use a distinctive party name, and books and articles use the authors' last names, for example. Writers can also choose their own short names for references using the `short` and `hereinafter` parameters.

Two problems can arise with these short names. First, two references may use the same text for a short name. In this case, the short name is ambiguous, so one or the other reference should use a different short name. Second, a writer may define a short name for a reference, say using `hereinafter`, but cite that reference only once in the document. A hereinafter parenthetical added to that first full citation is thus extraneous and unnecessary, and ought to be removed.

This package deals with both of these problems, by issuing warnings about ambiguous short names and automatically deleting unnecessary short name parentheticals. No user intervention is required other than to respond to warnings by adding distinctive short names.[15]

It is worth briefly noting that this short name management feature exemplifies the benefit of software-automated legal citations. A human cite-checker wanting to solve these two short name problems must keep a table of citations, to be searched and cross-referenced for every short name used. In a law review article citing hundreds or thousands of references, this task is tedious and error-prone, especially since the tabulation must be redone if the document is revised. For a computer, though, this task takes a matter of seconds.

[14] Short *names* are to be distinguished from short *citations*, which are described in section 26. A short name is text used to identify a reference the second and subsequent times, and that name may be used as part of short citations, but most short (non-inline) citations include further information, such as a *supra* note number or locator information.

[15] A possible future improvement would be to generate such short names automatically. To do so, though, it would be necessary to build into the package knowledge of conventional ways of disambiguating short names: Cases traditionally have roman numerals appended to them, while authored works typically have a distinctive word from the title added.

Part V
Formatting Algorithms

23 General Formatting

After parsing a citation string, the package must produce formatted text for each of the citation items in the string. This section describes the overall process of joining together citations, choosing among several possible citation forms for a given reference, and setting up any other requisite information for proper citation display.

An important element of citation formatting is keeping track of "state," namely information about references cited so far that is needed for proper formatting of later citations in the document. As a simple example, it is necessary to keep track of the previously cited reference so that if the next reference is the same, text such as *id.* can be written. This feature of legal citation is discussed in section 26.

23.1 Drawing a Citation Item

The following tasks must be performed when drawing each citation item within a citation string. While this list of tasks is largely of importance only to software implementers, it may be of interest to some cite-checkers to see the full list of considerations that go into formatting a single citation item.

1. Citation state must be updated; see section 26.

2. Each citation item maintains a list of parentheticals, which come from either the reference definition or the user's citation string content.

3. The citation item's components must be collected. Some references are able to modify the components afterwards.

4. The punctuation that precedes the citation item must be determined, according to the previous and current signals.

5. The short, long, or *id.* form of the citation must be chosen. That choice must be accounted for in computations for future citations.

6. If the citation is in a footnote, then how the citation affects this and other footnotes must be considered. See the discussion in section 26.

7. The citation must be recorded for indexing in a Table of Authorities.

8. If the citation text ends with a period, then any period following the item must be suppressed. This is obvious to human editors but surprisingly difficult for computers.

9. For a sentence-style citation, the first letter of each sentence must be capitalized. That could be the signal or a part of the citation text itself.

23.2 Citation Forms

When formatting a citation item, each reference can generate a variety of citation texts. The types are given and explained here, along with their internal macro prefixes:

Full (`fc`) The full citation form.

Short (`sc`) The non-*id.* short form. Defaults to the full form.

List (`lc`) The form used in the Table of Authorities. Defaults to the full form.

Pin cite (`pc`) Some references have special ways of treating pin cite numbers. This citation form is used in the pin cite formatting routine (see section 17).

Id. (`idc`) The form to be used for immediately successive citations, if not the standard *id.* form.

Inline full (`ifc`) The inline citation form to be used upon the first inline usage.

Inline short (`isc`) The inline citation form to be used on subsequent inline usages.

For each reference ⟨*ref-name*⟩, a macro of the form `\`⟨*prefix*⟩`@`⟨*ref-name*⟩ may be defined containing the relevant text for the citation form. For example, `\fc@brown` would be the definition of the full citation for a reference named `brown`. It may be helpful for debugging to see these macros. To do so, include the following in your document:

```
\expandafter\show\csname ⟨prefix⟩@⟨ref-name⟩\endcsname
```

23.3 Choosing the Citation Form

Upon each citation to the reference, it is necessary to choose among these forms. The following rules inform that choice.

- An *id.* citation is appropriate only when the reference being cited matches the last reference, recorded as described in section 26. In any event, it is never appropriate in an inline citation.

- The table of authorities form is chosen for non-nested citations inside a table of authorities. If no table of authorities form is present for a given reference, the full citation form is used instead. For nested citations inside a table of authorities, the full citation is always used.

- In other cases, a complex algorithm must be applied to choose between the short and full citation forms.[16] This algorithm is described in section 22. In any event, if no short citation form is defined for a reference, the full citation form is used.

- In an inline context, the same algorithm described in section 22 is used, except the inline full/short citation forms are preferred if they are defined.

In addition to choosing a citation form, the package sometimes allows a “nameless” variant of a citation form. For example, a citation to a case may omit the name of the case if the name was previously used in the text. In general, a nameless citation to a reference is permitted if the previous inline citation was to the same reference. There are some nuances to what constitutes the “previous” citation, as described in section 26.

[16] As described in section 22, the algorithm is complex because it depends on where previous citations were used: A full citation in a footnote, for example, does not justify a short form in main body text.

23.4 Missing References

Sometimes it is desirable to cite to a reference that has not yet been defined. This allows authors familiar with a reference to cite it without having to stop immediately to enter the reference definition and parameters. The package formats the undefined references in bold and also compiles a list of undefined references, which is displayed at the end of document compilation.

As a convenience, the package also keeps track of defined references that are missing page numbers. The user enters the unknown page number as a question mark, and the package displays a list of all unentered page numbers at the end of compilation. This tracking is turned on by default; if the `nomissing` option is given to the package then an error will be raised for missing reference definitions.

24 Abbreviations

Legal citations use a variety of abbreviation schemes for different types of texts. Case party names, court documents, and journal titles are not only abbreviated using different word lists, but with different abbreviation algorithms. Knowledge of these will generally not be important to writers, but they are discussed briefly below.

`\AbbreviateFor{⟨scheme⟩}{⟨text⟩}` abbreviates a text based on a given scheme. The schemes available are described below.

24.1 Standard Abbreviation Algorithm

The standard abbreviation algorithm is described in the package `abbrev.sty`, which is of general applicability. The rules applied there are as follows:

- There exist different classes of abbreviations, each of which contains a map associating full words with their abbreviated forms.
- To abbreviate a text against a class, each word in the text is considered in sequence to find the longest matches. This avoids ambiguity when there are overlapping potential matches (a classic example being for law review articles, where "Lawyer" abbreviated to "Law.," which then theoretically abbreviated to "L..").
- Duplicative spaces are removed, to deal with situations where a word abbreviates to nothing (i.e., it should not be included).
- Two forms of dot management are performed. First, a space between two single-letter initials, or between a single-letter initial and a number, is removed. Thus, "West Virginia Railroad" would become "W.V.R.R." with no spaces. Second, two dots are collapsed into one, dealing with situations where an abbreviated word is followed by a dot in the original text (e.g., in a URL).

24.2 Cases and Other Names

Case names must be abbreviated in two ways: for inline citations (with only a few words abbreviated), and for non-inline citations. In addition to the standard abbreviation rules, there is a set of words that are unabbreviatable and also disfavored as short names (primarily country names and identifiers of the state in criminal cases).

The case name abbreviation rule is also applied to other names, such as institutional authors and agencies.

The case abbreviation scheme is called name, and the inline case abbreviation scheme is inlcase.

24.3 Legislative Materials

Legislative materials use the standard abbreviation rule, with just a different table. However, "Senate," "House," and "House of Representatives" alone are not abbreviated. The scheme is called leg.

24.4 Court Documents

Court documents use the standard abbreviation rule, with just a different table. The scheme is called cdoc.

24.5 Journals

Journal titles require the most complex scheme for abbreviation, because they do not follow the usual abbreviation rules in several ways. The scheme for these abbreviations is journal.

Initially, some reference types perform a check on journal names to see if they were entered already abbreviated (based on the presence of any dots in the name). If so, then a warning is issued and the journal name is not further abbreviated. (The parameter noabbrevjrn will suppress this.) If not, then abbreviation proceeds.

Next, there is the special case of single-word journal titles or journal titles of "The ⟨*word*⟩." These titles are never abbreviated and the word "The" is retained to prevent titles from becoming incomprehensible (*The Register* would become merely "*Reg.*").

Finally, there is an exception for the deletion of spaces between single-letter initials: The space cannot be removed when the one of the letters is geographic or institutional, and the other letter is for subject matter. For example, "North Carolina Law Journal" would be abbreviated "N.C. L.J.," deleting all spaces except that between "C." and "L." To implement this, the tables effectively use two types of dots, normal periods for geographic/institutional words and a special dot macro for subject matter words, and the dot management algorithm is executed twice.

In terms of abbreviation of words, there are three sources: the table of geographic locations, the table of institution names, and the table of common words in journal titles. These tables implement the differentation between institutional and subject-matter word dots.

24.6 Procedural Phrases

Procedural phrases use a unique set of word-based abbreviations, but there is also a defined list of standard procedural phrases. Additionally, there is a rule about use of commas at the end of phrases. Thus, the abbreviation algorithm must consider a few additional rules. If a phrase to be abbreviated is a predefined one, then the predefined abbreviated phrase is returned, with comma usage established. If it is not, then the phrase is abbreviated, but a warning is issued because the correctness of any trailing comma cannot be established.

The abbreviation scheme is called expl.

The macro `\ExplanatoryPhrase{`⟨*phrase*⟩`}` defines a new explanatory phrase. The unabbreviated phrase should be given, including any trailing comma if needed.

24.7 Publishing Terms

Abbreviations for publishing terms are used in the book publication parenthetical, described in section 34.1. The scheme is called pub.

25 Fonts

Legal citations use a variety of fonts in formatting citations. Complicating matters, there is not consistency in which fonts are used in any given context. Case names are italicized in legal memoranda but not in law review articles; journal titles are set in small caps in some law review articles but in roman in legal memoranda and some journals. Yet despite this variance across dialects of legal citation, there tends to be much consistency within a single dialect: Article titles tend to be formatted identically across a variety of article-like references, as are book authors, signals, and so on.

To deal with this variance, the package introduces a layer of abstraction for fonts. It first defines a set of font contexts: case names, article titles, book authors, and so on. It then introduces a concept of "font packs," namely mappings between the font contexts and the actual fonts to be used. Reference type definition macros do not invoke fonts directly, but rather specify font contexts, which are then translated into actual fonts based on the font packs the user chooses. This way, a variety of font conventions can be supported without the need to revise reference definitions.

25.1 Font Contexts

The following is a table of all font contexts supported.

Name	Description
bookauthor	Author of a book
booktitle	Title of a book
congdoc	Congressional document number
hearing	Hearing title
arttitle	Journal article title
webtitle	Website page title
jrntitle	Journal title
caseproc	Procedural phrase in a case
casefc	Case full name
casesc	Case short name
const	Constitution name
statute	Codified statute
sig	Signal
comment	Signal before commentary
ell	The letter (l) in subsections
dateprefix	The qualifier words before a date
latin	Latin words that are typically italicized

25.2 Fonts

Next, the available font commands that will be linked with font contexts are defined. Standard LaTeX font commands are not used. This is to help with keeping track of which font is being used, to manage capitalization, and to provide a few special features.

The fonts provided are emph, empht, rm, and sc. All have meanings identical to typical LaTeX, except for empht which is like emph except that it does not turn internal emph into roman.

25.3 Font Packs

A font pack associates font contexts with font commands. Each context receives two font commands: one for use in non-inline citations, and one for use with inline texts. The syntax for defining a font pack is as follows:

```
\MakeFontPack{⟨name⟩}{⟨font-specs⟩}
⟨font-specs⟩ := ⟨font-spec⟩ ( , ⟨font-spec⟩ )*
⟨font-spec⟩ := ⟨context⟩ : ⟨font⟩ / ⟨font⟩
⟨font⟩ := emph ‖ empht ‖ rm ‖ sc
```

Example:

```
\MakeFontPack{new}{booktitle: sc/empht, bookauthor: sc/rm}
```

A font pack need not define every font context, since a user can select multiple font packs. In the case of two font packs containing the same font context, the last selected pack overrides earlier ones.

To invoke a font pack, call `\UseFontPack{⟨name⟩}`.

The following font packs are predefined:

- **review**: Law review fonts.
- **memo**: Legal memorandum fonts.
- **italcase**: Full case names are always italicized. Select this after review or memo.

25.4 Shortcut Commands for Fonts

Occasionally, the user needs to select a font manually, for example when defining a hereinafter form that includes a part of a journal article's title. The command `\UseFontFor{⟨context⟩}{⟨text⟩}` selects the font for ⟨*context*⟩ and sets ⟨*text*⟩ in that font.

26 Full, Short, and *Id.* Citations

Except in limited circumstances,[17] this package processes citations linearly starting from the beginning of the document and never looking backwards or forwards. As a result, it must constantly store "state" information about citations previously seen, in order to generate contextually correct citations.

Two types of state information are discussed here: information about where references were first or previously cited, and information about the last citation shown.

If you are looking for commands that change the citation state, for example to suppress *id.* or short citation forms, those commands are in section 14.6.

26.1 Short Forms

For most references, there is a full citation form and a short one. The full form is used first, and the short one used subsequently according to one of two rules: Some references use a *supra* short form that may be used subsequently throughout, while for others the short form may only be used if a full-form citation appeared in close proximity. The purpose of the previous citation state memory, described below, is to determine which of these two forms to use.

Complicating matters, citations can appear in footnotes and in body text, in both law review articles and legal memoranda, and both in inline and non-inline forms. All of these factors interact to affect whether a full or short form citation should be used.

This package follows the following rules to determine long and short forms:

1. An inline short form may be used in text after the full form is used in text, either inline or in a citation.
2. An inline short form may be used in a footnote (a) if an inline short form may be used in text, (b) after a full citation in this or any previous footnote, or (c) after a full inline form in this footnote.
3. A short cite may be used in text after the full citation is used in text.
4. A short cite may be used in a footnote (a) after the full citation has been used in text, (b) after the full citation has been used in a footnote if this is a supra-type short cite, or (c) if the full citation has been used in the last five footnotes.

The command `\resetcitationforms` resets the aforementioned citation state, forcing all citations to be the full form.

The macro `\ShowCitationState{`⟨*ref*⟩`}` gives a textual account of the state of a reference at the point that the macro is called.

26.1(a) First Full Citation

For references that follow the five-footnote rule, formatting sometimes will depend on whether a full citation is the first full citation in the document or a later repeated one. For example, consider a case citation where several cases with the same parties are being cited. Consider the following footnote citations:

[17] These are namely internal cross-references and Table of Authorities generation, described in section 10 and section 20 respectively, both of which use LaTeX's aux file system.

1. Plaintiff v. Defendant (*Plaintiff I*), 123 F. Supp. 456 (S.D.N.Y. 1980).
10. Plaintiff v. Defendant (*Plaintiff II*), 234 F.2d 567 (2d Cir. 1981).
25. *Plaintiff I*, 123 F. Supp. 456 (S.D.N.Y. 1980).
27. *Plaintiff I*, 123 F. Supp. at 460.

The citation in footnote 25, despite being a full citation, ought to use the short-form case name established earlier. It would be odd to repeat the citation in footnote 1 entirely, because that would duplicatively include the short-form parenthetical. But omitting the parenthetical renders footnote 27 ambiguous. As a result, the best presentation for footnote 25 is to use just the established short-form name with the rest of the case fully cited.

This package accounts for this problem by recording whether a reference following the five-footnote rule has been cited previously, enabling citation macros to adjust the contents of full citations accordingly.

26.2 *Id.* Forms

In addition to long and short citation forms, a citation to a reference may be replaced with *id.* or otherwise altered if the immediately preceding citation contains duplicative information. Another set of citation state information must be maintained for this.

Generally, *id.* may be used and citation information may be omitted when it is the same as the previous citation. Assuming that `2 ref at 46` was previously cited, then:

Next input	Produces	Explanation
`2 ref at 46`	*Id.*	Volume and page omitted
`2 ref at 48`	*Id.* at 48	Volume omitted
`3 ref at 50`	3 *id.* at 50	Both numbers included
`3 ref at 46`	3 *id.* at 46	Page cannot be omitted, as volume differs

However, there are two main exceptions to this information omission process. First, not every duplicative element can be omitted, if a “higher level” element differs. In the last example above, the page number 46 is duplicative, but it cannot be omitted because the volume number differs. Second, the presence of multiple citation items in a context can prevent the use of *id.* In this example:

```
\sentence{see ref1 at 5; ref2 at 7}. Further text.
\sentence{see ref2 at 7}.
```

No *id.* may be used in the final citation, even though the reference and page are identical to the immediately preceding one, because the first citation string included two distinct references so *id.* would be ambiguous. However:

```
\sentence{see ref1 at 5; but see ref1 at 7}. Further text.
\sentence{see ref1 at 7}.
```

Here, the last citation should be formatted “*See id.* at 7.” The *id.* is permissible because only `ref1` was cited in the first citation string, but the page number cannot be omitted. Furthermore, a footnote can use *id.* in its first citation only if the previous footnote cited only one reference.

To implement all of this, the package defines several state variables, generally reflecting elements of citation items. Each state variable maintains four pieces of information:

- A current value, set while the citation item is being processed and formatted

- A last value, reflecting the previous citation item's elements to current citation item is allowed to use those previous values

- A flag indicating whether the state variable has received too many values within a citation string such that it cannot be used for element omission in a subsequent string. This "state of the state variable" flag can take on one of three conditions:

  - Unset: The value at the start of the citation string.
  - Set: A citation item has set the value of the state variable, and it may be used in a future citation string.
  - Invalid: The state variable has been set two or more times to different values. It cannot be used in future citation strings, and it invalidates "lower-level" state variables as well.

- An "invalidation list," namely the list of lower-level state variables that become invalid if this state variable is invalid.

26.2.1 List of State Variables

Variable	Invalidates
case	page, orig@page, inline, opt, vol, title
orig@page	
page	orig@page
inline	
opt	
vol	title, page, orig@page, opt
title	vol, page, orig@page, opt

26.2.2 Updating Citation State

The citation state needs to be updated for each citation string, citation item, and footnote. The hook macros defined below are used throughout section 23.

At the beginning of a citation string, all the variables' last values are based on the state of the system prior to the citation, and they all have a condition of Unset.

At the beginning of each citation element, all current values are set to `\relax`. Each element in a citation string can rely on the last values in displaying itself, and can set the current variables as desired. Some current variables may be set prior to the citation-specific macros (e.g., `\@this@case`).

At the end of each element in the citation string, all the current values are checked against their last values.

- If the condition is Unset then the condition becomes Set.

- If the condition is Set and the last value equals the current value, do nothing.

- Otherwise, the condition becomes Invalid, and the condition for any dependent state variables become Invalid.

In all cases, the last value becomes the current value, because within a single citation string the invalidity state is irrelevant.

At the end of a citation string, all Invalid variables have their last value set to `\relax`, and all Set variables retain their last value. (There should be no Unset variables since every variable has its state reviewed for each citation element, and there must be at least one citation element.)

At the end of a footnote, the last values of the state variables are either globalized or relaxed, depending on whether there was one or more references cited.

For inline citations, special handling of citation state is required as described in section 14. Thus, several macros are provided for saving all the state variables to a macro so that they can be restored or reset later.

26.3 Footnote State Management

Footnotes interact closely with both short and *id.* citations: Whether a short citation can be used depends in part on whether the full citation was in text or a footnote, and *id.* citations can be used at the start of a footnote depending on the content of the previous footnote. How footnotes interact with citation forms also depends in part on whether a document is a law review or legal memorandum document.

As a result, several accounting tasks must be performed at the beginning and ending of footnotes. For all types of articles, a flag is set indicating that text is in a footnote, and at the end of footnotes any deferred note text that has not been emitted (as described in section 19) is put at the end of the note.

For law review articles, a counter is kept of how many unique citations have appeared in the footnote. Each reference cited advances the counter by one, except *id.* citations advance the counter only from zero to one. The *id.* system above uses this counter to determine state after the footnote. Legal memoranda, on the other hand, prohibit *id.* cites at the beginnings of footnotes, to avoid ambiguity as to whether *id.* refers to the last footnote or the last in-text citation.

27 *Supra* Short Forms

Some citations use a short form that reads "⟨*desc*⟩, *supra* note ⟨*num*⟩." Additionally, a writer may specify the `hereinafter` reference parameter for a reference, which applies a *supra* short form to that reference. This section describes how such citations are produced and managed.

A reference type that uses a *supra* citation should execute:

```
\hi@supra@form{⟨ref⟩}{⟨desc⟩}
```

in constructing citation macros. The argument ⟨*ref*⟩ is the reference name, and ⟨*desc*⟩ is the description to precede the word "*supra*." The description is usually the reference author's last name, and it may use font shortcut commands as described in section 25.

Setting up a *supra* form requires performing the following tasks:

- Registering the description as a short citation name for duplicate checking, as described in section 22.

- Creating both a short citation form and an inline form, where the inline form displays the description text.

- Records that ⟨*ref*⟩ is a *supra*-type reference that does not follow the five-footnote rule, to conform with the expectations of short citation state described in section 26.

28 Miscellaneous Formatting

This section describes various formatting utility macros that are used when defining citation formats.

28.1 Ordinal Numbers

In several situations it is useful to convert numbers to text. Several macros do this, and can add ordinal number suffixes as well.

28.2 Trailing Dot Management

When the text of a formatted citation item ends with a dot but the citation string is followed by a period, the extraneous dot needs to be removed. To do so, this package provides an infrastructure for tracking whether a citation item ends with a dot. Reference definer macros must use this infrastructure to indicate when the last item they emit to the document is a dot.

28.3 Capitalization

At the beginnings of sentence citations, the first letter needs to be capitalized.[18] The first letter could be part of a signal or text within the citation text itself, and the citation text might contain prefatory font or other macros. Thus, this package provides an infrastructure for citation macros to capitalize the first letter of displayed text when necessary, but not to further capitalize subsequent text.

The font macros in section 25 use the capitalization system extensively.

28.4 "The"

Statutes and other reference names may start with the word "The." When those are used inline as adjectives, the word "The" generally ought to be removed. This is done by replacing the leading definite article with a special macro that displays as text in non-inline citations but may be suppressed in adjective inline citations.

28.5 Other Macros

A variety of other macros deal with using citation parameters, differentiating between citation contexts (inline, nameless, optional arguments), and spacing.

[18] Capitalization also occurs inside sentence citations between signal classes, as explained in section 16.

Part VI
Reference Types

This Part is a comprehensive list of every reference type that *Hereinafter* supports. Reference types are organized topically into chapters which contain both introductory material describing overall characteristics of reference types and then specific definitions. The chapters contain three types of subheadings indicating the material contained:

- Numbered headings in serif type are general descriptions of reference type features.

- Unnumbered headings in **sans serif type** describe specific reference types, which may be invoked using `\def`⟨*reference type*⟩ commands as described in section 6.2.

- Headings in ***italic sans serif type*** describe reference types that are aliases for other types.

29 What If the Type I Need Isn't Listed?

The available reference types reflect the package author's own citation needs over the years. Having used this package many times to write a diverse range of works involving technical, historical, judicial, political, and media references, he has covered nearly every type of conventionally cited work. Nevertheless, in the event that a writer needs to cite a reference type not specifically defined here, the package makes several affordances.

First, several reference types are designed to be catch-all types. For government documents, govdoc provides a generic format for references cited in the *Federal Register* and other reporters, and for miscellaneous agency promulgations that are essentially unpublished. For non-governmental works published in some periodical format, website works as a generic format, especially since it accepts any publication parameters that can be attached to a book (editor, publisher, etc.; see section 34.1). Works that are part of a published book or volume can generally be defined using the highly flexible citecontainer type.

If truly nothing works, then the reference verbatim can be used to inject arbitrary text into a citation. Even when that is used, the writer still enjoys most of the structural capabilities of this package: signals management, string citations, and so on.

Finally, writers faced with a class of unexpected reference types are encouraged to develop new citation macros. The syntax and programming conventions for reference type definer macros are not simple, so it may be easiest to make small tweaks to the existing reference definitions. In doing so, the package author recommends using more specific, uncommon reference types as a starting point, such as those in section 37. The widely used reference types such as case and book require a great deal of code to deal with many different inputs and the idiosyncractic historical rules of legal citation for those types, and would be difficult for would-be package improvers to understand at first. Reference types such as procart, designed to cover a narrower class of references and that include fewer features, would be easier to revise to one's tastes.

30 Cases

30.1 Parties and Case Titles

There are three ways a case can be named, which dictate the parameters that must be given for a case reference definition:

- With adversarial parties, such as *Brown v. Board of Education*. The parameters p and d set each party name, and the parties parameter can be used to set both at once.

- With a single name, such as *In re Alappat*. The name parameter accepts this type of case name.

- With both adversarial parties and a common name. This occurs with older, well-known Supreme Court cases, such as *Consolidated Electric Light Co. v. McKeesport Light Co.* (The Incandescent Lamp Patent Case), and with modern class-action litigation. All of the aforementioned parameters should be used for these cases.

These input parameters are used to produce five formatted names for the case:

- The full title, used in full inline citations. This is abbreviated according to the inlcase scheme.

- The abbreviated full title, used in full non-inline citations. This is abbreviated according to the name scheme.

- The abbreviated full title to be used in the Table of Authorities. This differs in spacing (an optional line break is placed before the versus) and in text order (phrases like *In re* are placed at the end rather than the beginning). It is abbreviated according to the inlcase scheme.

- The short form for inline citations, abbreviated with the inlcase scheme.

- The abbreviated short form for non-inline citations, using the name scheme.

30.1(a) Party Name Syntax Party names for p and d have the form:

⟨*party*⟩ := ⟨*text*⟩ [`ex rel.` ⟨*text*⟩]

Examples:

```
Doe
State ex rel. Adams
```

The "base" name and the name after "ex rel." are treated as two separate names, each a candidate for being the case's short name.

For common case names with the name parameter, there are more possibilities because prefixes *In re* and *Ex parte* can be used:

⟨*name*⟩ := [`In re` ‖ `Ex parte`] ⟨*text*⟩ [`ex rel.` ⟨*text*⟩]

Examples:

```
In re Comiskey
Ex parte New York ex rel. Jones
```

The latter example is procedurally implausible but nevertheless syntactically acceptable.

30.1(b) Full Name Formatting There are three full names required, for inline, non-inline, and Table of Authorities citations. How each appears depends on which parameters are given:

- If party names are given alone, then the full names are "⟨*p*⟩ v. ⟨*d*⟩" appropriately abbreviated.
- If party names and a common name are given, then the full names are "⟨*p*⟩ v. ⟨*d*⟩ (⟨*name*⟩)" appropriately abbreviated.
- If only the name is given, this is the most complex case. If the name begins with "The," then the inline form uses `\hi@inline@the`, and the word "The" is placed at the end of the Table of Authorities form. If the name begins with a prefix like *In re*, then the prefix is placed at the end of the Table of Authorities form. Otherwise, the name is assembled with "The" and the prefix attached, appropriately abbreviated.

Additionally, the fonts need to be set up in assembling the full name forms. This is because the name can use multiple fonts internally, for different parts.

30.1(c) Selecting the Short Name In theory, there are up to six possible candidates for the short name of a case citation: the two parties, the common name, and the *ex rel.* components of each of those. To decide which of them is the best, this package employs a scoring system applied to each candidate name. The following factors are considered:

- Names that are impermissible party names receive the lowest scores; they will be used only if there is no other option.
- Names with capitalized first words or the word "of" are less likely to be good short names so they receive lower scores.
- *Ex rel.* names are disfavored over base party names to the extent that none of the above factors apply.
- Common names are preferred over topside parties, which are preferred over bottomside parties, to the extent that none of the above factors apply.

30.1(d) Formatting the Short Name Short names may be user-provided or automatically determined based on the selection algorithm above. To format the short name, several tasks must be performed:

- For user-given short names, the word "the" at the beginning needs to be processed for `\hi@inline@the`.
- For automatically determined short names, some conventional words are removed from the name as described below.
- The short name must have appropriate fonts set.
- If the short name is not obviously connected to the full name, then the full name inline name requires a parenthetical, and the full citation name (a) requires a parenthetical the first time it is ever used and (2) should actually use the short name on subsequent uses, as described in section 26.1(a).

Regarding the removal of conventional words, these are typically business or geographical designations that are uninformative for the short name. A list of ignored words follows:

,	Inc.	Co.	Corp.
LP	LLP	LLLP	P'ship
Partnership	LLC	Ltd.	America
Am.	USA	International	Int'l
of			

30.2 Nominative Reporters

Generally, reporter names are not abbreviated or changed from what is given as input to the parameter rep. This is because reporters are conventionally known by their abbreviations, and there is no consistency in reporter abbreviations. However, some common reporters have both an official series name and nominative reporter names. Volume 21 of the *United States Reports*, for example, is also volume 8 of Wheaton's reports, and typically both are cited:

```
21 U.S. (8 Wheat.) 1
```

In most cases, the pagination of the series reporter and the nominative reporter is the same, and adding the nominative reporter is a simple matter of looking up the corresponding name in a table. As a result, this package handles these nominative reports automatically. For nominative reporters with different pagination, use the parallel parameter.

30.3 Date Parenthetical

Ordinarily the date parenthetical for cases contains the court and year parameters. A complication arises when the case is being cited as part of a casedoc reference. In that case two things need to happen:

- If this case has a reporter listed (rep is set), then a parenthetical for the docket number must be added.
- If the case document has its own date set, then that date overrides the date given for the case.[19]

30.4 Procedural History

Procedural history for a case is given in the prior and subsequent parameters, which are used to construct parentheticals. The syntax for those parameters is:

⟨*history*⟩ := ⟨*phrase*⟩ **:** ⟨*ref*⟩

Example:

```
reversed: prior-case
```

[19] This does not conform to some legal citation practices, which attach the date of the case decision rather than the date of the document. That practice seems questionable since the date of interest to the reader is more likely the case document's date. Nevertheless, it is simple enough to assign the case decision's date to the case document when defining the case document reference.

The ⟨*phrase*⟩ is an explanatory phrase that will be abbreviated and also checked against a table of valid explanatory phrases. If a desired phrase is not already listed, the macro `\ExplanatoryPhrase{`⟨*phrase[,]*⟩`}` is used. Note that the comma at the end of the argument is significant, as some explanatory phrases use a comma and others do not.

The main task to be accomplished in producing case history is reconciliation of names. If the case being defined and the case in its history parenthetical have the same-named parties, then the name can be omitted from the case in the history parenthetical. If they have different names, then the phrase "sub nom." must be added to explanatory phrases that end with a comma.

case: A case citation

Key Parameters:

p, **d**: Parties named in the case.
name: A common or single name for the case. Required unless parties are given.
vol, **rep**, **page**, **cite**: Citation to the reporter containing the case.
docket: The docket number of the case. Required if cite parameters are not given, or may optionally be given if casedoc references will use this case.
court: The court deciding the case, which may be omitted if the reporter uniquely identifies the court.
year: The date of decision. For citing cases not yet decided, a qualifier word should be included to explain the procedural posture.
inline: A user-defined short name for this case.

Optional Parameters:

inlinedefendant: Don't use the first party as the short name.
dbid: A database identifier for unreported cases. This is optional since the docket number and court are sufficient to identify an unreported case.
slip: Whether this is a slip opinion; this adds a note to the page numbers.
prior, **subsequent**: Procedural history of the case, see section 30.4.
enbanc, **mem**, **percuriam**: Parentheticals denoting the nature of the opinion.

Example:

```
\defcase{goodyear-dunlop}{
    parties={Goodyear Dunlop Tires Operations, SA v. Brown},
    cite=564 U.S. 915,
    year=2011,
}
```

Produces:

Goodyear Dunlop Tires Operations, SA v. Brown, 564 U.S. 915 (2011).

In this example, note the use of the parties and cite parameters, as shortcuts to entering p, d, vol, rep, and page. Also observe that the party names are entered unabbreviated, but reporters and court names are entered abbreviated (because those are never used unabbreviated, and the abbreviations are too inconsistent).

Example:

```
\defcase{nimitz}{
    name=In re Nimitz Technologies LLC,
    court=Fed. Cir.,
    percuriam,
    date=dec 8 2022,
    docket=2023-103,
}
```

Produces:

In re Nimitz Techs. LLC, No. 2023-103 (Fed. Cir. Dec. 8, 2022).

This demonstrates a case with a single name rather than adversarial parties, and with a docket number rather than a reporter citation.

admincase: An administrative decision

This is the same as a case with the default category set to `admin`. Parameters are identical to case with the following addition:

Key Parameters:

agency: Same as court but abbreviates the name.

Example:

```
\ExplanatoryPhrase{affirmed without opinion,}
\defadmincase{biodelivery}{
    parties={BioDelivery Sciences International, Inc. v. RB
        Pharmaceuticals Ltd.},
    docket=No. IPR2014-00325,
    court=P.T.A.B.,
    date=jun 30 2015,
    slip,
    paren=final written decision,
    subsequent=affirmed without opinion: biodelivery-cafc
}
```

Produces:

BioDelivery Scis. Int'l, Inc. v. RB Pharms. Ltd., No. IPR2014-00325 (P.T.A.B. June 30, 2015) (final written decision), *aff'd without opinion*, ...

This shows an administrative decision citation with subsequent history. The reference `biodelivery-cafc` would be defined elsewhere.

casedoc: Document used in a case

Key Parameters:

name: The name of the document.
citation: The reference name of the case in which this document was filed.
docket: The docket number of the case, if it was not already given in the case.
year: The date of the document itself.
number: The document number or docket list entry. This is added as a parenthetical, preceded by "Doc. No."

Example:

```
\defcasedoc{wyze}{
    citation=case: {
        parties={Wyze Labs, Inc. v. Beijing Rockrobo
            Technology Co.},
        docket=2:21-cv-941,
        court=W.D. Wash.,
        date=dismissed jan 3 2022,
    },
    name=Complaint for Declaratory Judgment,
    date=july 15 2021,
}
```

Produces:

Complaint for Declaratory Judgment, *Wyze Labs, Inc. v. Beijing Rockrobo Tech. Co.*, No. 2:21-cv-941 (W.D. Wash. July 15, 2021).

Here, the underlying case is defined using an anonymous reference definition (see section 13).

courtdoc: A document in the docket of current litigation

It is intended for use in a legal brief or filing, to cite to other documents in the docket of the same case in which the document is being filed. For example, a brief in opposition to a motion may use this reference type to cite arguments in the motion papers. To cite briefs or documents filed in another case, use casedoc.

This package does not use parentheses around citations to court documents. This is because it makes the citation form incompatible with other citations (e.g., if one cites a court document and a statute in one string cite). Furthermore, most court documents filed today do not appear to use parentheses, and courts do not appear to require them. If they are desired, commands such as `\sentence` or `\clause` can be surrounded with parentheses.

In most appellate work and particularly before the Supreme Court, this reference type is rarely used. The more common practice now is to name the document to be cited in text and then include "(at ⟨*page*⟩)" within the sentence:

> Petitioner's brief argues (at 8) that the sky is red, but that is wrong.

There is not really a need for automation of these kinds of citations.

Key Parameters:

name, **inline**: Full and short names for the document. They will be abbreviated according to the cdoc scheme.

number: The document number of the document being cited, if one is present. The phrase "Doc. No." will be prepended to it, and the number will be made a parenthetical to the citation. If inline is not given, then the document number will be used as the short name. (Otherwise the short form will be the same as the full name.)

year: The date the document was filed.

31 U.S. Constitutions and Founding Documents

These references cover the U.S. Constitution, state constitutions, and other founding documents such as the *Federalist* papers and the Declaration of Independence.

const: Citation to a constitution

Because of the intensive way in which parts of constitutions are analyzed, this package provides that each *part* of a constitution to be cited should be defined as a separate reference.

Key Parameters:

name: The textual name of the constitution part being cited. This can be a textual representation of the pin cite (e.g., “Article II”) or a common name (e.g., “the Equal Protection Clause”).

page: The pin cite specification for the constitution part for this reference. Typically this will be a listing of named divisions: `article 1, S 8, clause 8`, for example.

Optional Parameters:

place, **state**: The state from which this constitution originates, if not “U.S.” The place will be abbreviated using the name scheme.

year: The year of adoption, for superseded constitutions. To enter years for adoption of specific amendments or provisions, use paren.

Pin cites given in citation items will use the pin cite joining algorithm given in section 18.

Example:

```
\defconst{a2s3}{
    name=the Take Care Clause,
    page={Article II, S 3}
}
```

constamend: A Constitutional amendment

This is just a shortcut for const when defining whole amendments. The parameters and usage are identical to const except the name and page parameters are automatically set to “the ⟨*ordinal-number*⟩ Amendment
and “amendment ⟨*number*⟩” respectively.

Key Parameters:

number: The number of the amendment. This should be a regular Arabic number; it will be converted to Roman numerals for the citation form and ordinal number text for the inline form.

Example:

```
\defconstamend{1a}{
    number=1,
}
```

founding: A national founding document

This generalizes the form for citing the Declaration of Independence, for which the reference type is otherwise unclear.[20] Mainly, this form provides a flexible citation container that sets the given title in the same font as that for constitutions.

Key Parameters:

name: The name of the document.

Optional Parameters:

year: The relevant year of the document.

Pin cites given in citation items will use the pin cite joining algorithm given in section 18.

federalist: A Federalist paper

Key Parameters:

number: The number of the paper.

Optional Parameters:

author: The author of the paper.

[20] The *Bluebook* does not define a type for the Declaration of Independence but gives it as an example of how to cite subdivisions. Nevertheless, the chosen format is puzzling. It puts “The” in front of the name, even though “The” is typically omitted from titles of statutes. (“The” is not omitted from book titles that actually start with “The,” but the Declaration does not have an official title on its face.)

It then puts “U.S.” in the date parenthetical. Putting aside the question of whether the United States existed at the time the Declaration was signed, it is unclear why the country is included in the parenthetical rather than as part of the title. Typically, that placement of a jurisdiction would identify a court or similar tribunal issuing a decision, but it is a stretch to say that the Declaration is an adjudicated result. If it is to identify the country issuing the Declaration to distinguish from other nations’ declarations of independence, then “U.S.” ought to be added in a separate parenthetical based on the usual rule for foreign materials. The best interpretation perhaps is that the Declaration is being cited as a book, explaining the superfluous “The,” and that the United States is being identified as the publisher of a pre-1900 work—a strange approach for many reasons.

32 Statutes

32.1 Short Statute Names

Many types of references will use the following convention for statute names. The following inputs are relevant, with syntax as follows:

Key Parameters:

name: The full, unabbreviated name of the statute or document. The word “The” should not be included at the beginning; it is assumed that the full name would be recited with “The” in ordinary speech.

inline: The short, abbreviated name of the statute or document. Because some abbreviated names include “the” at the beginning and others do not (compare “the FFDCA” with “HIPAA”), it is necessary to specify the form. For references that use “the” before the short name, include it in this parameter.

For both the full and short forms, the package will automatically prepend “the” or “The” at the beginning of the name depending on citation context, particularly when `\Inline` or `\adjective` are used (see section 14).

If no short name is given, then one is automatically constructed that is identical to the full name, except any year specification will be removed (e.g., “Communications Act of 1930” would be shortened to “Communications Act”). The word “the” is implicitly added to the start of the constructed short name as well.

statcode: A statute in a code

Key Parameters:

vol, **rep**: The volume number and title of the statutory code.

page: The section or other subdivision number of the statute in the code.

Optional Parameters:

name, **inline**: The full and short names for this reference, as described with regard to statute names in section 32.1.

origsect: If the statute has a section number different from its codification section number, that original section number may be included here.

year: The year of publication of the code containing the statute.

Any book publication parameters described in section 34.1 are also permitted.

Pin cites given in citation items will use the pin cite joining algorithm given in section 18.

Example:

```
\defstatcode{15-usc-1}{
    cite=15 U.S.C. S 1,
}
```

Produces:

15 U.S.C. § 1.

The `\defcitegroupcommand` is especially useful in combination with `\defstatcode`; see section 6.3 for examples.

Example:

```
\defstatcode{337}{
    name=Tariff Act of 1930,
    origsect=S 337,
    cite=19 U.S.C. S 1337,
    paren=as amended,
}
```

Produces:

Tariff Act of 1930 § 337, 19 U.S.C. § 1337 (as amended).

This demonstrates a statute that has been codified with a section number different from its session law section, where the session law section number is still commonly used.

regcode: A regulation in a code

This only differs from statcode in the toacat being `regulation` rather than `statute`. The parameters and features are the same as those for statcode.

statsess: A statute in session laws

Key Parameters:

type: The name to be prefixed to the statute's serial number. By default this is "Pub. L. No."

number: The serial number of the statute, which should conform to the value expected for type.

vol, **rep**, **page**, **cite**: Citation locator information for the statute in a session law compilation such as the *Statutes at Large*. This will be used unaltered and should be entered abbreviated.

year: The date of enactment of the statute.

Optional Parameters:

name, **inline**: The full and short names for this reference, as described with regard to statute names in section 32.1. If no name is given, one will be constructed out of the date (which should be a full date as a result).

chapter: This sets type to "ch." and then sets number to the parameter value.

in: For session laws contained within another larger session law (e.g., a part of an omnibus budget act), it may be necessary to cite both session laws to identify the contained one. This parameter can be given the reference name of the larger session law to effect this citation form.

slip: For slip laws not yet included in a compilation, use this flag.

place: The place of publication (e.g., the state).

publisher: The publisher of the statute, if relevant.

Example:

```
\defstatsess{nttaa}{
    name=National Technology Transfer and
    Advancement Act of 1995,
    inline=NTTAA,
    publiclaw=104-113,
    cite=110 Stat. 775
}
\sentence{nttaa at S 12/d1 ::stat: 783}.
```

Produces:

National Technology Transfer and Advancement Act of 1995 (NTTAA), Pub. L. No. 104-113, § 12(d)(1), 110 Stat. 775, 783.

Note how a pin cite to a session law must include not only the section number, but also the Statutes at Large page number following a `::stat:` segment marker. See section 17.1(a) for more on segments in pin cites.

stattitle: A titled statutory citation

This is for statutes that, having been amended many times, are treated like codifications based on their original section numbers. The Communications Act of 1930, the Public Health Service Act, and the Smoot–Hawley Tariff Act are examples of statutes with well-known section numbers that do not correspond to their codifications; a parallel cite to the U.S. Code is required and provided in a struct that maps original section numbers to U.S.C. citations.

This reference type should supersede statcode with origsect.

The scope of the reference item here is the whole statute, not just an individual section therein. To cite to a section of the statute, the section number is given as the pin cite information. The reference aliasing system described for alias can assist in simplifying references to these statutes if a particular section number is repeatedly used.

Key Parameters:

name, **inline**: The full and short names for this reference, as described with regard to statute names in section 32.1. Here the name is mandatory, since section numbers correspond to the statute number with that name.

vol, **rep**, **page**, **cite**: The citation locator information for the statute in a codification. Here, page should refer to the range of section numbers in the codification that the statute covers. Note that the range will likely never be displayed, if all citations to the statute are to particular sections.

struct: Following the format described in section 12, this should be a map of section numbers of the statute to vol, rep, and page values in the codification. The structure is key to ensuring that pin cites display correctly.

Any book publication parameters described in section 34.1 are also permitted.

rule: A rule of evidence or procedure

Citations to judicial rules are not especially well-defined. Probably the better option is to characterize the name of the rule set as a book title and each rule as a named division therein. That would produce output along the following lines:

```
\defrule{frcp12}{rep=Fed. R. Civ. P., page=rule 12}
\sentence{frcp12 at /b6}.
```

Produces:

Fed. R. Civ. P. R. 12(b)(6).

The oddity of this form is that "R." appears twice. To avoid that, it is not enough to remove "rule" from the pin cite (`page=12`), because then the number looks like a page number that will not accept subdivisions like "(b)(6)." The solution is to use the dash character before the pin cite as described in section 17:

```
\defrule{frcp12}{rep=Fed. R. Civ. P., page=-12}
\sentence{frcp12 at /b6}.
```

Produces:

Fed. R. Civ. P. 12(b)(6).

The dash forces the pin cite to be interpreted as a "named" division with no name.

The other option is to define a specific rule as a reference. With this approach, *id.* citations would not be used when two different rules are cited, even if they are from the same rule book.

Key Parameters:

rep: The abbreviated name of the rule set being cited.

page: The rule number, see above.

Optional Parameters:

name: A name of the rule, if any.

court: The court issuing the rule, if not apparent from rep.

year: The year of promulgation of the rule, if needed.

Pin cites given in citation items will use the pin cite joining algorithm given in section 18.

modelcode: A model code, restatement, or other standard

A model code is cited as something of a hybrid between a statute and a book, the main feature of note being that the "author" of the model code (typically the sponsoring institution) is styled like an author of a book, but is placed in the date parenthetical like a court.

Key Parameters:

name: The name of the model code, to be abbreviated like a case name.

instauth: The sponsoring organization of the model code.

Optional Parameters:

edition: If the document being cited is a named draft ("Tentative Draft" or such), give the draft name here.

33 Government Works

These references are for legislative and executive materials that do not fall into the above categories.

bill: An bill or resolution of Congress

Key Parameters:

number: The serial number of the bill. This is used as-is, and must include any prefix (e.g., “S. 1234”).

congress: The term number of the Congress in which the bill was introduced. This should just be a number, and an ordinal suffix will be added automatically.

year: The date of the bill. By default, the date should be the date of introduction. Otherwise, use a date qualifier or the status parameter as described below.

Optional Parameters:

name, **inline**: The full and short names for this reference, as described with regard to statute names in section 32.1.

status: The status of the version of the bill being cited, for example “introduced,” “reported by ⟨*committee*⟩,” or “passed by Senate.” This will be added before the date, so no qualifier should be added before the date if this parameter is used. Text after “by” will be abbreviated according to the leg abbreviation scheme.

Example:

```
\defbill{yoda-2015}{
    bill=H.R. 862,
    congress=114,
    name=You Own Devices Act,
    date=feb 11 2015
}
```

Produces:

You Own Devices Act, H.R. 862, 114th Cong. (Feb. 11, 2015).

congdoc: A Congressional numbered document

This is for reports, and documents. Unnumbered documents should use congprint.

Key Parameters:

number: The document number. This should include the abbreviated description for the number (e.g., “H.R. Rep. No.”). The description must match a predefined list of permitted descriptions, and those descriptions will be de-abbreviated to make inline citation forms.

year: The date of the report.

Optional Parameters:

author, **name**: The author(s) and title of the document.

page: A pin cite that will be included in the full citation. This might be used, for example, for multipart reports (i.e., `page` could be `part 2`).

Example:

```
\defcongdoc{aia-report}{
    name=America Invents Act,
    serial=H.R. Rep. No. 112-98,
    year=2011,
}
```

Produces:

America Invents Act, H.R. Rep. No. 112-98 (2011).

congprint: A Congressional unnumbered document

This is essentially a more structured citation to a book.

Key Parameters:

- **author**: Author(s) of the report or document.
- **committee**: If no author is given but this parameter is, then the author becomes "Staff of ⟨*committee*⟩." The committee name is abbreviated.
- **name**: The title of the document.
- **congress**: The term number of the Congress in which the bill was introduced. This should just be a number, and an ordinal suffix will be added automatically.
- **year**: The year of publication.

congrec: A Congressional Record debate

This supports both the daily edition and the permanent edition, determined based on the page number. Note also the cr reference below.

Key Parameters:

- **vol**: The volume number.
- **rep**: The name of the record, typically "Cong. Rec."
- **page**: The page number where the relevant debate begins. For the daily edition, include the letter prefix in the page number (preceded with "!" to ensure it is not treated as a division name, see section 17.2). The phrase "daily ed." will automatically be added to the date parenthetical in this case.
- **year**: The date of the debate or of publication of the volume. It typically is a full date for the daily edition, and a year for the permanent edition.

Any book publication parameters described in section 34.1 are also permitted. This is primarily useful for volume 1 of the *Annals of Congress*.

Example:

```
\defcongrec{kastenmeier-statement}{
cite=126 Cong. Rec. 29895,
year=1980,
paren=statement of Mr.\ Kastenmeier
}
```

`cr`: The Congressional Record

As an alternative to congrec above, the predefined reference `cr` can be used to cite arbitrary pages in the *Congressional Record* permanent edition. The date should be given as the optional argument:

```
\sentence{see 123 cr[1977] 17147}.
```

Produces:

See 123 Cong. Rec. 17147 (1977).

hearing: A hearing in Congress

These should be hearings where the Goverment Printing Office has printed a transcript of the proceedings. Use testimony to cite separately published testimony in a hearing.

Key Parameters:

committee: The committee before which the hearing was held.

congress: The term number of the Congress in which the bill was introduced. This should just be a number, and an ordinal suffix will be added automatically.

Optional Parameters:

type: The nature of the proceeding, default "Hearing."

name: A title to the hearing, if given.

number: A bill number (same as the bill reference type), to be given if the hearing deals with a specific bill.

Example:

```
\defhearing{azar-hearing}{
    name=Nomination of Alex Azar to Serve as Secretary
        of Health and Human Services,
    committee={Senate Committee on Health, Education,
        Labor, and Pensions},
    congress=115,
    date=nov 29 2017,
}
```

prespaper: A Presidential Paper

Key Parameters:

name: The title of the paper.

vol, **rep**, **page**, **cite**: Citation information for the paper. Typically rep is "Pub. Papers" or "Weekly Comp. Pres. Doc."

year: The date of the document.

dcpd: Daily Compilation of Presidential Documents

Because documents in the DCPD are numbered rather than consecutively paginated, a different reference type is required.

Key Parameters:

name: The name of the document.

number: The number of the document.

year: The date of the document.

patent: A U.S. patent

Key Parameters:

number: The patent number. It should be entered as an unpunctuated raw number, preceded by "D" for design patents. Commas will be added to the number, and the last three digits will be used for the short form.

year: The date of the patent. The date *must* contain a qualifier, likely "issued" or "filed."

issueyear: Equivalent to setting year to ISSUED ⟨*issueyear*⟩.

Optional Parameters:

name: The title of the patent, if relevant.

govdoc: A government document

This is a generic reference type amenable primarily to two types of government documents. First, it can be used for *Federal Register* notices and other government documents published in agency reporters or compilations. In this case, the abbreviated reporter and such information should be included as parameters. In most cases, this produces output not unlike admincase, the main difference that the document title has no special font applied to it (admincase would apply whatever font is applied to case names).

Second, this reference type can be used without the reporter parameter, to cite short government papers or promulgations. The traditional route is to cite such documents as book references, but for one-page fact sheets or brief policy statements this gives too much apparent authoritative weight. However, the agency or institution name is placed in the date parenthetical, consistent with other government or court documents, thereby emphasizing the governmental origin of the document.

Key Parameters:

name: The title of the document.

author, **instauth**, **agency**, **court**: The name of the agency or institution issuing the document. It should be abbreviated according to the name scheme. If instauth or agency are used, this will be done automatically.

vol, **rep**, **page**, **cite**: The citation locator information for the document, if the document was published in a reporter. The reporter should be abbreviated when entered.

docket: If citation locator information is not entered, a docket number may be given to aid in identifying the document.

year: The date of the reference.

Optional Parameters:

prior, **subsequent**: Procedural history associated with the document, as described in section 30.4.

number: A serial number for the document, placed before the document title.

Example:

```
\defgovdoc{des}{
    name=Encryption Algorithm for Computer Data Protection,
    cite=40 Fed. Reg. 12134,
    date=mar 17 1975,
}
```

Produces:

Encryption Algorithm for Computer Data Protection, 40 Fed. Reg. 12134 (Mar. 17, 1975).

fedreg

Alias for govdoc.

testimony: Testimony in a congressional hearing

Many congressional hearings are not published today, given the availability of video recordings. As a result, to cite a witness's testimony at a hearing, it is often necessary to use the witness's own copy, usually made available online. There is no well-defined way to cite this testimony. One option, commonly used, is to cite the privately published testimony as a hearing reference. This is plainly wrong and confusing, especially if multiple witnesses' testimonies are to be cited. Another way would be as a book reference, but that obscures the testimony's legislative involvement. It is also not clear what the title should be or where the congressional subcommittee should be identified when taking this approach.

This package defines a new reference type for these privately published testimony documents. It uses a citecontainer-like approach, with the hearing in which the testimony was delivered being defined separately as a reference to be used as part of the testimony definition. This has the benefit that, when two different witnesses' testimonies are cited, the hearing is cited with the short form for the second witness cited.

Key Parameters:

citation, **in**: The hearing reference name or anonymous reference definition.
author: The witness testifying.

Optional Parameters:

type: The nature of the testimony, by default "Testimony."

comments: Comments on an agency proceeding

This is a mildly deprecated reference type for comments filed in an agency proceeding. (It remains here because it is used in the package author's CV.) The better option is actually to use casedoc, defining the agency proceeding in which the comments were filed as a separate reference (good choices would be admincase and govdoc). In that case, the title of the document would be given as "Comments of ⟨*party*⟩."

Key Parameters:

commenter: Name of the entity filing the comments.
name: The title of the proceeding in which the comments were filed.
vol, **rep**, **page**, **cite**: Identifier information for the proceeding (typically a *Federal Register* citation if the comments were filed pursuant to such a notice).

year: The date of the comments.

Optional Parameters:

author: The author of the comment, usually the signatory attorney. This probably should not be included in most cases, in the same way that the signing attorney is not identified for legal briefs in casedoc.

secfiling: An SEC filing

Key Parameters:

instauth: The company filing the report.
name: The name of the report, typically along the lines of "Annual Report."
number: The number of the form filed (e.g., "10-K").
year: The date of filing.

34 Books and Containers

This section describes two reference types for larger book-like works, and also a common parenthetical used across many citation types.

34.1 Publication Parentheticals

Many citation types accept a common parenthetical that contains a date of publication and other publication information, such as editions, editors, and publishers. This standard parenthetical accepts the following parameters, all of which are optional (although some reference types will require year):

Optional Parameters:

editor: Names of one or more editors of the book.

edtype: The abbreviation to follow the editor names, by default “ed.” or “eds.”

number: A serial number for the book. This parameter is used only if publisher is also given. (For book references, a serial number with no publisher will be used differently.)

edition: The edition number of the book. An ordinal suffix will be added.

publisher: The name of the publisher.

forthcoming: If set, the word “forthcoming” will be added to the parenthetical.

year: The year of publication.

book: A book or non-periodic material

Key Parameters:

author, **name**: The author(s) and title of the work.

vol: A specific volume number being cited. Preferably, though, the volume number would be included as part of the citation item information.

number: A serial number for the book. If the serial number is attached to the work’s author (i.e., it’s an institutional series), then enter the serial number alone, protecting any commas with braces. If the serial number is attached to a publisher’s series, then use a comma as described with regard to the publication parenthetical.

Optional Parameters:

struct: If volumes of the work have different authors, titles, or other information, provide the differing values in a struct as described in section 12.

Any book publication parameters described in section 34.1 are also permitted.

Example:

```
\defbook{nimmer}{
    author=Melville B. Nimmer,
    author=David Nimmer,
    title=Nimmer on Copyright,
    year=2014,
    hereinafter=\UseFontFor{booktitle}{Nimmer},
}
```

Produces:

Melville B. Nimmer & David Nimmer, Nimmer on Copyright (2014) [hereinafter Nimmer].

citecontainer: A citation in another citation

A cite container is a highly flexible vehicle for citing works that are contained inside other larger works: chapters in edited volumes, introductions to books, letters reprinted in compilations, and documents in appendices to judicial opinions. The defining feature of the cite container is that the pagination of the contained work follows the pagination of the container, such that any pin cite needs to be attached to the container's locator information.

There are two flavors of cite containers. First, the contained item may be a standalone reference with a type, such as a letter or case. If so, then the contained item is defined as a reference and passed to the citation container's citation parameter.[21]

Second, the contained item may not require a standalone definition, as would be the case for a book chapter. In that case, the cite container takes parameters author, name, and year to specify the contained work's information, instead of the citation parameter.

Key Parameters:

citation: The reference name or anonymous reference definition for the contained work.

author, **name**, **year**: Information for the contained work, used in lieu of citation.

in: The reference name or anonymous reference definition for the container work.

vol, **page**: The pin cite information for the contained work inside the container (e.g., for a chapter in an edited volume, the volume and page number where the chapter begins).

Optional Parameters:

type: The preposition that should join the contained work and the container; default is "in." If the word is "to" or "of," then no comma is prepended and the word is set in roman type, as is useful for introductions or forewords.

singleauthor: Indicates that all works in the contained volume are by the same author, such that the author name should be set in the font used for book authors.

inline: A short form name for the citation. The hereinafter parameter is probably better in most situations.

[21] This may be done through anonymous references as described in section 13.

Example:

```
\defcitecontainer{pantzer}{
    author=Katharine F. Pantzer,
    title={Printing the English Statutes, 1484--1640:
        Some Historical Implications},
    in=book: {
        title=Books and Society in History,
        editor=Kenneth E. Carpenter,
        year=1983
    },
    page=69,
}
```

Produces:

Katharine F. Pantzer, *Printing the English Statutes, 1484–1640: Some Historical Implications*, *in* Books and Society in History 69 (Kenneth E. Carpenter ed., 1983).

This is the book-chapter type of cite container, where the contained item has a title and author but is not of its own reference type.

Example:

```
\defcitecontainer{adams-lee}{
    citation=letter: {
        author=Samuel Adams,
        to=Arthur Lee,
        date=july 31 1771,
    },
    vol=2,
    page=173,
    in=book: {
        title={Life of Arthur Lee, LL. D.},
        author=Richard Henry Lee,
        year=1829,
        publisher={Boston, Wells and Lilly},
    },
}
```

Produces:

Letter from Samuel Adams to Arthur Lee (July 31, 1771), *in* 2 Richard Henry Lee, Life of Arthur Lee, LL. D. 173 (Boston, Wells & Lilly, 1829).

This exemplifies the first type of cite container, where a standalone reference (here, a letter) is republished inside another reference (here, a book). The contained and container references are both defined anonymously (see section 13).

Example:

```
\defcitecontainer{rastell}{
    citation=citecontainer: {
        author=John Rastell,
        title=Prohemium,
        type=to,
        in=book: {
            title=The Abbreviation of the Statutes,
            year=1519,
        },
    },
    vol=1,
    page=327,
    in=book: {
        title=Typographical Antiquities,
        year=1785,
        editor=Joseph Ames,
        editor=William Herbert,
        publisher={London, Society of Antiquaries},
    },
    type=reprinted in,
}
```

Produces:

John Rastell, *Prohemium* to The Abbreviation of the Statutes (1519), *reprinted in* 1 Typographical Antiquities 327 (Joseph Ames & William Herbert eds., London, Soc'y of Antiquaries 1785).

This defines an introduction (Prohemium) to a publication of statutes (the *Abbreviation*), where no copies of the publication itself are known to remain but the introduction alone was reprinted in another compilation called *Typographical Antiquities*. The Prohemium is thus properly defined as a cite container (of the second type) where the container is the *Abbreviation*, but then that whole thing is wrapped inside another cite container for *Typographical Antiquities*. This is probably the most complex reference definition that I have constructed.

35 Articles and Manuscripts

jrnart: A consecutively paginated journal article

This reference type is used for articles published in consecutively paginated journals.

Key Parameters:

author, **name**: The author(s) and title of the article.

vol, **rep**, **page**, **cite**: The citation information for locating the article. The journal title is given in rep, and should be unabbreviated as it will be abbreviated according to the journal scheme. If the page is "forthcoming," then it is treated as if the forthcoming parameter were set.

forthcoming: Indicates that the article is not yet published. The date should match the date of an available draft if one exists, or the expected date of publication.

year: The year of publication. If this is identical to the volume number, then it is omitted.

Optional Parameters:

type: The type of article (Note, Essay, Comment, etc.).

citation: Instead of an actual title, an article can use a case citation as a name. In this case, citation is the reference name of the case to be used as a title.

issue: The issue number of the journal in which the article appeared.

useissue: Include the issue number, which by default is omitted.

Example:

```
\defjrnart{griswold}{
    author=Erwin N. Griswold,
    title=Government in Ignorance of the Law---A Plea
    for Better Publication of Executive Legislation,
    cite=48 Harvard Law Review 198,
    year=1934,
}
```

Produces:

Erwin N. Griswold, *Government in Ignorance of the Law—A Plea for Better Publication of Executive Legislation*, 48 Harv. L. Rev. 198 (1934).

Note the unabbreviated journal name in the input.

procart: A non-consecutively paginated proceedings article

An oddity of traditional legal citations is that scholarly works published in journals that do not use consecutive pagination are cited as if they were magazine articles, suggesting that they are of lower prestige. This citation format uses a more journal-like citation format, making articles appear in citation form more like traditional journal articles.

The parameters are largely the same as those for jrnart, with the following exceptions.

Key Parameters:

page: This should be an identifier of the article being cited, probably along the lines of `No. 12`. The pin cite will be placed after the article title rather than after the journal name, following the usual convention that the page number follows the item that is sequentially paginated.

issue: If given, it is equivalent to setting page to `No.` ⟨*issue*⟩.

magart: A magazine article

This should be used for non-consecutively paginated journals that do not carry scholarly weight.

Key Parameters:

author, **name**: The author(s) and title of the article.

rep, **page**: The citation information for locating the article. The journal title is given in rep, and will be abbreviated according to the journal scheme.

year: The issue date of the magazine.

Optional Parameters:

publisher: The magazine publisher, if relevant.

Example:

```
\defmagart{duhigg}{
    author=Charles Duhigg,
    title=How Companies Learn Your Secrets,
    journal=New York Times Magazine,
    page=!MM30,
    date=Jan 1 2016,
}
```

Produces:

Charles Duhigg, *How Companies Learn Your Secrets*, N.Y. Times Mag. MM30 (Jan. 1, 2016).

Note the exclamation mark in front of the page number, which must be included because the page number does not start with a digit (see section 17).

newsart

Alias for magart.

website: A web site or generic citation

This reference type is useful as a catch-all for published works that do not fit cleanly in any other category. The "website" denotation is somewhat irrelevant, as the url parameter is usable independent of reference type.

The parameters are the same as those for magart. However, rep is optional, which can be occasionally useful for web pages that don't have a site title separate from a particular page title. If given, rep will be abbreviated according to the journal scheme. Any book publication parameters described in section 34.1 are also permitted.

Example:

```
\defwebsite{mitzman}{
    author=Dany Mitzman,
    title=Nutella: How the World Went Nuts for a Hazelnut
        Spread,
    date=may 18 2014,
    journal=BBC News,
    url=http://www.bbc.com/news/magazine-27438001
}
```

Produces:

Dany Mitzman, *Nutella: How the World Went Nuts for a Hazelnut Spread*, BBC News (May 18, 2014), http://...

manuscript: An unpublished manuscript

This reference type is usable for any unpublished written work, including Ph.D dissertations, manuscripts, and reports.

Key Parameters:

author, **name**: The author(s) and title of the manuscript.
type: The type of work, by default "manuscript."
onfile: Where a copy of the work can be found; "author" is a good choice.
year: The date of the manuscript.

Example:

```
\defmanuscript{princess-bride}{
    title=The Princess Bride,
    author=William Goldman,
    type=draft screenplay,
    date=may 3 1986,
}
```

Produces:

William Goldman, The Princess Bride (May 3, 1986) (unpublished draft screenplay).

workingpaper: A working paper

This is for working papers with a serial number issued by an institution.

Key Parameters:

author, **name**: The author(s) and title of the paper.
publisher: The name of the institution publishing the working paper.
number: The serial number, with series name (e.g., "Economics Working Paper No. X"). See the serial param to set this and the publisher simultaneously.
year: The date of the paper.

Example:

```
\defworkingpaper{cockburn}{
    author=Iain M. Cockburn,
    author=Megan MacGarvie,
    title={Patents, Thickets, and the Financing of Early-Stage
        Firms: Evidence From the Software Industry},
    sponsor=National Bureau of Economic Research,
    serial=Working Paper No. 13644,
    year=2007,
    url=http://www.nber.org/papers/w13644.pdf
}
```

Produces:

Iain M. Cockburn & Megan MacGarvie, *Patents, Thickets, and the Financing of Early-Stage Firms: Evidence from the Software Industry* (Nat'l Bureau of Econ. Res., Working Paper No. 13644, 2007).

36 Foreign and International Materials

The selection of reference types defined here is very much incomplete. To the extent it is necessary, use verbatim.

engdebate: An English parliamentary debate

Key Parameters:

vol, **rep**, **page**, **cite**: The locator information for the debate. The reporter should include the House of Parliament abbreviated.

engstat: An English statute

This supports both modern statutes that are arranged by section numbers, and historical statutes that are reported in *Statutes of the Realm*. Note the difference in page for each.

Key Parameters:

name: The name of the statute.
year: The year of enactment. The year is assumed to be part of the name, appended to the name after a comma as is conventional for U.K. statutes.
regnal: The regnal year specification.
chapter, **number**: The chapter number of the statute.
vol, **rep**: The volume and code name for codified statute publications (primarily "Stat. Realm").
page: If rep is given, then this parameter should receive the page number in the reporter for the statute. Otherwise, it should receive a section number of the statute if any.

Pin cites given in citation items will use the pin cite joining algorithm given in section 18.

engcommand: A command paper in England

Key Parameters:

instauth: The government department issuing the paper.
name: Title of the paper.
year: Year the paper was issued.
number: The command paper number. This must be prefixed with the appropriate "Command" abbreviation that identifies the series, as is conventional in U.K. practice.

foreignconst: A foreign constitution

Key Parameters:

name: The name of the constitution. A translation may be given in square brackets.
page: A pin cite for the part of the constitution being cited.

treaty: A treaty or international agreement

Key Parameters:

- **name**, **inline**: The full and short names for this reference, as described with regard to statute names in section 32.1.
- **countries**: Countries that are parties to the treaty. They will be abbreviated. The names should be separated with en dashes. Countries need not be named for large multilateral treaties.
- **year**: Date the treaty was signed. Use qualifiers "done" or "opened for signature" if there is no single signing date.
- **vol**, **rep**, **page**, **cite**: Locator information for the treaty, if available.

eucase: A European Union case

Key Parameters:

- **name**, **inline**: The full and short names of the case.
- **docket**: The number of the case.
- **vol**, **rep**, **page**, **cite**: Locator information for the case.
- **court**: The court issuing the decision.
- **year**: The year of decision.

unresolution: A U.N. resolution

Key Parameters:

- **name**: The name of the resolution.
- **number**: The serial number of the resolution, with the body and abbreviation for the resolution (e.g., "G.A. Res. 123").
- **type**: The institution issuing the document control number, default "U.N."
- **intdoc**: The international institution control number, typically containing slashes.
- **year**: The date of the document.

intdoc: A U.N. or international document

Typically these are reports.

Key Parameters:

- **author**, **instauth**: The author of the document. The institutional author will be separated from the human author, with the latter placed in a parenthetical.
- **name**: Title of the document.
- **type**: The institution issuing the document control number, default "U.N."
- **intdoc**: The international institution control number, typically containing slashes.
- **year**: The date of the document.

frcase: A French case

Key Parameters:

- **court**: The name of the court.
- **number**: The decision number.
- **year**: The year of decision.

rep, **page**: The locator information for the decision.

frstat: A French statute

Key Parameters:

number: The law number.
year: The year of enactment.
rep: The statutory compilation.
pubyear: The year of publication of the statutory compilation.
page: The page in the compilation.

37 Specific Reference Types

These reference types could fall under the umbrella book reference type, but it is helpful to include them to provide unique formatting and/or to save on data entry.

cjs: Corpus Juris Secondum

Key Parameters:

vol: The volume number.
name: The entry name.
year: The year of publication.

rfc: An IETF Request for Comments

This reference type automatically adds the serial number, inline citation format, and URL based just on the RFC number.

Key Parameters:

author, **name**, **year**: Bibliographic information for the RFC.
number: The RFC number.
Any book publication parameters described in section 34.1 are also permitted.

38 Other Reference Types

letter: A letter or memorandum

The defining characteristic of this reference type is that it is a communication sent from one party to another. Nevertheless, the recipient is optional and can be omitted, useful perhaps for open letters or memoranda not directed to any particular person.

Letters between institutions are typically still signed by and addressed to individuals. As a result, the ability to attach institutional affiliations to human authors (see section 8) is especially useful for these references, to identify the companies, agencies, or institutions that are the actual communicants.

Key Parameters:

author: The sender of the letter.
to: The recipient of the letter.
date: The date the letter was sent.

Optional Parameters:

type: The type of document, default “Letter.”
name: A subject line or title of the letter (more common for memoranda).

Example:

```
\defletter{alice-memo}{
   date=may 13 2013,
   type=Memorandum,
  author={Andrew H. Hirshfeld, United States Patent and Trademark Office},
   instto=Patent Examining Corps,
  title=Federal Circuit Decision in \emph{CLS Bank et al. v. Alice Corp.},
}
```

Produces:

Memorandum from Andrew H. Hirshfeld, U.S. Patent & Trademark Off., to Patent Examining Corps, *Federal Circuit Decision in* CLS Bank et al. v. Alice Corp. (May 13, 2013).

pressrelease: A press release

Key Parameters:

name: The title of the press release.
year: The date of the press release.

Optional Parameters:

type: The type of document, default “Press Release.”
author, **instauth**: The name of the issuer of the press release. This is optional and not included in other citation systems, but unless the issuer is in the press release title, there may be no other way to identify who published the press release.

Example:

```
\defpressrelease{lockyer-pr}{
title=Attorney General Bill Lockyer Files ``Friend of Court'' Brief over Unocal
Gasoline Patent,
date=Sep 14 2000,
}
```

Produces:

Press Release, *Attorney General Bill Lockyer Files "Friend of Court" Brief over Unocal Gasoline Patent* (Sept. 14, 2000).

speech: A speech, address, or presentation

Besides the obvious uses for delivered speeches, this reference form is useful for citing slide decks made available online. The more conventional practice of citing slide decks as books is undesirable insofar as it gives more apparent authoritative weight than is due most slide decks.

Key Parameters:

author: The speaker or presenter.
year: The date the speech was given.

Optional Parameters:

type: The type of speech, default "Address." For slide decks, "Presentation" is a good choice.
place: The location, event, or institution where the speech was given.
name: A title of the speech.

Example:

```
\defspeech{purvis}{
    author={Sue A. Purvis, U.S. Patent and Trademark Office},
    type=Presentation,
    title=The Role of the Patent Examiner,
    date=april 8 2013,
}
```

Produces:

Sue A. Purvis, U.S. Patent & Trademark Off., Presentation: The Role of the Patent Examiner (Apr. 8, 2013).

film: A film (movie/motion picture)

This format is somewhat deprecated as it can be implemented entirely with website.

Key Parameters:

name: The name of the film.
publisher: The studio that published the film. It is unclear which entity this is when there are multiple studios involved in the distribution chain.
year: The year the film was released.

tvshow: A television show

This format is somewhat deprecated as it can be implemented entirely with website.

Key Parameters:

name: The name of the show.
publisher, **broadcaster**: The broadcaster of the show.
year: The year the show aired.

opinion: An ethics or other opinion

These are issued by the American Bar Association, for example. Opinions issued by the government should use a different reference type.

Key Parameters:

author: The author of the opinion.
number: The serial number of the opinion.
year: The date of the opinion.

Optional Parameters:

type: The type of document, default "Formal Opinion."

39 Non-Document References

These references and reference types are not associated with documents or authorities, but serve other purposes in this package's system.

abbrev: An abbreviation for a name

The full citation form produces the unabbreviated name and a definitional parenthetical, and the short citation form produces the abbreviated name. This reference type should only be used in inline citations.[22] The abbreviation reference type makes use of the short name management features of this package described in section 22. If an abbreviation is used only once, the parenthetical is automatically suppressed. It also manages use of "The" at the beginning of names correctly, in accordance with `\adjective` as described in section 14. Thus, if the following is defined:

```
\defabbrev{fda}{
    name=the Food and Drug Administration,
    inline=the FDA
}
```

then the following will be produced:

Input	Output
`\inline{fda}` (used only once)	the Food and Drug Administration
`\inline{fda}`	the Food and Drug Administration ("FDA")
`\inline{fda}`	the FDA
`\adjective{fda} regulations`	FDA regulations

This reference type produces invisible inlines (see section 14.3(a)) so that it does not affect *id.* citations surrounding it.

Key Parameters:

name: The unabbreviated name. Include "The" at the beginning if that is how the full name is used in ordinary speech.

inline: The short or abbreviated name. Include "The" at the beginning if that is how the abbreviation is used in ordinary speech.

alias: An alias for a citation reference

Aliases a reference name to another. The main advantage is that the alias can have a default volume and pin cite. For example, consider the alias:

```
\defalias{sec-5}{
    name=ftc-act,
    page=S 5
}
```

with `ftc-act` defined as the relevant statute. Now, the following two are equivalent:

[22] In future versions, it is hoped that named abbreviations can be used as institutional author titles.

```
\sentence{sec-5}.
\sentence{ftc-act at S 5}.
```

Moreover, pin cites given for the alias in citations use the pin cite joining algorithm given in section 18. Thus:

```
\sentence{sec-5 at /a}.
```

will cite § 5(a) of the statute.

Key Parameters:

citation, **rep**: Reference name to be aliased.
page: Pin cite to be used as the default pin cite for the alias.

selfcite: A self-citation

This is useful for quickly replacing self-citations when submitting to journals that require anonymity. No parameters are required; a unique name is automatically generated for the citation.

`this`

Reference `this` is not a reference type but the name of a single predefined reference, used for cross-references. See section 10.

`verbatim`

Reference `verbatim` is not a reference type but the name of a single predefined reference, the purpose of which is to enable arbitrary text in a citation item. When using `verbatim` in a citation item, the pin cite text is not parsed but is used verbatim as the formatted citation text:

```
\sentence{see verbatim at {the \textbf{Moon}} (shining)}.
```

Produces:

See the **Moon** (shining).

Note how signals and parentheticals are still parsed and formatted around the user-defined citation text.

Part VII
Reference Parameters

These are all of the parameters that may be given when defining a reference. Parameters in *italics* are aliases for others.

The long history of development of this package has meant that some parameter names do not conform with the naming conventions otherwise followed in this package. In particular, inline and related parameters relate not to inline citations but to short reference names (see section 22), and date parameters typically end in `year` even though full date specifications are permitted (see section 9). Generally, aliases to properly conforming names are provided.

40 Generally Applicable Parameters

These parameters may be used in defining any reference, regardless of type.

accessdate

URL access date. This is added as a parenthetical immediately after the url parameter. (It can be used without url, perhaps to indicate the access date of a non-linked online resource such as a case in a database service.) The date qualifier "last visited" is prepended to the date unless another qualifier is given.

bookreview

[Value optional] Indicator that an article is a book review. This adds a parenthetical. If no value is given, then the parenthetical reads just "book review." Otherwise, the value should be a reference name for the book being reviewed, which will be cited in the parenthetical.

toacat

Table of Authorities category for this citation. The valid category types are listed in section 20.1.

comment

Alias for paren. This parameter was originally used for commentary, but now the paren parameter handles those as well.

country

Country for citation. This should generally be added to any reference with legal authority outside the United States. The country name is automatically abbreviated.

defaultopt

The default optional parameter. Some reference types that use the optional argument will use this value if the optional argument is unset.

hereinafter

Text to use for hereinafter citation form. This parameter will define the short-form citation for the reference.

hyphenate

Set hyphenation for a word. This just calls `\hyphenation` on the word(s) given in the value. It is convenient if unusual hypheniation is required for a word in the reference (e.g., a named party to a case), because it keeps the hyphenation as bibliographic data.

hyphenation

Alias for hyphenate.

kind

Type of reference. This is automatically set when the `\def`⟨*kind*⟩ command is called to define a reference, and it should not be changed.

noetal

[Value optional] No et al. with 3 or more authors. This parameter has no display effect, but advises the package not to issue a warning if it detects an name list of three or more entities.

notoa

[Value optional] Do not include in the table of authorities.

opturl

Optional URL. Generally this parameter is ignored. However, if the package option useopturl is set, then this parameter is an alias for url. This is useful for references where a URL could be helpful for some publications but not others, as it enables the writer to quickly switch between presentation forms.

origyear

Year of original publication of the reference. This is typically used when a modern edition of a pre-1900 work is cited, such that the year parameter is set to the modern publication year and this parameter set to give the original publication date of the work.

origdate

Alias for origyear.

parallel

Parallel citation. This must be entered in the form ⟨*key*⟩**:** ⟨*cite*⟩. The ⟨*key*⟩ is used as a pin cite segment key, described in section 17. When the relevant reference is cited, the parallel cite is added before the date parenthetical, along with a page number found in the appropriate section of the given pin cite if present. The ⟨*cite*⟩ text is used as given and not parsed or abbreviated.

Note: Currently the parallel citation is only displayed in the full citation. This is because the package has no mechanism for displaying document-level parentheticals in short citations. Hopefully this will be fixed in the future.

paren

A parenthetical to be included with citation. The parenthetical will be included in any full citation. If the parenthetical contains the words `in:`, `by:`, or `at:`, it is treated as commentary such that what follows the colon is a citation string to be formatted. Thus, `quoted in: ref` will produce a parenthetical "(quoted in `\clause{ref}`)."

parse

Apply automatic citation parser. The value is a formatted reference input as described in section 7. It is useful if that input contains an equal sign, which would otherwise cause this package to mistakenly interpret the formatted input as parameter-value input.

reprinted

Reprinted in parenthetical. The value should be a reference identifier or an anonymous reference defintion as described in section 13.

supplement

Year/date of a supplement to the reference. The value should be a date (see section 9), for which the date qualifier "Supp." will be added if no other qualifier is given. If both this parameter and year are set, then the two will be joined with an ampersand (e.g., "1995 & Supp. 1996").

toapage

[Value optional] Include page numbers in Table of Authorities. Generally, Table of Authorities entries are for individual references; this parameter will cause an entry to be generated for each page of the reference cited. Some reference types, such as statcode, do this by default. Additionally, some reference types, such as book, will format Table of Authorities entries differently when this parameter is given.

url

URL for reference. See section 11 for information on how to present the URL and how it is formatted.

41 Common Parameters

These parameters are frequently used across many reference types. A general description of each is given, but unlike the generally-applicable parameters in section 40, some reference types may use these parameters in special or idiosyncratic ways, which should be described in the documentation for each reference type.

authln

Last name of the first author, for short citations. The author parameter will set this parameter automatically. This parameter can be set after giving the authors to customize the last name.

author

Author of the reference. The value should be a name as described in section 8. If there are multiple authors, provide this parameter multiple times, but see also parameter noetal if entering three or more names. For institutional authors, see parameter instauth.

If at least one author is given, then the author names will automatically be used as the inline citation forms for the reference.

cite

Volume-reporter-page citation. The value for this parameter should follow the syntax:

> ⟨*vol*⟩ ⟨*rep*⟩ ⟨*page*⟩ [(⟨*year*⟩)]
>
> *Example:*
>
> `123 F.3d 456 (2009)`

and this parameter will set the parameters vol, rep, page, and year accordingly. See the discussion of volume-reporter-page parsing in section 7 for more details.

date

Alias for year.

inline

Short name for the reference.

instauth

Institutional author. Adds an institutional author to the author parameter list.

jcite

Alias for cite.

journal

Alias for rep.

journaltitle

Alias for rep.

booktitle

Alias for rep.

name

Name of the reference. Nearly every reference type accepts this parameter, which generally should be the title of the reference. See the relevant reference type description for more information.

number

Number (for constitutional amendments, session laws). The `number` parameter generally is used for a serial number associated with the work, which may or may not contain text prefixes. Examples are patent numbers (patent), bill numbers (bill), and constitutional amendments (constamend). For most reference types (patents being an exception), the number is not analyzed or formatted in any way.

For book and workingpaper references, consider also the serial parameter to set the publisher parameter as well.

page

Page number. See section 17 for the usual formatting of this parameter's value.

pages

Alias for page.

rep

Reporter or journal name.Whether the value of this parameter needs to be abbreviated will depend on the reference type.

series

Alias for number.

short

Alias for inline.

shorttitle

Alias for inline.

src

Alias for rep.

subdiv

Alias for page.

title

Alias for name.

type

Type of work. For jrnart references, this would be “Note” or “Comment,” for example. Other types of references use this parameter to replace default values indicating the work type: comments for example will use this value instead of the word “Comments.” The use of this parameter is highly specific to the reference type.

vol

Volume number.

volume

Alias for vol.

year

Year/date of the case, article, regulation.

42 Parameters for Cases

These parameters are specifically for case and similar references.

agency

Administrative agency name. Essentially the same as court except the name will be abbreviated.

court

Court deciding the case. The court name will *not* be abbreviated, at least currently.

d

Second party name in the case. See section 30.1 for the input syntax for this parameter.

dbid

Database identifier for an online case.

docket

Docket number of the case. If the value starts with a number, the word “No.” will be automatically prepended to the docket number.

enbanc

[Value optional] En banc case parenthetical. Although intended for cases, this parameter can be attached to any reference.

inlinedefendant

[Value optional] Don’t use the first party as main party. For case citations, the first party named is typically used as the short name (with some exceptions). This parameter forces the second party to be used.

mem

[Value optional] Memorandum opinion parenthetical. Although intended for cases, this parameter can be attached to any reference.

p

First party name in the case. See section 30.1 for the input syntax for this parameter.

parties

Named parties to a case, separated by “ v. ”. This is a shortcut for setting parameters p and d.

percuriam

[Value optional] Per curiam opinion.

prior

Prior history for a case. The syntax for this parameter is described in section 30.4.

revparties

[Value optional] Parties to this case are reverse of named case.

sameparties

[Value optional] Parties to this case are same as named case.

slip

[Value optional] Slip opinion or law.

slipop

Alias for slip.

subsequent

Subsequent history for a case. The syntax for this parameter is described in section 30.4.

43 Parameters for Legislative Materials

These parameters are specific to legislative materials, including statutes and bills.

bill

Alias for number. As described for bill references, the bill (or number) parameter should be the full identifier of the bill (e.g., `S. 123`).

chapter

Session law chapter. This sets parameter number to the given value, and changes parameter sesslawid to “ch.” It is used for older statute references that were organized by chapters rather than law numbers.

codified

Citation for codification of law or regulation. The value should be a formatted citation (as described for parameter cite). Typically this parameter should be given for reference types like statsess or govdoc, but it can be given for anything.

codifiedamended

Citation for codification of law or regulation. Like codified except the text before the citation will be “codified as amended at.”

committee

House or Senate committee. The name will be abbreviated.

congress

Congressional session number. This has to be a parameter separate from number because bill references already use number for the bill identifier.

origsect

Original section number of a statute in a code. This is used, for example, when a U.S. Code section corresponds to a section of a statute with different numbering (for example, section 337 of the Tariff Act of 1930 being 19 U.S.C. § 1337). It is specific to statcode references, and should be obviated with the newer reference type stattitle.

publiclaw

Alias for number.

publno

Alias for number.

sesslawid

Alias for type. By default, this is “Pub. L. No.,” and this parameter enables changing it.

regnal

Regnal citation for English pre-1963 statutes.

status

Status of bill.

44 Parameters for Books and Articles

This section describes parameters typically specific to books and articles not otherwise described above. Some of the parameters will be applicable to other reference types such as statutes, because they use the publication parenthetical described in section 34.1. This section also includes parameters for the citecontainer reference type, because those are typically books.

edition

Edition number of book. If the value is a number, then it will be converted to an ordinal number (e.g., 3 becomes “3d”) and the word “ed.” will be appended to it. Otherwise, the value is abbreviated in accordance with the pub abbreviation scheme.

editor

Editor(s) of book. The value should be a name, as described in section 8. For more than one editor, enter this parameter as many times as needed, but noetal must be used if there are three or more editors. The “editor” parameter is also used for translators or other roles; see edtype. For institutional editors, see parameter insted.

edtype

Editor type (default ed.). Changes the abbreviation placed after the editors. For example, use “trans.” for translators, and “eds. & trans.” for editors who also translated. Note that there is currently no support for works with an editor who is different from a translator. This will be abbreviated according to the pub abbreviation scheme.

forthcoming

[Value optional] Journal article is forthcoming.

in

Reference name of a citation that the current item is in. The value may be a reference name or an anonymous reference definition (see section 13). The primary use of this parameter is to identify the containing work in a citecontainer.

insted

Institutional editor. Adds an institutional editor to the editor parameter list.

issue

Issue number of journal.

issuer

Alias for publisher.

noabbrevjrn

[Value optional] Don't abbreviate the journal name. This is useful if the journal was entered abbreviated already.

publisher

Publisher of book. The publisher name will automatically be abbreviated by the name abbreviation scheme.

serial

Serial number of a published work. This parameter is essentially an alias for number, except if it contains a comma or colon, the prefix is used to set the publisher parameter.

singleauthor

[Value optional] All works in a collection are by one author. This is used in citecontainer to change the font used for the author.

usetitle

[Value optional] Use title rather than author in short cites.

useissue

[Value optional] Include issue number in journal citations.

45 Other Parameters

These parameters are used by other specific reference types.

broadcaster

Alias for instauth.

citation

Citation to another reference. The value may be a reference name or an anonymous reference definition (see section 13). Different reference types will use this parameter for different purposes. For example, jrnart will cite the reference as the article title (used for notes reviewing cases), casedoc will use it as the case in which a document was filed, and citecontainer uses it as the contained work (the container work is given in the parameter in).

commenter

Names of parties submitting agency comments.

countries

Countries to a treaty. The country names will be abbreviated.

inlineparen

[Value optional] Always include a parenthetical for inline form.

instto

Institutional recipient of letter, memorandum, etc.. Adds an institutional recipient to the to parameter list.

intdoc

Document number for international documents. International resolutions and other documents may have two numbers: a serial number and a document control number issued by the international body. This parameter is for the second number. It should not include a prefix such as "U.N. Doc.," as that will be added separately.

issuedate

Alias for issueyear.

issueyear

Date a patent was issued. This is equivalent to `date=issued` ⟨*date*⟩.

noinlineparen

[Value optional] Never include a parenthetical for inline form.

noshorttitle

[Value optional] Do not include title for short cites.

onfile

Where the document is on file.

place

Place (for speeches, constitutions, statutes).

producer

Alias for instauth.

pubdate

Alias for pubyear.

pubyear

Year of publication (for French statutes). This is an additional date parameter used for French statutes.

sponsor

Alias for publisher.

state

Alias for place.

struct

Structure for data of multivolume works. See the discussion of parameter structures in section 12.

to

Recipient of a letter. This is primarily used for letter references. The value should be a name as described in section 8. If there are multiple recipients, provide this parameter multiple times, but see also parameter noetal if entering three or more names. For institutional authors, see parameter instto.

Part VIII
Supporting Packages

Hereinafter includes several supporting packages, primarily consisting of lower-level tools for macro writing. They are distributed in standalone form in case they are useful for other package authors.

46 String Processing Macros

This package provides a variety of low-level text processing commands and macro helpers. The section in the main *Hereinafter* documentation just provides an overview of features; the full documentation for this file should be consulted for the particular macros and their usage.

Initially, the package provides several syntatic sugar macros for conditionals, definitions, and expansion control. These are similar to the `etoolbox` package.

46.1 Expansion Control

Although, there are several useful macros defined in this section, the notable one is `\@expand`, which expands a token multiple times. It takes three arguments: the callback to be executed when done expanding, the stuff to expand, and a count of how many expansions are to be performed, represented by the number of tokens in the argument. Thus:

```
\def\macro{Hello World}
\@expand{\process}{\csname macro\endcsname}{ii}
```

would expand `\csname macro\endcsname` twice (once for each “i”), obtaining the meaning of `\macro`, and then run the callback `\process`, to result in:

```
\process{Hello World}
```

46.2 Finders

These macros provide convenient ways of defining text finders that, when called, are fully expandable. They provide the ability to find text ⟨*needle*⟩ at the beginning, middle, or end of text ⟨*haystack*⟩. The pattern is to call:

```
\make@find@⟨type⟩{⟨needle⟩}
```

to initialize the finder, and then:

```
\find@⟨type⟩{⟨needle⟩}{⟨haystack⟩}
     {⟨true-callback⟩}{⟨false-callback⟩}
```

to execute the finder. ⟨*type*⟩ is `in`, `start`, `end`, or `eq` depending on the type of matching desired.

The callbacks for finders are given consistent arguments. The ⟨*false-callback*⟩ always receives no arguments. The ⟨*true-callback*⟩, however, receives arguments indicating the remaining unmatched text. Thus, for `\find@in`, the ⟨*true-callback*⟩ receives two arguments, the prefix and suffix relative to the matched text. However, the callbacks for `\find@start` and `\find@end` receive only one argument, and `\find@eq`’s callback receives none.

46.3 Cardinal Numbers

The `\cardinaltext` macro changes a number into its capitalized, cardinal textual form. For example, “123” becomes “One Hundred Twenty-Three”.

46.4 Text Tokenizer

Often it is useful to tokenize a string of TeX tokens into words, in a way that properly handles macros and groups, and differentiates the tokens by type. The following macros help to do that.

The `\find@next` macro finds the next single token by type and runs a command on it. This macro can detect spaces, groups, and characters at the beginning of a string. Its usage is as follows:

```
\find@next{⟨str⟩}{⟨space⟩}{⟨group⟩}{⟨char⟩}{⟨end⟩}
```

⟨*str*⟩:	the string to read
⟨*space*⟩:	callback for a space
⟨*group*⟩:	callback for a group
⟨*char*⟩:	callback for a character
⟨*end*⟩:	callback if the string is empty

The ⟨*space*⟩, ⟨*group*⟩, and ⟨*char*⟩ callbacks each take two arguments, the found item and the remainder of the string. The ⟨*end*⟩ callback receives no arguments.

Finds the next word in the given string and runs a callback on it. This is like `\find@next` except (1) it aggregates letters into a single word, and (2) it differentiates letters from other characters. Its usage is as follows:

```
\find@word{⟨str⟩}{⟨sym⟩}{⟨group⟩}{⟨word⟩}{⟨end⟩}
```

⟨*str*⟩:	the string to read
⟨*sym*⟩:	callback for a symbol (space or non-letter character)
⟨*group*⟩:	callback for a group
⟨*word*⟩:	callback for a word
⟨*end*⟩:	callback if the string is empty

46.5 Searching for Braced Groups

The `\find@group` macro finds a braced group in a string and runs a callback with three arguments: the pre-group text, the group (without braces), and the post-group text.

The `\find@group@end` macro similarly finds a braced group, but only at the end of a string. The callback receives only two arguments.

47 Abbreviations

This package provides for abbreviating texts according to tables of abbreviations. It also provides functions for removing spaces between dotted initials. The abbreviation algorithms are fully expandable so that they may be used within `\edef` definitions.

47.1 Abbreviation Classes

There may be multiple abbreviation classes, namely separate schemes of abbreviations with different words abbreviated in each.

An abbreviation class is defined in the macro `\abb@class@`⟨*name*⟩. Within that class are a further set of macros of the form `\abb@class@`⟨*name*⟩`@`⟨*word*⟩, where ⟨*word*⟩ is the first word of the text to be abbreviated. The value of the macro is a list of two-item pairs of the form:

> {⟨*remaining text*⟩}{⟨*abbrev.*⟩}

`\abb@new{`⟨*class*⟩`}` creates a new abbreviation class.

`\abb@add{`⟨*class*⟩`}{`⟨*word*⟩`}{`⟨*abbrev*⟩`}` Adds an item to the abbreviation class. ⟨*class*⟩ is the class, ⟨*word*⟩ the unabbreviated form, and {⟨*abbrev*⟩} the abbreviated form.

If the unabbreviated form starts with an empty set of braces, then the given abbreviation pair is treated as an initial-text abbreviation, which will only abbreviate at the beginning of a text. That is, the abbrevation pairs:

> {}Building → Bldg.
> Building → Build.

would produce abbreviations as follows:

> The Building and Loan Office → The Build. and Loan Office
> Building and Loan Office → Bldg. and Loan Office

47.2 Algorithm for Abbreviation

`\abb@abbrev{`⟨*class*⟩`}{`⟨*text*⟩`}{`⟨*callback*⟩`}` is the entrypoint for abbreviating a text. ⟨*class*⟩ is the class, ⟨*text*⟩ is the text to abbreviate, and ⟨*callback*⟩ is a callback function for the result.

Text is abbreviated by reading each word in the text, using `\find@word` from section 46.

1. If it is not a word or punctuation, do not process the item and continue.
2. Otherwise, retrieve the list of abbreviations starting with this word. If there are none, then continue.
3. Iterate through each of the abbreviations associated with the word, and see if the remainder of the text to be abbreviated starts with any of the abbreviation texts. If so, then insert the relevant abbreviation text, correcting for dots as appropriate.

47.3 Removal of Spaces Between Dotted Initials

`\abb@initialdots{`⟨*text*⟩`}{`⟨*dot*⟩`}{`⟨*callback*⟩`}` processes a text to remove dots between single-letter initials. Given a text, it will convert sequences such as “A. A. Milne” to “A.A. Milne” and “F. 4th” to “F.4th”. ⟨*text*⟩ is the text to process and ⟨*callback*⟩ is the callback to run on the resulting output.

⟨*dot*⟩ is the dot character to search for. According to some abbreviation schemes, it is desirable not to collapse spaces between single initials when the initials refer to different classes of words. The macro `\abb@dot` can be used in the place of the period character to indicate this differentiation between classes.

For example, the text “Northern University Law Journal” may be abbreviated to “N.U. L.J.” with only the first and third spaces collapsed, but not the second because institution words should not be joined with publication words. This can be implemented by abbreviating publication words with `\abb@dot` (shown here for an abbreviation class `journal`):

```
\abb@new{journal}
\abb@add{journal}{Northern}{N.}
\abb@add{journal}{University}{U.}
\abb@add{journal}{Law}{L\abb@dot}
\abb@add{journal}{Journal}{J\abb@dot}
```

Now when the text is abbreviated with `\abb@abbrev`, it produces:

```
N.␣U.␣L\abb@dot␣J\abb@dot
```

The dot removal algorithm can now be run twice, once with a period as ⟨*dot*⟩ and once with `\abb@dot`, thereby producing the desired text.

48 Sorted Lists

This package provides for insertion sorting of lists.

A list can have a condition for testing whether items ought to be included in it. That is set with `\ListInclusionMacro{`⟨*list*⟩`}`. A common test is that list elements must be unique; that is done with `\ListElementsMustBeUnique{`⟨*list*⟩`}`.

`\AddToList{`⟨*list*⟩`}`element adds an element to a list.

The `\ShowList{`⟨*list*⟩`}{`⟨*callback*⟩`}` macro executes ⟨*callback*⟩ on every item of the list, in order. ⟨*callback*⟩ should be in the form of a macro definition that accepts one argument, which will be set to the list item.

48.1 Sorting Functions

As explained above, sorted lists require a sorting function that takes two parameters and sets `\ifCompReverse` if the elements are in the wrong sorting order.

The following macros provide some standard sorting functions. Each will take two additional arguments besides the ones described, which will be the list elements for testing.

`\ReverseSort{`⟨*test*⟩`}` performs ⟨*test*⟩ on the list elements and gives the reverse result.

`\SortNum` sorts elements numerically; all elements must be numbers.

`\NoSort` performs no sorting, meaning that elements are listed in reverse order of addition. (Consider using `\ReverseSort` to get elements in original order of addition.)

`\SortLen` sorts elements by their length, counted by number of tokens.

`\SortEasyAlpha` sorts simple text elements alphabetically. Its input ought to have been converted by `\StripForAlpha` first.

`\StripForAlpha{`⟨*text*⟩`}{`⟨*callback*⟩`}` is a helpful function to precede alphabetical sorting. It takes ⟨*text*⟩, uppercases it, and removes all non-alphabetical characters from it. It also pads numbers of fewer than four digits, so that “Volume 3” is sorted before “Volume 14”.

`\SortAlpha` sorts elements alphabetically, after they have been stripped of non-alphabetical characters. (This macro is probably outdated in view of `\SortEasyAlpha` and `\StripForAlpha`.)

49 Biblatex Compatibility

Because metadata for legal citations is more structured and complex than for other citation systems, *Hereinafter* does not use Biblatex files as its primary input format. However, the `hibib` package provides a compatibility layer that uses Biblatex to process `.bib` files into *Hereinafter* data structures. The package also enables rudimentary production of `.bib` files from *Hereinafter* reference definitions.

49.1 Producing Biblatex Files

The `\hiBibFile{`⟨*file*⟩`}` macro initiates production of a Biblatex file. The command should be called before any *Hereinafter* references are defined.

49.2 Reading a Biblatex File

To use a Biblatex file for input, include it using the usual `\addbibresource` command. The `hibib` package patches Biblatex's input commands so that each time a reference is read, a corresponding *Hereinafter* reference is defined.

Note that for Biblatex to operate, the `.tex` file must be compiled first to produce a `.bcf` file, which is then processed through the `biber` program to produce a `.bbl` file that actually provides reference definitions. This package requires the same process. The document with citation commands must be compiled first, producing warnings about unknown references for every citation. Then `biber` may be called and the document recompiled.

49.2(a) Saving Biblatex Entries in *Hereinafter* Format The `\hiBibConvert{`⟨*file*⟩`}` macro constructs a *Hereinafter* format reference file based on any Biblatex entries read. This command should be called before any entries are read via `\addbibresource`.

49.3 Citation Command Compatibility

Biblatex uses different commands for inserting citations. Fundamentally there cannot be straightforward compatibility between those commands and *Hereinafter*, because the underlying data model of citations is different. Nevertheless, some partial compatibility is provided for the `\autocite` and `\autocites` commands, which could help with automatic conversion of Pandoc markdown documents.

Part IX

Appendix

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the definition; numbers in roman refer to the pages where the entry is used.