The `amsrefs` package

Michael Downes and David M. Jones
American Mathematical Society

Version 2.12, 2013/01/16

Contents

1 Introduction

The amsrefs package is a LaTeX package for bibliographies that provides an archival data format similar to the format of BibTeX database files, but adapted to make direct processing by LaTeX easier. The package can be used either in conjunction with BibTeX or as a replacement for BibTeX.

This document is written for anyone who wants to implement a new bibliography style for amsrefs or who is just curious about how the package is implemented. The reader should be familiar with the contents of the "User's Guide to the amsrefs Package" [1] (amsrdoc.tex).

For the publisher or implementor, the chief advantages of the amsrefs package are as follows:

Preservation of structure The internal structural information of the bibliography entries is not lost when they are imported from the database file into the LaTeX document. This takes on its greatest significance when archiving documents in LaTeX form or transmitting them to another user (such as a publisher).

Deferred formatting This means that the style of the bibliography can be readily changed without reimporting everything from the original database(s).

Setup requires only LaTeX knowledge All bibliography setup can be done in LaTeX; learning another programming language (such as the one used in BibTeX bst files) is unnecessary.

2 Package options

In addition to the options documented in the user's guide, there are a few additional options that were omitted either because they are obsolete or deprecated options included only for backwards compatability or because they are still considered experimental and not yet ready for widespread use.

? Informational option. This causes amsrefs to display a pointer to the User's Guide on the terminal an in the log file. (In previous versions, it displayed much more material, including a summary of package options.)

traditional-quotes, logical-quotes With the *traditional quotes* option (default), quotation marks produced by \bibquotes (§5) fall outside of other punctuation, "like this," whereas with the *logical quotes* option the order is reversed, "like this".

3 More about the \bib command

3.1 Field names for the \bib command

In addition to the fields discussed in the user's guide, the following fields are used internally:

fulljournal Used internally by `\DefineJournal`.

name Used internally by the `name` bibliography type and `\DefineName`.

transition A dummy field used inside `\BibSpec`s when we want to force an action unconditionally.

The following fields are included for backwards compatibility:

institution, school These are provided as aliases for `organization` for compatibility with BibTeX.

place A synonym for `address`. In earlier versions of `amsrefs`, `place` was preferred and `address` was considered as an alias for `place`. However, this seemed like a gratuitous incompatibility with BibTeX to me, so I have reinstated `address` as the primary field and `place` is now an undocumented alias.

The following fields are reserved for future use:

doi Digital Object Identifier

setup This is a special field that can be used to give arbitrary commands to be executed at the beginning of the current `\bib` entry, after all the fields have been read. The idea is that one can alter the formatting of an individual entry through this field, to handle special cases.

This is fully implemented, but I've been unable to think of any good examples of its use; so, I've decided to suppress it until such an example comes to light.

url Universal Resource Locator.

3.2 Bibliography entry types

The following additional entry types (or, really, pseudo-entry types) are used internally by `amsrefs`:

collection.article

proceedings.article

partial

conference

innerbook

name

nameLE

nameBE

nameinverted

publisher

The following are currently undocumented aliases for various of the standard types:

miscellaneous

periodical

4 Customizing the bibliography style

If you use the `amsrefs` package as is, the bibliography style you get is the kind of style customarily seen in AMS publications. The recommended way to get a different bibliography style is to write a LaTeX package which loads the `amsrefs` package with `\RequirePackage` and then makes the desired changes by using suitable `\BibSpec` commands as explained below. Thus, the general form of the custom package will be

```
\ProvidesPackage{xyzbib}[2002/11/06 v1.28]

\RequirePackage{amsrefs}\relax

\BibSpec{article}{
  ...
}

\BibSpec{book}{
  ...
}
```

The interior formatting within entries is specified by `\BibSpec` commands, one for each entry type. To illustrate, let's look at an example style specification for entries of type `article`:

```
\BibSpec{article}{%
  +{}{\PrintAuthors}  {author}
  +{,}{ \textit}      {title}
  +{,}{ }             {journal}
  +{}{ \textbf}       {volume}
  +{}{ \parenthesize} {date}
  +{,}{ }             {pages}
  +{,}{ }             {note}
  +{.}{}              {transition}
  +{}{ }              {review}
}
```

It should be pretty obvious that each line specifies the formatting for a particular field. After reading the data for a particular `\bib` command, LaTeX steps through the style specification and for each field listed, prints the field with the given formatting *if and only if the field has a nonempty value.* The `+` character at the beginning of each field specification must be followed by three arguments: the punctuation to be added if the field is nonempty; space and/or other material to be added after the punctuation; and the field name. It is permissible for the second part to end with a command that takes an argument, such as `\textbf`, in which case it will receive the field's value as its argument. By defining a suitable command and using it here you can place material after the field contents as well as before; `\parenthesize` is an example of this.

The reason that the punctuation and the following space are specified separately is that between them there is a crucial boundary for line breaks. If you put a `\linebreak` command at the end of a field value, the break point will

actually be carried onward to a suitable point after the next bit of punctuation (whose actual value may vary depending on which of the following fields is the first to turn up with a nonempty value).

The meaning of the `\parenthesize` command, supplied by `amsrefs`, should be obvious. The meaning of the `\PrintAuthors` command is a different story. But I don't think it is all that hard to understand. If we have two or more author names which were given separately, and we need to combine them into a conventional name list using commas and the word "and", then it would be nice if we had a command which could take a list of names and Do The Right Thing. And that is just what `\PrintAuthors` is.

The `rkeyval` package allows keys to be defined as additive: if the key occurs more than once, each successive value will be concatenated to the previous value, along with a prefix. The setup done by `amsrefs` for the `author` field is

```
\DefineAdditiveKey{bib}{author}{\name}
```

This means that if two names are given, as in

```
author={Bertram, A.},
author={Wentworth, R.},
```

then the final value of the `author` field seen when LaTeX processes the style specification will be

```
\name{Bertram, A.}\name{Wentworth, R.}
```

The `transition` field in our `\BibSpec` example is a dummy field to be used when punctuation or other material must be added at a certain point in the bibliography without regard to the emptiness or non-emptiness of the fields after it. The `transition` field always tests as non-empty but has no printed content. So when you use it you always get the indicated punctuation and space at the indicated point in the list of fields. If it were the last thing in this `\BibSpec` example, it could serve just to put in the final period that is always wanted. But in AMS bibliographies, if a *Mathematical Reviews* reference is given, it is conventionally printed *after* the final period. Using the `transition` field as shown here ensures that the final period will be always printed, even when the `review` field is empty.

5 Miscellaneous commands provided by the `amsrefs` package

Most of the following commands are helper commands for use in `\BibSpec` statements. The others are intended for use in bibliography data.

`\parenthesize` This command adds parentheses around its argument. It is useful in `\BibSpec` statements because there is no special provision for adding material after the field value.

`\bibquotes` This command is much like `\parenthesize` but it adds quotes around its argument and it has one other important difference: there are special arrangements to print the closing quote *after* a following comma or similar punctuation (unless the `amsrefs` package is invoked with the

logical-quotes option, in which case \bibquotes puts the closing quote immediately after the quoted material).

\voltext This is used to format volume numbers. By default, it precedes the volume number by "vol."

\issuetext This is used to format issue numbers. By default, it precedes the volume number by "no."

\editiontext This command produces "ed." following an edition number. See \PrintEdition for more information.

\DashPages This command is similar in spirit to \voltext but more complicated in its implementation. It takes one argument which is expected to contain one or more page numbers or a range of page numbers. The argument is printed with a prefix of "p." if it seems to be a single page number, otherwise with a prefix of "pp.".

\tsup, \tsub, \tprime These are for text subscripts and superscripts, with \tprime producing a superscript prime symbol. Unlike the standard \textsuperscript and \textsubscript functions provided by LaTeX, these do not use math mode at all.[1]

\nopunct This command causes following punctuation to be omitted if it is added with the internal function \@addpunct.

\PrintPrimary This is a relatively complicated function that determines the "primary" contributors for an entry and formats them, or replaces them by \sameauthors if appropriate. It should be used when an entry type might have editors or translators instead of authors. It prefers authors over editors and editors over translators and generates a warning if there are no primary contributors.

\PrintAuthors This is used to format the list of authors as the primary contributors for an entry type.

\PrintEditorsA This is similar to \PrintAuthors but adds (ed.) or (eds.) following the editors.

\PrintEditorsB This is similar to \PrintEditorsA but puts parentheses around the entire list of editors. It's used by, for example, the article type to print the editors of a proceedings or collection.

\PrintEditorsC Similar to \PrintEditorsA but precedes the editors by Edited by. It's used when the editors should be treated as subsidiary contributors, rather than the primary contributor.

\PrintTranslatorsA This is similar to \PrintEditorsA but adds (trans.) following the translators.

\PrintTranslatorsB This is similar to \PrintEditorsB. It's not currently used, but is provided for symmetry.

\PrintTranslatorsC Similar to \PrintEditorsC but precedes the translators by Translated by.

[1] There is one drawback: If you don't want to get the prime symbol for \tprime from the cmsy font, you will need to redefine \tprime in some suitable way.

\sameauthors This is a function of one argument. If you use the default set of \BibSpecs from the amsrefs, \sameauthors is applied to the author name for a given \bib command if it matches exactly the author name of the preceding \bib command. Change the definition of \sameauthors if you don't want to get a bysame dash.

\bysame This is a horizontal rule of length 3 em. The default definition of \sameauthors prints \bysame instead of the author names.

\Plural, \SingularPlural These are helper functions that allow you to conditionally print singular or plural forms such as (ed.) or (eds.) depending on the number of names in the current name list. The definition of \PrintEditorsA reads, in part,

```
... (ed\Plural{s}.) ...
```

\PrintReviews This is similar to \AuthorList but is used for printing (possibly multiple) MR numbers given in the review field.

\BibField This is for more complicated programming tasks such as may be necessary for some \BibSpecs. It takes one argument, a field name, and yields the contents of that field for the current \bib entry.

\IfEmptyBibField If one writes

```
\IfEmptyBibField{isbn}{A}{B}
```

then the commands in A will be executed if the isbn field is empty, otherwise the commands in B.

\PrintEdition If a bibliography entry has

```
edition={2}
```

and the \BibSpec used \PrintEdition to handle this field, then the edition information will be printed as "2nd ed."—that is, the number is converted to cardinal form and "ed." is added (taken from \editiontext).

\CardinalNumeric This provides the conversion to cardinal number form used by \PrintEdition.

\PrintDate, \PrintYear These functions convert a date in canonical form (ISO 8601) to the form required by the current bibliography style. You can get your preferred date form by redefining these functions or by changing your \BibSpec statements to use another function of your own devising. The original definition of \PrintDate adds parentheses (as for the year of a journal article in normal AMS style), whereas the \PrintYear function simply prints the year without any additional material (as for a book's year of publication in normal AMS style).

\mdash, \ndash These are short forms for \textemdash and \textendash, recommended instead of the more usual --- and -- notation. From the textcmds package.

et cetera ... [mjd,2002-01-03] See the .dtx files for further possibilities that I have not managed to get properly documented yet!

6 Implementation

6.1 Overview

It will be a while yet before we get to any actual code. First we need to understand what the code needs to accomplish in order to provide the user interface described above in a way that is as compatible as possible with existing LaTeX mechanisms.

6.1.1 Normal LaTeX processing of cites

First LaTeX pass Various commands are written to the `.aux` file that are mostly used by BibTeX.

1. A `\cite{moo}` command writes one line to the `.aux` file: `\citation{moo}`. This indicates to BibTeX that it should include 'moo' in the list of cited items to be searched for. The `\cite` command also checks to see if `\b@moo` contains the corresponding citation label, but since this is the first pass, the label won't be known yet, so LaTeX emits an 'Undefined citation' warning and prints a placeholder (i.e., **???**) instead of the citation label.
2. A `\bibliographystyle{har}` command writes one line to the `.aux` file: `\bibstyle{har}`. This indicates to BibTeX that it should use `har.bst` to determine the style for sorting and formatting the bibliography items.
3. A `\bibliography{hij,klm,...}` command writes one line to the `.aux` file: `\bibdata{hij,klm,...}`. This indicates to BibTeX that it should look in `hij.bib`, `klm.bib`, ... for bibliographic data. The `\bibliography` also tries to input the `.bbl` file, but on the first pass it won't exist yet.

On the first pass all `\cite`'s normally are reported as undefined because the `.bbl` file has not yet been created.

BibTeX pass For a document named `xyz.tex`, the command `bibtex xyz` is used to invoke BibTeX. It looks in `xyz.aux` to find the citation information written there by LaTeX. For each `\citation` line, BibTeX searches for a corresponding entry in the specified `.bib` files and formats it. The entire list is then sorted in whatever way dictated by the bibliography style, and written out to the file `xyz.bbl`. This normally produces entries that look something like:

```
\bibitem{BGL} P. Busch, M. Grabowski and P. J. Lahti:
{\it Operational Quantum Physics.}
Springer Verlag, New York (1995).
```

Second LaTeX pass Now the `.bbl` file exists and contains some `\bibitem` commands. At `\begin{document}`, LaTeX reads the `.aux` file, hoping to find some `\bibcite` commands, but it will not find them until the next time around. `\citation`, `\bibstyle`, and `\bibdata` commands in the `.aux` file are simply ignored by LaTeX. Then LaTeX proceeds to typeset the body of the document.

1. Instances of `\cite` still print question marks.
2. The `\bibliography` command causes LaTeX to input `xyz.bbl` and typeset its contents.

3. A `\bibitem{moo}` command writes one line to the `.aux` file: `\bibcite {moo}{9}`, where 9 is the current item number.
4. A `\bibitem[Moody]{moo}` command writes one line to the `.aux` file: `\bibcite{moo}{Moody}`, using the supplied label instead of a number.

Third LaTeX pass Now the `.aux` file contains some `\bibcite` commands. Once again, LaTeX reads the `.aux` file when it reaches `\begin{document}`.

1. A `\bibcite{moo}{Moody}` causes LaTeX to define `\b@moo` with 'Moody' as the replacement text.
2. If two `\bibcite` commands have the same citation key, LaTeX gives a warning message. This happens at `\begin{document}`, during the reading of the `.aux` file.
3. Instances of `\cite` in the body of the document will print the appropriate labels obtained from the `.aux` file.
4. If there are any `\cite` commands for which the `.aux` file did not have a `\bibcite` command, LaTeX will give an 'Undefined citation' warning. This often happens if the `.aux` file is incomplete due to a TeX error on the preceding pass.

6.2 How cites are processed by `amsrefs`

In order to support its additional features (e.g., author-year citations and the `backrefs` option), the `amsrefs` package stores additional information for each cite in the macro `\b@whatever`. Instead of simply using the defined or undefined status of this macro to trigger the standard warnings, we add some boolean flags to allow us to discriminate more finely what the current situation is.

- Each time an item is cited in the body of the document, a `backref` entry is added to the info of that item. The `backref` info is the current page and section location. Section location is a bit hard to get right without better support from the document class. So we provide a hook to allow it to work better when the support is there.
- When a cite occurs, if the info is undefined then a warning is issued and the info structure is created. A `\citation` command and a `\citedest` command (providing backref info) are written to the `.aux` file. Because the backref info includes page number, it has to be a non-immediate write. An undefined info structure would normally happen only on a first pass when no `.aux` file exists, or when a new cite is added. I.e., when the corresponding `\citation` command is not yet present in the `.aux` file.
- When a citation command occurs in the `.aux` file, it initializes the info structure if necessary, setting the "bib-info-present" flag to 0.
- When a `\citedest` command occurs in the `.aux` file, it initializes the info structure if necessary—but this shouldn't happen: if the corresponding `\citation` command did not already get processed, then something is wrong. So normally, the `\citedest` command merely needs to add its backref info to the existing info structure.

- When a `\bibcite` command occurs in the `.aux` file, it will normally find that `\b@whatever` is already defined, if the bibliography occurs after all the `\cite` commands. What it must do is fill in the appropriate blank slots in the info structure set up by a previous `\citation` command.
- The `.aux` file is actually processed two times, once at the beginning of the document and once at the end. In the latter case, `\bibcite` should give a warning if the backref-list is empty, since that means there were no `\cite` commands for the given key.
- When processing the bibliography: The `\bib` command needs to check if it is using a key that is already used by another `\bib` command.

We therefore have

```
\b@xyz -> \citesel 00{label}{year}{backref-list}
```

where the first 0 is replaced by 1 if there has already been another citation for the same key earlier in the document (some citation styles use abbreviated forms for all instances after the first), and the second 0 is replaced by 1 if the same key was already used by an earlier `\bib` command.

Because the backref-list often includes page number information, it cannot be built on the fly as we go along; instead we have to write the information to the `.aux` file and read it in at the beginning of the next run.

If there was no `\bibcite` in the `.aux` file for a given key, then the info is

```
\b@xyz -> \citesel 00{}{}{backref-list}
```

If there was neither `\citation` nor `\bibcite` in the `.aux` file for a given key, then the `\cite` command should find that `\b@xyz` is undefined.

If the author-year option is in effect, the "label" contains the author last names instead of a label:

```
\b@xyz -> \citesel 00{\name{Smith}\name{Jones}}{...}{...}
```

Full name information is included in the data because some citation styles give full names at the first citation and abbreviated forms for subsequent instances.

6.3 Data structures

The result of scanning the key/value pairs of a `\bib` command is an assignment statement for `\rsk@toks`. (Cf. the `rkeyval` package.) For example, consider the entry

```
\bib{miller83}{article}{
  author={Miller, G.},
  title={Eine Bemerkung zur Darstellung von Polynomen \"{u}ber
    Verb\"{a}nden}*{language={german}},
  journal={J. Math. Sent.},
  volume={10},
  year={1983},
  pages={26\ndash 30},
}
```

The scanned result is to assign

```
\global\rsk@toks{%
```

```
    \set:bib'author{Miller, G.}{}%
    \set:bib'title{Eine Bemerkung zur Darstellung von Polynomen
        \"{u}ber Verb\"{a}nden}{language={german}}%
    \set:bib'journal{J. Math. Sent.}{}%
    \set:bib'volume{10}{}%
    \set:bib'year{1983}{}%
    \set:bib'pages{26\ndash 30}{}%
}
```

The code in the last arg of `\RestrictedSetKeys` then invokes `\bib@exec` to do something with the value of `\rsk@toks`.

```
\bib@exec{miller83}{\the\rsk@toks}{\setbib@article}{}
```

6.4 Preliminaries

1 ⟨∗pkg⟩

Standard declaration of package name and date.

```
2 \NeedsTeXFormat{LaTeX2e}[1995/12/01]
3 \ProvidesPackage{amsrefs}[2013/01/16 v2.12]
```

`\amsrefs@warning@nl`

```
4 \def\amsrefs@warning@nl{\PackageWarningNoLine{amsrefs}}
```

Backward handling for beta and jpa options.

```
5 \@ifpackagewith{amsrefs}{beta}{%
6     \amsrefs@warning@nl{The beta option is obsolete}%
7 }{}
8 \@ifpackagewith{amsrefs}{jpa}{%
9     \amsrefs@warning@nl{The jpa option is obsolete}%
10 }{}
11 \IfFileExists{url.sty}{%
12     \RequirePackage{url}\relax
13     \@gobble
14 }{%
15     \@firstofone
16 }
17 {
18     \DeclareRobustCommand{\url}[1]{%
19         \def\@tempa{#1}%
20         \texttt{\@urlsetup $\expandafter\strip@prefix\meaning\@tempa$}%
21     }%
22     \def\@urlsetup{%
23         \check@mathfonts \textfont\@ne\the\font \textfont\z@\the\font
24         \@apply\@urlfix{\do\+\do\=\do\:\do\-\do\.\do\,\do\;}%
25         \@apply\@urlbreak{\do\&\do\/\do\?}%
26     }%
27     \def\@urlbreak#1{%
28         \mathcode`#1="8000
29         \begingroup \lccode`\~=`#1 \lowercase{\endgroup \edef~}%
30         {\mathchar\number`#1\penalty\hyphenpenalty}%
31     }%
```

```
    \def\@urlfix#1{%
        \mathcode`#1=`#1\relax
    }%
}
\@ifundefined{NormalCatcodes}{\RequirePackage{pcatcode}\relax}{}
\PushCatcodes\NormalCatcodes
%% WARNING WARNING WARNING: Catcode of apostrophe ' is letter
%% throughout this file.
\catcode`\'=11 % letter
```

6.5 Utilities

Some of these useful functions are also found in AMS document classes.

`\after@deleting@token` Similar in concept to `\afterassignment`, except it deletes the next token in the stream before putting its argument back into the input. Useful for skipping past tokens during parsing.

```
\def\after@deleting@token#1{%
    \afterassignment#1%
    \let\@let@token= % Don't delete this space!
}
```

`\@ifempty` `\@ifnotempty` Some frequently used tests for empty arguments. Note that an argument consisting entirely of spaces (e.g., `\@ifempty{␣}`) counts as empty.

```
\long\def\@ifempty#1{\@xifempty#1@@..\@nil}

\long\def\@xifempty#1#2@#3#4#5\@nil{%
    \ifx#3#4\@xp\@firstoftwo\else\@xp\@secondoftwo\fi
}

\long\def\@ifnotempty#1{\@ifempty{#1}{}}
```

`\macrotext`

```
\def\macrotext{\expandafter\strip@prefix\meaning}
```

`\vdef` "Verbatim" def.

```
\def\vdef#1#2{%
    \def#1{#2}%
    \edef#1{\macrotext#1}%
}
```

`\auto@protect` Sometimes it's convenient to render a given control sequence unexpandable for a time. `\auto@protect` provides a way to do that.[2]

An earlier version of this code read simply `\let#1\relax` but that had the disadvantage of making all `\auto@protect`ed macros compare equal via `\ifx`. This version allows macros to keep their identities under comparisons.

```
\def\auto@protect#1{\def#1{\@nx#1}}
```

[2] There really should be a special name for macros that, like `\auto@protect`, take a control sequence as an argument and redefine that control sequence in order to achieve some special effect. Pending happier inspiration, I'm going to call them "wrapper" macros.

`\g@undef` Globally undefine a control sequence.

```
\def\g@undef#1{\global\let#1\relax}
```

`\@concat` Concatenate onto the end of a token list. Expands everything.

```
\def\@concat#1#2{\edef#1{#1#2}}
```

`\add@toks@` This saves a few tokens of main memory and a lot of typing.

```
\def\add@toks@{\addto@hook\toks@}
```

`\@lappend` Append an element to a `\do`-delimited list. As long as the element to be appended (`#2`) is a single token, nothing is expanded. If it contains multiple tokens, all tokens after the first will be expanded.

```
\def\@lappend#1#2{%
    \begingroup
        \def\do{\@nx\do\@nx}%
        \edef\@tempa{\def\@nx#1{#1\do#2}}%
    \@xp\endgroup
    \@tempa
}
```

`\@apply` Apply a macro to each element of a `\do`-delimited list.

```
\def\@apply#1#2{%
    \let\do#1%
    #2%
}
```

`\get@numberof` This is a generic macro for counting the number of elements in a LaTeX-style list. The first argument is a `\count` register that will receive the final count; the second argument is the control sequence that separates elements of the list, and the third argument is the list itself. So, for example,

```
\get@numberof\@tempcnta\do\dospecials
```

would count the number of special characters in `\dospecials` and store the number in `\@tempcnta`.

```
\def\get@numberof#1#2#3{%
    \begingroup
        \def#2{\advance\@tempcnta\@ne \@gobble}%
        \@tempcnta\z@
        #3\relax
        \edef\@tempb{#1=\the\@tempcnta\relax}%
    \@xp\endgroup
    \@tempb
}
```

`\safe@set` This is a quick and dirty way of extracting an integer prefix from a string and assigning it to a counter. If the string does not begin with an integer, the counter receives the value 0. The suffix after the integer prefix is discarded. (But bad things will happen if the string contains the token `\@nil`.)

```
\def\safe@set#1#2{%
    \afterassignment\@nilgobble
    #1=0#2\relax\@nil
}
```

`\@chomp` Vaguely reminiscent of Perl's `chomp` function, which removes a substring from the end of a variable, but ours works with tokens (more-or-less) and takes the substring to be removed as its second argument. Note the use of `\@empty` to anchor the chomped substring to the end of the string. Note also that the second argument will be fully expanded during the chomping.

```
\def\@chomp#1#2{%
    \begingroup
        \toks@\@emptytoks
        \def\@chomper##1##2#2\@empty##3\@nil{%
            \ifx\@let@token\bgroup
                \toks@{{##1}##2}%
            \else
                \toks@{##1##2}%
            \fi
        }%
        \@xp\chomp@ #1\@empty#2\@empty\@nil
        \edef\@tempa{\def\@nx#1\@xp{\the\toks@}}%
    \@xp\endgroup
    \@tempa
}
```

`\chomp@` Before passing control to `\@chomper`, we peek ahead at the next token in the stream. That way, if the next token is an open brace, we know we need to surround `\@chomper`'s first argument with braces. Unfortunately, this might still remove braces from the second argument, but I think that's ok for our purposes.

```
\def\chomp@{%
    \futurelet\@let@token
    \@chomper
}
```

`\amsrefs@warning`

```
\def\amsrefs@warning{\PackageWarning{amsrefs}}
```

`\amsrefs@error`

```
\def\amsrefs@error{\PackageError{amsrefs}}
```

`\MessageBreakNS` This suppresses the leading space in `\on@line` in error and warning messages.

```
\def\MessageBreakNS{\MessageBreak\romannumeral`\^^@}
```

`\@addpunct` The `\@addpunct` function is defined by AMS document classes and the `amsgen` package. But if we find it undefined we had better define it.

```
\@ifundefined{@addpunct}{%
```

```
    \def\@addpunct#1{%
        \relax\ifhmode
            \ifnum\spacefactor>\@m \else#1\fi
        \fi
    }
    \def\frenchspacing{%
        \sfcode`\.1006
        \sfcode`\?1005
        \sfcode`\!1004
        \sfcode`\:1003
        \sfcode`\;1002
        \sfcode`\,1001\relax
    }
}{}
```

`\nopunct` Omit any following punctuation that would normally be inserted by `\@addpunct`.

```
\providecommand{\nopunct}{\spacefactor \@nopunctsfcode}
```

`\@nopunctsfcode`

```
\def\@nopunctsfcode{1007 }
```

6.6 Declaring package options

We call the `ifoption` package to facilitate some option tests.

```
\RequirePackage{ifoption}[2000/02/15]
```

The `sorted` option is a no-op and is no longer documented. I'm only leaving it here for backwards compatibility.

```
\DeclareExclusiveOptions{sorted,citation-order}
```

The `alphabetic` option corresponds to the standard `alpha` biblio style with labels like Knu66 (three letters from name plus two digits of year). Maybe should provide an alias LllYY for this option. Numeric is the default since it is commoner in AMS publications.

```
\DeclareExclusiveOptions{alphabetic,shortalphabetic,author-year,numeric}
```

`y2k`

```
\DeclareBooleanOption{y2k}
```

`nobysame`

```
\DeclareBooleanOption{nobysame}
```

The standard `abbrv` bibliography style uses abbreviations for month names and journal names, and first names of people are abbreviated to their initials. Since the second test bibliography that I tested with had unabbreviated month names but abbreviated journal names, perhaps it is a good idea to let these choices be specified separately.

```
\DeclareBooleanOption{short-journals}
\DeclareBooleanOption{short-publishers}
```

The `short-journals` and `short-publishers` options only affect journal and publisher names that are defined with `\DefineJournal` and `\DefinePublisher` commands.

```
\DeclareBooleanOption{short-months}
\DeclareBooleanOption{initials}
```

Nevertheless, it's to be expected that the preceding four options would typically be used together, so we provide a short-hand for requesting them all.

```
\DeclareOption{abbrev}{%
    \@pass@ptions
        \@currext
        {initials,short-months,short-journals,short-publishers}%
        \@currname
}
```

In the bibliography, if a title or something is enclosed in quotes, should the closing quotes go inside the punctuation (logical position) rather than outside (traditional)? These options give you a choice.

```
\DeclareExclusiveOptions{traditional-quotes,logical-quotes}
```

A sequence of cites will be sorted and ranges of length three or greater will be compressed if these options so indicate. Note that the `non-sorted-cites` option automatically disables compression. This is probably a feature.

```
\DeclareExclusiveOptions{sorted-cites,non-sorted-cites}
\DeclareExclusiveOptions{non-compressed-cites,compressed-cites}
```

In the bibliography, print page numbers showing where each entry was cited.

```
\DeclareBooleanOption{backrefs}
```

Option for giving information about the available options:

```
\DeclareBooleanOption{?}
```

This option means to forgo loading of the `textcmds` and `mathscinet` packages.

```
\DeclareBooleanOption{lite}
```

This option can be used by later releases as a sign that fall-back adaptations need to be done.

```
\DeclareBooleanOption{beta}
\DeclareBooleanOption{bibtex-style}
\DeclareBooleanOption{msc-links}
\ExecuteOptions{numeric,traditional-quotes,sorted-cites,compressed-cites}

\ProcessOptions\relax

\ProcessExclusiveOptions
\IfOption{backrefs}{%
    \IfFileExists{backref.sty}{%
        \RequirePackage{backref}[1999/05/30]
```

```
    }{%
        \amsrefs@warning@nl{The backrefs option cannot be used^^J%
            unless the backref package is also installed.^^J%
            (backref is part of the hyperref package)}%
    }%
}{}

\IfOption{msc-links}{%
    \IfFileExists{hyperref.sty}{%
        \RequirePackage{hyperref}[1999/07/08]
    }{
        \amsrefs@warning@nl{The msc-links option cannot be used^^J%
            unless the hyperref package is installed}%
    }%
}{}
```

6.6.1 The `?` option

```
\IfOption{?}{%
    \typeout{^^J%
        Documentation for the amsrefs package is found in amsrdoc.dvi^^J%
        (or .pdf or .tex).
        ^^J%
    }%
}{}%
```

6.7 Loading auxiliary packages

Now, if these other packages make use of the `pcatcode` package like they should, then we don't need to make any fuss here about the special catcode of `'`. Just load the packages.

```
\RequirePackage{rkeyval}[2001/12/22]
```

6.7.1 The `lite` option

In my opinion, this is misguided, since `amsrefs` shouldn't be loading these packages to begin with. But it's too late to change it now.

```
\IfOption{lite}{% True? Then don't load the next two packages.
}{%             False? OK, let's load them:
    \RequirePackage{textcmds}[2001/12/14]
    \RequirePackage{mathscinet}[2002/01/01]
}
```

6.8 Key-value setup

`\BibField` This provides easy access to individual fields for user-defined formatting functions.

```
\newcommand{\BibField}[1]{\csname bib'#1\endcsname}
```

`\IfEmptyBibField` A convenient partial application of `\rkvIfEmpty`.

```
\newcommand{\IfEmptyBibField}{\rkvIfEmpty{bib}}
```

6.8.1 Standard field names (the bib group)

And here are the predefined key names. You could always add some more if you needed them. Only worry is about compatibility if you want to share your data with other people.

\fld@elt
\name

We want the list macros used above to be unexpandable except when special processing is done. (It's not clear to me there's any real benefit to using these instead of just using \do.—dmj)

```
\let\fld@elt=?
\let\name=?
```

First the fields that could be repeated more than once in a single entry. Maybe publisher should be allowed to repeat also, for co-published works. But then need to worry about the address handling.

```
\DefineAdditiveKey{bib}{author}{\name}
\DefineAdditiveKey{bib}{editor}{\name}
\DefineAdditiveKey{bib}{translator}{\name}
\DefineAdditiveKey{bib}{contribution}{\fld@elt}
\DefineAdditiveKey{bib}{isbn}{\fld@elt}
\DefineAdditiveKey{bib}{issn}{\fld@elt}
\DefineAdditiveKey{bib}{review}{\fld@elt}
\DefineAdditiveKey{bib}{partial}{\fld@elt}

\DefineSimpleKey{bib}{accessdate}
\DefineSimpleKey{bib}{address}
\DefineSimpleKey{bib}{book}
\DefineSimpleKey{bib}{booktitle}
\DefineSimpleKey{bib}{conference}
%\DefineSimpleKey{bib}{contributor}
\DefineSimpleKey{bib}{copula}
\DefineSimpleKey{bib}{date}
\DefineSimpleKey{bib}{doi}
\DefineSimpleKey{bib}{edition}
\DefineSimpleKey{bib}{eprint}
\DefineSimpleKey{bib}{fulljournal}
\DefineSimpleKey{bib}{hyphenation}
\DefineSimpleKey{bib}{institution}
\DefineSimpleKey{bib}{journal}
\DefineSimpleKey{bib}{label}
\DefineSimpleKey{bib}{language}
\DefineSimpleKey{bib}{name}
\DefineSimpleKey{bib}{note}
\DefineSimpleKey{bib}{number}
\DefineSimpleKey{bib}{organization}
\DefineSimpleKey{bib}{pages}
\DefineSimpleKey{bib}{part}
\DefineSimpleKey{bib}{place}
\DefineSimpleKey{bib}{publisher}
\DefineSimpleKey{bib}{reprint}
```

```
\DefineSimpleKey{bib}{school}
\DefineSimpleKey{bib}{series}
\DefineSimpleKey{bib}{setup}
\DefineSimpleKey{bib}{status}
\DefineSimpleKey{bib}{subtitle}
\DefineSimpleKey{bib}{title}
\DefineSimpleKey{bib}{translation}
\DefineSimpleKey{bib}{type}
\DefineSimpleKey{bib}{url}
\DefineSimpleKey{bib}{volume}
\DefineSimpleKey{bib}{xref}
\DefineSimpleKey{bib}{year}
```

The `transition` key is used when we want to insert punctuation or other material at a given point in the sequence unconditionally. The key appears to have a non-empty value to `\IfEmptyBibField`, but its value (expansion) is empty.

```
\DefineDummyKey{bib}{transition}
```

6.8.2 Auxiliary properties (the `prop` group)

```
\DefineSimpleKey{prop}{inverted}
\DefineSimpleKey{prop}{language}
```

6.9 Bibliography type specifications

`\BibSpec` Accumulate specification material in `\toks@`, then define `\setbib@TYPE` from it.

```
\newcommand{\BibSpec}[2]{%
    \toks@\@emptytoks
    \@ifnotempty{#2}{%
```

The `\@ifnextchar` removes an optional `+` at the beginning of a specification. From then on, each time `\bibspec@scan` is invoked, it expects to find four arguments. The four `\@empty`s appended to the specification (`#2`) below ensure that this is so.

```
        \@ifnextchar{+}{\@xp\bibspec@scan\@gobble}{\bibspec@scan}%
        #2\@empty\@empty\@empty\@empty
    }%
    \@xp\edef\csname setbib@#1\endcsname{\the\toks@}%
}
```

`\bibspec@scan` The `\bibspec@scan` function scans one field specification from the second arg of `\BibSpec`. Each field specification has the form

```
+{punctuation}{prelim material}{field name}
```

Note however that because the initial `+` is stripped off by `\BibSpec` (see above), the actual order that `\bibspec@scan` reads the field specification is

```
#1={punctuation} #2={prelim material} #3={field name} #4=+
```

where the fourth argument is actually expected to be either the `+` from the following specification, or one of the special `\@empty` tokens inserted by `\BibSpec`.

If it is neither of these special values, it means we have a malformed specification; so, we issue an error and then try to pick up where we left off.

```
\def\bibspec@scan#1#2#3#4{%
    \add@toks@{\bib@append{#1}{#2}}%
    \edef\@tempa{%
        \toks@{\the\toks@ \@xp\@nx\csname bib'#3\endcsname}%
    }%
    \@tempa
    \ifx\@empty#4%
        \@xp\@gobble % end the recursion
    \else
        \ifx +#4\else\bibspec@scan@error\fi
    \fi
    \bibspec@scan
}
```

\bibspec@scan@error

```
\def\bibspec@scan@error{\amsrefs@error{Bad BibSpec: Expected '+'}}
```

\bib@append The function \bib@append prints the value of a field, together with associated punctuation and font changes, unless the value is empty. Arg 1 is punctuation (that may need to be swapped with a preceding line break), arg 2 gives the space to be added after the punctuation, and possibly a function to be applied to the contents of arg 3, which is a macro containing the field value. So if we have \moo and \bib'pages, from pages={21\ndash 44}, then we want to arrange to call

```
\moo{21\ndash 44}
```

We don't want to simply call \moo\bib'bar because that makes it rather difficult for \moo to look at the contents of \bib@bar.

```
\def\bib@append#1#2#3{%
    \ifx\@empty#3%
    \else
```

Known bug: Need better error message here.

```
        \ifx\relax#3%
            \errmessage{#3=\relax}%
        \else
            \begingroup
                \series@index\m@ne
                \def\current@bibfield{#3}%
                \@ifempty{#1}{%
                    \@temptokena{\ifnum\lastkern=\@ne\ignorespaces\fi #2}%
                }{%
                    \@temptokena{\SwapBreak{#1}#2}%
                }%
                \toks@\@xp{#3}%
                \edef\@tempa{\the\@temptokena{\the\toks@}}%
```

```
                \rkvIfAdditive#3{}{%
                    \get@current@properties
                    \select@auxlanguage
                }%
                \@tempa
            \endgroup
        \fi
    \fi
}
```

`\select@auxlanguage`

```
\def\select@auxlanguage{%
    \ifx\prop'language\@empty
    \else
        \@xp\selectlanguage\@xp{\prop'language}%
    \fi
}
```

`\erase@field` There are some fields that can appear in more than one place in a reference, depending on context. For example, if a book has an editor but no author, the editor appears at the beginning of the entry, but if the book has both an editor and an author, the editor appears at the end of the entry. A simple way to handle this is to "erase" the `editor` field after printing it, which is what `\erase@field` is for.

The obvious definition of `\erase@field` is

```
\def\erase@field#1{\global\let#1\@empty}
```

but that doesn't work because the top-level value of `rkeyval` fields isn't `\@empty`; instead, it contains a setter function used by `\RestrictedSetKeys` when processing a key-value list (see `\rkv@DSAK`, `\rsk@set@a` and `\rsk@set@b`).

On the other hand, rewriting the field locally won't work either, since `\erase@field` will typically be executed inside the group established by `\bib@append`. Instead, we want to rewrite the value right after `\bib@append`'s group ends. One way to do this would be to keep a list of fields to be erased and have `\bib@append` iterate over the list after its `\endgroup`.

However, as long as the call to `\erase@field` is never nested within any deeper groups, it's simpler just to use `\aftergroup`, which is what we'll do ("Sufficient unto the day is the evil thereof" and all that).

```
\def\erase@field#1{%
    \aftergroup\let\aftergroup#1\aftergroup\@empty
}
```

`\get@current@properties` This retrieves the auxiliary properties for the current field value, as defined by `\current@bibfield` and `\series@index`.

```
\def\get@current@properties{%
    \begingroup
        \@xp\get@nth@property\@xp\@tempa\current@bibfield\series@index
```

```
        \edef\@tempa{%
            \@nx\RestrictedSetKeys{}{prop}{%
                \def\@nx\@tempa{\@nx\prop@reset \@nx\the\@nx\rsk@toks}%
            }{\@tempa}%
        }%
        \@tempa
    \@xp\endgroup
    \@tempa
}
```

\BibSpecAlias This is a \def rather than a \let because using \let would make \BibSpecAlias statements order-sensitive in a way that seems frequently to be a stumbling block to unwary package writers. But then we should probably do at least the simplest kind of infinite loop check.

```
\newcommand{\BibSpecAlias}[2]{%
    \@xp\def\@xp\@tempa\@xp{\csname setbib@#1\@xp\endcsname}%
    \@xp\ifx\csname setbib@#2\endcsname\@tempa
        \amsrefs@error{%
            Mirror alias #1->#2 not allowed (infinite loop)}\@ehc
    \else
        \@xp\def\csname setbib@#1\@xp\endcsname
            \@xp{\csname setbib@#2\endcsname}%
    \fi
}
```

6.10 The standard bibliography types

```
\BibSpec{article}{%
    +{}  {\PrintAuthors}                {author}
    +{,} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintContributions}         {contribution}
    +{.} { \PrintPartials}              {partial}
    +{,} { }                            {journal}
    +{}  { \textbf}                     {volume}
```

The date form is tricky depending on presence or absence of DOI.

```
    +{}  { \PrintDatePV}                {date}
    +{,} { \issuetext}                  {number}
    +{,} { \eprintpages}                {pages}
    +{,} { }                            {status}
    +{,} { \PrintDOI}                   {doi}
    +{,} { available at \eprint}        {eprint}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}
```

```
337
338 \BibSpec{partial}{%
339     +{}  {}                                {part}
340     +{:} { \textit}                        {subtitle}
341     +{,} { \PrintContributions}            {contribution}
342     +{,} { }                               {journal}
343     +{}  { \textbf}                        {volume}
344     +{}  { \PrintDatePV}                   {date}
345     +{,} { \issuetext}                     {number}
346     +{,} { \eprintpages}                   {pages}
347 }
348
349 \BibSpec{contribution}{%
350     +{} {}                                {type}
351     +{} { by \PrintNameList}              {author}
352 }
353
354 \BibSpec{book}{%
355     +{}  {\PrintPrimary}                   {transition}
356     +{,} { \textit}                        {title}
357     +{.} { }                               {part}
358     +{:} { \textit}                        {subtitle}
359     +{,} { \PrintEdition}                  {edition}
360     +{}  { \PrintEditorsB}                 {editor}
361     +{,} { \PrintTranslatorsC}             {translator}
362     +{,} { \PrintContributions}            {contribution}
363     +{,} { }                               {series}
364     +{,} { \voltext}                       {volume}
365     +{,} { }                               {publisher}
366     +{,} { }                               {organization}
367     +{,} { }                               {address}
368     +{,} { \PrintDateB}                    {date}
369     +{,} { }                               {status}
370     +{}  { \parenthesize}                  {language}
371     +{}  { \PrintTranslation}              {translation}
372     +{;} { \PrintReprint}                  {reprint}
373     +{.} { }                               {note}
374     +{.} {}                                {transition}
375     +{}  {\SentenceSpace \PrintReviews}    {review}
376 }
377
378 \BibSpec{collection.article}{%
379     +{}  {\PrintAuthors}                   {author}
380     +{,} { \textit}                        {title}
381     +{.} { }                               {part}
382     +{:} { \textit}                        {subtitle}
383     +{,} { \PrintContributions}            {contribution}
384     +{,} { \PrintConference}               {conference}
385     +{}  {\PrintBook}                      {book}
386     +{,} { }                               {booktitle}
```

```
    +{,} { \PrintDateB}                 {date}
    +{,} { pp.~}                        {pages}
    +{,} { }                            {status}
    +{,} { \PrintDOI}                   {doi}
    +{,} { available at \eprint}        {eprint}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{conference}{%
    +{}  {}                         {title}
    +{}  {\PrintConferenceDetails} {transition}
}

\BibSpec{innerbook}{%
    +{,} { }                            {title}
    +{.} { }                            {part}
    +{:} { }                            {subtitle}
    +{,} { \PrintEdition}               {edition}
    +{}  { \PrintEditorsB}              {editor}
    +{,} { \PrintTranslatorsC}          {translator}
    +{,} { \PrintContributions}         {contribution}
    +{,} { }                            {series}
    +{,} { \voltext}                    {volume}
    +{,} { }                            {publisher}
    +{,} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \PrintDateB}                 {date}
    +{.} { }                            {note}
}

\BibSpec{report}{%
    +{}  {\PrintPrimary}                {transition}
    +{,} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintEdition}               {edition}
    +{,} { \PrintContributions}         {contribution}
    +{,} { Technical Report }           {number}
    +{,} { }                            {series}
    +{,} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \PrintDateB}                 {date}
    +{,} { \eprint}                     {eprint}
    +{,} { }                            {status}
    +{}  { \parenthesize}               {language}
```

```
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{thesis}{%
    +{}  {\PrintAuthors}                {author}
    +{,} { \textit}                     {title}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintThesisType}            {type}
    +{,} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \PrintDateB}                 {date}
    +{,} { \eprint}                     {eprint}
    +{,} { }                            {status}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{webpage}{%
    +{}  {\PrintAuthors}                {author}
    +{,} { \emph}                       {title}
    +{:} { \emph}                       {subtitle}
    +{}  { \PrintDate}                  {date}
    +{,} { \url}                        {url}
    +{.} { Accessed \PrintDateField}    {accessdate}
    +{.} { }                            {note}
    +{.} {}                             {transition}
}
%    \begin{macrocode}
\BibSpecAlias{periodical}{book}
\BibSpecAlias{collection}{book}
\BibSpecAlias{proceedings}{book}
\BibSpecAlias{manual}{book}
\BibSpecAlias{miscellaneous}{book}
\BibSpecAlias{misc}{miscellaneous}
\BibSpecAlias{unpublished}{book}
\BibSpecAlias{proceedings.article}{collection.article}
\BibSpecAlias{techreport}{report}
```

\setbib@incollection

```
\edef\setbib@incollection{%
    \@xp\@nx\csname setbib@collection.article\endcsname
}
```

\setbib@inproceedings

```
\edef\setbib@inproceedings{%
    \@xp\@nx\csname setbib@collection.article\endcsname
}
```

Some more entry types for implementing abbreviations.

```
\BibSpec{name}{%
    +{}  {\PrintAuthors}    {name}
}

\BibSpec{publisher}{%
    +{,} { } {publisher}
    +{,} { } {address}
}
```

6.11 The biblist environment

The biblist environment can be used with a section or chapter heading.

Use a standard LaTeX counter for numbering bibliography items.

```
\newcounter{bib}
```

```
\DefineSimpleKey{biblist}{prefix}
\DefineSimpleKey{biblist}{labels}
```

biblist

```
\newenvironment{biblist}{%
    \setcounter{bib}\z@
    \@biblist
}{%
    \@endbiblist
}
```

biblist*

```
\newenvironment{biblist*}{%
    \@biblist
}{%
    \@endbiblist
}
```

\biblistfont

```
\newcommand{\biblistfont}{%
    \normalfont
    \footnotesize
}
```

\amsrefs@@lbibitem
\amsrefs@bibitem

Reference processing at the AMS sometimes results in raw \bibitem entries being interspersed with \bib entries in a bibliography. For that to work, we need to modify \@lbibitem and \@bibitem to interoperate more smoothly with amsrefs.

```
\def\amsrefs@lbibitem[#1]#2{%
```

```
    \begingroup
        \def\CurrentBib{#2}%
        \def\thebib{#1}%
        \@nmbrlistfalse
        \item\leavevmode
        \if@filesw
            {\let\protect\noexpand
             \immediate\write\@auxout{\string\bibcite{#2}{{#1}{}}}}%
        \fi
    \endgroup
    \ignorespaces
}

\def\amsrefs@bibitem#1{%
    \def\CurrentBib{#1}%
    \item
    \if@filesw
        \immediate\write\@auxout{\string\bibcite{#1}{{\the\value{\@listctr}}{}}}%
    \fi
    \ignorespaces
}
```

\@biblist

```
\newcommand\@biblist[1][]{%
    \stepcounter{bib@env}
    \biblistfont
    \labelsep .5em\relax
    \let\@bibitem\amsrefs@bibitem
    \let\@lbibitem\amsrefs@lbibitem
    \list{\BibLabel}{%
        \restore@labelwidth
        \@maxlabelwidth\z@
        \@nmbrlisttrue
        \def\@listctr{bib}%
        \let\makelabel\bib@mklab
        #1\relax
    }%
    \sloppy
```

Discourage page breaks within bibliography entries and disable them completely for entries that are less than four lines long.

```
    \interlinepenalty\@m
    \clubpenalty\@M
    \widowpenalty\clubpenalty
    \frenchspacing
    \ResetCapSFCodes
    \star@{\@biblistsetup}{}%
}
```

\@biblistsetup

```
\newcommand{\@biblistsetup}[1]{%
    \RestrictedSetKeys{}{biblist}{\the\rsk@toks}{#1}%
    \rkvIfEmpty{biblist}{prefix}{}{%
        \let\amsrefs@label@prefix\biblist'prefix
    }%
    \rkvIfEmpty{biblist}{labels}{}{%
        \@ifundefined{amsrefs@option@\biblist'labels}{%
            \amsrefs@warning{Invalid label style `\biblist'labels`}%
        }{%
            \csname amsrefs@option@\biblist'labels\endcsname
        }%
    }%
}
```

`\@endbiblist` Change error for empty list (no items) to warning, to allow authors to leave their bibliography temporarily empty during writing:

```
\def\@endbiblist{%
    \save@labelwidth
    \def\@noitemerr{\@latex@warning{Empty bibliography list}}%
    \global\let\previous@primary\@empty
    \endlist
}
```

`\@maxlabelwidth`

```
\newdimen\@maxlabelwidth
```

`\bib@mklab`

```
\def\bib@mklab#1{%
    \settowidth\@tempdima{#1}%
    \ifdim \@tempdima > \@maxlabelwidth
        \global\@maxlabelwidth\@tempdima
    \fi
    #1\hfil
}

\newcounter{bib@env}
```

`\save@labelwidth`

```
\def\save@labelwidth{%
    \if@filesw
        \immediate\write\@auxout{%
            \string\newlabel{[bibenv:\the\c@bib@env]}{\the\@maxlabelwidth}%
        }%
    \fi
}
```

`\restore@labelwidth`

```
\def\restore@labelwidth{%
    \@xp\ifx \csname r@[bibenv:\the\c@bib@env]\endcsname \relax
```

```
        \resetbiblist{00}%
    \else
        \@xp\labelwidth\csname r@[bibenv:\the\c@bib@env]\endcsname
        \leftmargin\labelwidth
        \advance\leftmargin\labelsep
    \fi
}
```

\ResetCapSFCodes Presumably this is here because there has been a problem in the past with packages that change the \catcodes of capital letters.

```
\providecommand{\ResetCapSFCodes}{%
    \count@=`\A
    \def\@tempa{%
        \sfcode\count@=\@m
        \advance\count@\@ne
            \ifnum\count@>`\Z\relax \expandafter\@gobble \fi
        \@tempa
    }%
    \@tempa
}
```

\CurrentBib In case this is undefined sometimes.

```
\def\CurrentBib{??}
```

\BibLabel

```
\newcommand{\BibLabel}{%
    \hfill
    \Hy@raisedlink{\hyper@anchorstart{cite.\CurrentBib}\hyper@anchorend}%
    [\thebib]%
}
```

\resetbiblist

```
\newcommand{\resetbiblist}[1]{%
    \settowidth\labelwidth{\def\thebib{#1}\BibLabel}%
    \leftmargin\labelwidth
    \ifdim\labelwidth=\z@
        \leftmargin=1em
        \itemindent=-\leftmargin
    \else
        \advance\leftmargin\labelsep
    \fi
}
```

6.12 Processing bibliography entries

There are several things one might want to do when a \bib entry is encountered:

1. Format and print it. This corresponds to the direct entry of bibliography items as described in section 2.1 of the users's guide.

2. Copy it into a `.bbl` file. This corresponds to the use of `\bibselect` and an external `.ltb` database as described in section 2.2 of the user's guide.
3. Store the full information in memory. This is done by `\bib*`.

`\bib` Here is where the rubber hits the road.

```
\newcommand{\bib}{%
    \begingroup
        \@ifstar{%
            \@tempswatrue
            \let\@bibdef\star@bibdef
            \BibItem
        }{%
            \@tempswafalse
            \BibItem
        }%
}
```

`\BibItem` *Arguments*:
`#1` <- *citekey*.
`#2` <- *bibtype*.

```
\newcommand{\BibItem}[2]{%
    \vdef\@tempa{#1}%
    \edef\@tempa{%
        \edef\@nx\@tempa{\@nx\@xp\@nx\zap@space\@tempa\space\@nx\@empty}%
    }%
    \@tempa
    \edef\@tempb{%
        \@nx\@bibdef\@xp\@nx\csname setbib@#2\endcsname{#2}{\@tempa}%
    }%
    \@tempb
}
```

`\@bibdef` `\@bibdef` is a pointer to the procedure that should be handed the entry's key-value pairs. It has one of four values:

1. `\star@bibdef`
2. `\normal@bibdef`
3. `\copy@bibdef`
4. `\selective@bibdef`

Arguments:
`#1` <- `\setbib@`*bibtype*.
`#2` <- *bibtype*.
`#3` <- *citekey*.

```
\AtBeginDocument{\let\@bibdef\normal@bibdef}
```

`\bib@exec` And `\bib@exec` is a pointer to the procedure that `\normal@bibdef` will invoke to process the key-value pairs after they've been parsed. It has one of these values:

1. `\bib@store`
2. `\bib@print`

Arguments:
`#1 <-` *citekey*.
`#2 <- \the\rsk@toks`.
`#3 <- \setbib@`*bibtype*.

```
651 \AtBeginDocument{\let\bib@exec\bib@print}
```

6.12.1 `\@bibdef` Implementations

`\normal@bibdef` *Arguments*:
`#1 <- \setbib@`*bibtype*.
`#2 <-` *bibtype*.
`#3 <-` *citekey*.

```
652 \def\normal@bibdef#1#2#3{%
```

`\CurrentBibType` is used by `export-bibtex`, but there might be a better way to handle it. (dmj)

```
653     \def\CurrentBibType{#2}%
654     \ifx\relax#1%
655         \amsrefs@error{Undefined entry type: #2}\@ehc
656         \let#1\setbib@misc
657     \fi
658     \RestrictedSetKeys{}{bib}%
659         {\bib@exec{#3}{\the\rsk@toks}{#1}\endgroup}%
660 }
661
662 \let\@bibdef\normal@bibdef
```

`\star@bibdef` *Arguments*:
`#1 <- \setbib@`*bibtype*.
`#2 <-` *bibtype*.
`#3 <-` *citekey*.

```
663 \def\star@bibdef{%
664     \let\bib@exec\bib@store
665     \normal@bibdef
666 }
```

`\copy@bibdef` This is a variation that copies everything into the `.bbl` file. Used by `\bibselect*` and `\bib*` inside `.ltb` files.

```
667 \def\copy@bibdef{%
668     \if@tempswa
669         \@xp\defer@bibdef
```

```
    \else
        \@xp\copy@bibdef@a
    \fi
}
```

\copy@bibdef@a

```
\def\copy@bibdef@a#1#2#3#4{%
    \@open@bbl@file
    \process@xrefs{#4}%
    \bbl@write{%
        \string\bib\if@tempswa*\fi{#3}{#2}\string{\iffalse}\fi
    }%
```

Since we're supplying our own definition of \rsk@set, we don't actually need the group argument, so we leave it out to save a few tokens.

```
    \RestrictedSetKeys{\global\let\rsk@set\bbl@copy}\@empty
        {\bbl@write{\iffalse{\fi\string}^^J}%
         \endgroup}{#4}%
}

\catcode`\:=11

\def\modify@xref@fields{%
    \let\set:bib'author\output@xref@a
    \let\set:bib'editor\output@xref@a
    \let\set:bib'translator\output@xref@a
    \let\set:bib'journal\output@xref@a
    \let\set:bib'publisher\output@xref@a
    \def\set:bib'xref##1##2{\output@xref@{##1}\@empty}%
    \def\set:bib'book##1##2{\output@inner@xref@{##1}\@empty}%
    \let\set:bib'conference\set:bib'book
    \let\set:bib'partial\set:bib'book
    \let\set:bib'reprint\set:bib'book
    \let\set:bib'translation\set:bib'book
}

\catcode`\:=12

\def\process@xrefs#1{%
    \begingroup
        \RestrictedSetKeys{\modify@xref@fields}{bib}{\the\rsk@toks}{#1}%
    \endgroup
}

\def\output@xref@a#1#2{%
    \def\@tempa{#1}%
    \lowercase{\def\@tempb{#1}}%
    \ifx\@tempa\@tempb
        \output@xref@{#1}%
    \fi
```

```
}

\def\output@xref@#1{%
    \@ifnotempty{#1}{%
        \@ifundefined{bi@#1}{}{%
            \begingroup
                \let\star@bibdef\copy@bibdef@a
                \csname bi@#1\endcsname
            \endgroup
        }%
        \@xp\g@undef\csname bi@#1\endcsname
    }%
}

\def\output@inner@xref@#1{%
    \in@={#1}%
    \ifin@\else
        \output@xref@{#1}%
    \fi
}
```

\bbl@copy

```
\def\bbl@copy#1\endcsname#2{%
    \begingroup
        \def\@tempa{#1}%
        \toks@{{#2}}%
        \star@{\bbl@copy@a}{}%
}
```

\bbl@copy@a

```
\def\bbl@copy@a#1{%
        \@ifnotempty{#1}{%
            \add@toks@{*{#1}}%
        }%
        \bbl@write{ \space\@tempa=\the\toks@,}%
    \endgroup
    \rsk@resume
}
```

\selective@bibdef This is a variation that ignores anything not having a known citation key. Used by \bibselect.

Arguments:
#1 <- \setbib@*bibtype*.
#2 <- *bibtype*.
#3 <- *citekey*.

```
\def\selective@bibdef#1#2#3{%
    \@xp\selbibdef@a\csname b@#3\endcsname{#1}{#2}{#3}%
}
```

\selbibdef@a

```
\def\selbibdef@a#1{%
    \def\@tempa{\endgroup\@gobblefour}%
    \ifx\relax#1\else \@xp\selbibdef@b#1\@nil \fi
    \@tempa
}
```

\selbibdef@b

```
\def\selbibdef@b#1#2#3\@nil{%
    \ifx 1#2\let\@tempa\copy@bibdef\fi
}
```

\defer@bibdef This is a variation that ignores anything not having a known citation key. Used by \bibselect.

Arguments:
#1 <- \setbib@*bibtype*.
#2 <- *bibtype*.
#3 <- *citekey*.
#4 <- *key-val pairs*.

```
\def\defer@bibdef#1#2#3#4{%
        \@xp\gdef\csname bi@#3\endcsname{%
            \bib*{#3}{#2}{#4}%
        }%
        \@xp\addto@defer@list \csname bi@#3\endcsname
    \endgroup
}
```

\bibdefer@list

```
\let\bibdefer@list\@empty
```

\addto@defer@list

```
\def\addto@defer@list#1{%
    \begingroup
        \def\do{\@nx\do\@nx}%
        \xdef\bibdefer@list{\bibdefer@list\do#1}%
    \endgroup
}
```

6.12.2 \bib@exec Implementations

\bib@store This is the easy one. It just stores the entire set of key-value pairs in \bi@*citekey*.

```
\def\bib@store#1{%
    \afterassignment\@gobble
    \@xp\xdef\csname bi@#1\endcsname
}
```

\bib@print *Arguments*:
#1 <- *citekey*.

#2 <- \the\rsk@toks.
#3 <- \setbib@*bibtype*.

```
\def\bib@print#1#2#3{%
    \bib@start{#1}%
        \let\setbib@@#3%
        #2\relax       % execute definitions locally
        \bib@resolve@xrefs
        \bib@field@patches
        \bib@selectlanguage
        \generate@label
        \bib'setup
        \bib@cite{#1}%
        \kern\@ne sp
        \ifx\setbib@@\setbib@article
            \ifx\bib'booktitle\@empty
                \ifx\bib'book\@empty
                    \ifx\bib'conference\@empty
                    \else
                        \let\setbib@@\setbib@incollection
                    \fi
                \else
                    \let\setbib@@\setbib@incollection
                \fi
            \else
                \let\setbib@@\setbib@incollection
            \fi
        \fi
        \setbib@@
    \bib@end
}
```

`\bib@print@inner` Note that the order of the arguments is reversed with respect to `\bib@print`. Maybe that isn't such a great idea.

Arguments:
#1 <- \setbib@*bibtype*.
#2 <- \the\rsk@toks.

```
\def\bib@print@inner#1#2{%
    \begingroup
        #2\relax       % execute definitions locally
        \bib@resolve@xrefs
        \bib@field@patches
        \bib'setup
        #1%
    \endgroup
}
```

`\current@citekey`

```
\let\current@citekey\@empty
```

\prev@citekey

```
\let\prev@citekey\@empty
```

\bib@start There used to be more to it.

```
\def\bib@start#1{%
    \begingroup
        \def\current@citekey{#1}%
}
```

\bib@end Instead of being handled by \bib@end, ending punctuation is normally handled via the transition field (q.v.)

```
\def\bib@end{%
        \relax
        \@xp\PrintBackRefs\@xp{\CurrentBib}%
        \par
        \save@primary
        \global\let\prev@citekey\current@citekey
    \endgroup
}
```

6.12.3 Resolving cross-references

\bib@resolve@xrefs

```
\def\bib@resolve@xrefs{%
    \xref@check@c\bib'xref
    \xref@check@a\bib'author
    \xref@check@a\bib'editor
    \xref@check@a\bib'translator
    \xref@check@b\bib'journal
    \xref@check@b\bib'publisher
}
```

\xref@check@a Resolve a contributor (typically a \DefineName) alias. Requires rebuilding the list.

```
\def\xref@check@a#1{%
    \ifx\@empty#1\relax
    \else
        \begingroup
            \toks@\@emptytoks
            \@temptokenb\@emptytoks
            \series@index\z@
            \def\name{\xref@check@aa#1}%
            #1\relax
            \edef\@tempa{%
                \def\@nx#1{\the\toks@}%
                \the\@temptokenb
            }%
        \@xp\endgroup
        \@tempa
```

```
    \fi
}
```

\xref@check@aa

```
\def\xref@check@aa#1#2{%
    \advance\series@index\@ne
    \def\@tempa{#2}%
    \lowercase{\def\@tempb{#2}}%
    \ifx\@tempa\@tempb
        \ifx\@tempa\@empty
            \add@toks@{\name{}}%
        \else
            \@ifundefined{bi@#2}{%
                \BibAbbrevWarning{#2}%
                \add@toks@{\name{#2}}%
            }{%
                \xref@check@ab#1{#2}%
            }%
        \fi
    \else
        \add@toks@{\name{#2}}%
    \fi
}
```

\xref@check@ab

```
\def\xref@check@ab#1#2{%
    \csname bi@#2\endcsname
    \ifx\@empty\bib'name
        \@temptokena{#2}%
    \else
        \@temptokena\@xp{\bib'name}%
        \get@property\@tempa\bib'name
        \edef\@tempa{%
            \@nx\addto@hook\@temptokenb{%
                \@nx\reset@nth@property\@nx#1\the\series@index{\@tempa}%
            }%
        }%
        \@tempa
    \fi
    \edef\@tempa{\@nx\add@toks@{\@nx\name{\the\@temptokena}}}%
    \@tempa
}
```

\xref@check@b Resolve a `journal` or `publisher` alias (typically a `\DefinePublisher` or `\DefineJournal` alias).

```
\def\xref@check@b#1{%
    \ifx\@empty#1%
    \else
        \toks@\@xp{#1}%
```

```
        \edef\@tempb{\lowercase{\def\@nx\@tempa{\the\toks@}}}%
        \@tempb
        \ifx\@tempa#1\relax % all lowercase
            \@ifundefined{bi@#1}{%
                \BibAbbrevWarning{#1}%
            }{%
```

We pass control to `\xref@check@c` here to handle inheritance of multiple fields properly. This means some of the checking we've just done gets done again, but I can live with that.

```
                \let#1\@empty
                \xref@check@c\@tempa
            }%
        \fi
    \fi
}
```

`\xref@check@c` Resolve an `xref` field.

```
\def\xref@check@c#1{%
    \ifx#1\@empty
    \else
        \begingroup
            \@apply\auto@protect\amsrefs@textsymbols
            \@apply\auto@protect\amsrefs@textaccents
            \let\DSK@def\xref@add@toks
            \let\DSK@append\xref@append
            \toks@\@emptytoks
            \let\bib@reset\@empty
```

The `\@for` here is just a fancy way of expanding `#1`. (Or is it?)

```
            \@for\xref@ID:=#1\do{%
                \@ifundefined{bi@\xref@ID}{%
                    \XRefWarning{\xref@ID}%
                }{%
                    \csname bi@\xref@ID\endcsname
                }%
            }%
            \edef\@tempa{\endgroup\the\toks@}%
        \@tempa
    \fi
}
```

`\xref@add@toks` If any title occurs in an `xref`ed item, assume that it is a book title. This might not always be the best assumption? Let's see how it goes though. [mjd,2001-12-11]

Arguments:
`#1 <-` `\bib'`*field*.
`#2 <-` *value*.

```
\def\xref@add@toks#1#2#3{%
```

```
    \ifx#1\@empty
        \edef\@tempa{%
            \@nx\add@toks@{\@xp\@nx\csname\rkv@setter#1\endcsname{#2}{#3}}%
        }%
        \@tempa
    \else
        \in@\bib'title{#1}%
        \ifin@
            \ifx\bib'booktitle\@empty
                \edef\@tempa{%
                    \@nx\add@toks@{%
                        \@xp\@nx\csname set:bib'booktitle\endcsname
                    }%
                }%
                \@tempa
                \add@toks@{{#2}{#3}}%
            \fi
        \fi
    \fi
}
\def\xref@append#1#2#3#4{%
    \edef\@tempa{%
        \@nx\add@toks@{\@xp\@nx\csname\rkv@setter#2\endcsname{#3}{#4}}%
    }%
    \@tempa
}
```

\BibAbbrevWarning

```
\def\BibAbbrevWarning#1{\amsrefs@warning{Abbreviation '#1' undefined}}
```

\XrefWarning

```
\def\XRefWarning#1{\amsrefs@warning{Xref '#1' undefined}}
```

6.12.4 Bib field preprocessing

\current@primary

```
\let\current@primary\@empty
```

\previous@primary

```
\let\previous@primary\@empty
```

\save@primary

```
\IfOption{nobysame}{%
    \let\save@primary\@empty
}{%
    \def\save@primary{%
        \global\let\previous@primary\current@primary
    }%
}
```

\bib@field@patches Depending on your point of view, this macro either puts the bibitem into a canonical form or, alternatively, it fudges the data to fit our model. Either way, it simplifies formatting the bibliography.

```
\def\bib@field@patches{%
    \ifx\bib'author\@empty
        \ifx\bib'editor\@empty
            \let\current@primary\bib'translator
            \let\print@primary\PrintTranslatorsA
        \else
            \let\current@primary\bib'editor
            \let\print@primary\PrintEditorsA
        \fi
    \else
        \let\current@primary\bib'author
        \let\print@primary\PrintAuthors
    \fi
    \ifx\bib'address\@empty
        \let\bib'address\bib'place
    \fi
    \ifx\bib'organization\@empty
        \ifx\bib'institution\@empty
            \let\bib'organization\bib'school
        \else
            \let\bib'organization\bib'institution
        \fi
    \fi
    \ifx\bib'date\@empty
        \ifx\bib'year\@empty
            \let\bib@year\bib'status
        \else
            \bib@parsedate\bib'year
        \fi
    \else
        \bib@parsedate\bib'date
    \fi
```

Example 21 on page 74 of *Mathematics into Type* [2] seems to indicate that when the year serves as the volume number, the date should be suppressed. If so, this is where that is done.

```
    \def\@tempa{year}%
    \ifx\bib'volume\@tempa
        \let\bib'volume\bib@year
        \let\bib'date\@empty
    \fi
```

Some journals have "numbers" but no "volumes". AMS house style is to treat the number as volume.

```
    \ifx\setbib@@\setbib@article
        \ifx\bib'volume\@empty
            \ifx\bib'number\@empty\else
```

```
                \let\bib'volume\bib'number
                \let\bib'number\@empty
            \fi
        \fi
    \fi
```

`\bib'language` is used for producing the printed rendition of the language. `\bib@language` needs to be in the form required by `\selectlanguage`.

```
    \bib@language@fixup
}
```

6.12.5 Date setup

`\bib@year`

```
\let\bib@year\@empty
```

`\bib@month`

```
\let\bib@month\@empty
```

`\bib@day`

```
\let\bib@day\@empty
```

`\bib@parsedate` Parse an ISO 8601 date into its year, month and day components, but without actually verifying that any of the components are numeric. Hmmm.

```
\def\bib@parsedate#1{%
    \@xp\bib@parsedate@a#1---\@nil
}
```

`\bib@parsedate@a`

```
\def\bib@parsedate@a#1-#2-#3-#4\@nil{%
    \def\bib@year{#1}%
    \def\bib@month{#2}%
    \def\bib@day{#3}%
```

The rest of this macro tries to rewrite `\bib'date` into a normalized form. I'm not sure if this is a good idea.

```
    \ifx\@empty\bib@day
        \ifx\@empty\bib@month
            \let\bib'date\bib@year
        \else
            \def\bib'date{#1-#2}%
        \fi
    \else
        \def\bib'date{#1-#2-#3}%
    \fi
}
```

6.12.6 Language setup

\bib@language@fixup

```
\def\bib@language@fixup{%
    \ifx\bib'hyphenation\@empty
        \ifx\bib'language\@empty
            \let\bib@language\biblanguagedefault
        \else
            \let\bib@language\bib'language
        \fi
    \else
        \let\bib@language\bib'hyphenation
    \fi
    \def\@tempa##1 ##2\@nil{\lowercase{\def\bib@language{##1}}}%
```

The mysterious `\@firstofone` here is to preserve the space before the `\@nil`.

```
    \@firstofone{\@xp\@tempa\bib@language} \@nil
}
```

\bib@selectlanguage

For `\bib` purposes we are interested mainly in testing whether the hyphenation patterns are the same. So we use an if-same-patterns test (by which `babel`'s 'english' and 'american' compare as equal) rather than an if-same-language test. Also, the way that the `\selectlanguage` command checks to see whether a language has been properly defined for babel use is to see if `\dateLANGUAGE` is defined. And if we tried to select an undefined language, the result would be a LaTeX error.

```
\def\bib@selectlanguage{%
    \@ifsame@patterns{\languagename}{\bib@language}{}{%
        \@ifundefined{date\bib@language}{}{%
            \@xp\selectlanguage\@xp{\bib@language}%
        }%
    }%
}
```

\@ifsame@patterns

```
\def\@ifsame@patterns#1#2{%
    \@xp\@ifsamepat\csname l@#1\@xp\endcsname\csname l@#2\endcsname
}
```

\@ifsamepat

```
\def\@ifsamepat#1#2{%
    \ifnum \ifx\relax#1\m@ne\else#1\fi = \ifx\relax#2\m@ne\else#2\fi
        \@xp\@firstoftwo
    \else
        \@xp\@secondoftwo
    \fi
}
```

\languagename
\biblanguageEnglish
\biblanguagedefault
\bib@language

```
\providecommand{\languagename}{english}
\def\biblanguageEnglish{english}
\let\biblanguagedefault\biblanguageEnglish
\let\bib@language\@empty
```

6.12.7 Citation label setup

\generate@label

```
\let\generate@label\relax
```

\cite@label

```
\def\cite@label{\@currentlabel}
```

\alpha@label

```
\let\alpha@label\relax
```

\alpha@label@

```
\def\alpha@label@{%
    \ifx\@empty\bib'label
        \def\thebib{\CurrentBib}%
    \else
        \let\thebib\bib'label
    \fi
}%
```

\amsrefs@option@numeric

```
\def\amsrefs@option@numeric{%
    \let\alpha@label\relax
    \let\generate@label\relax
    % \@nmbrlisttrue
}
```

\amsrefs@option@alphabetic

```
\def\amsrefs@option@alphabetic{%
    \let\alpha@label\alpha@label@
    \let\generate@label\generate@alphalabel
    \let\calc@author@part\calc@author@part@
    \let\@suffix@format\@alph
    \let\append@label@year\append@label@year@
    % \@nmbrlistfalse
}
```

refs@option@shortalphabetic

```
\def\amsrefs@option@shortalphabetic{%
    \let\alpha@label\alpha@label@
    \let\generate@label\generate@alphalabel
    \let\calc@author@part\calc@author@part@short
    \let\@suffix@format\@arabic
    \let\append@label@year\@empty
    % \@nmbrlistfalse
}
```

`\bib@cite` When `\bib@cite` is called, author name and year are available in `\bib@author` and `\bib@year`.

Arguments:
`#1 <-` *citekey*.

```
\def\bib@cite#1{%
    \def\CurrentBib{#1}%
    \alpha@label            % modify \thebib if necessary
    \item\leavevmode
    \SK@\SK@@label{#1}%
    \@xp\bib@cite@a\csname b@#1\endcsname
    \bibcite@write{#1}%
}

\def\bib@cite@a#1{%
    \ifx\relax#1%
        \begingroup
            \auto@protect\etaltext
            \protected@edef\@tempa{%
                \gdef\@nx#1{%
                    \@nx\citesel 01{\cite@label}{\bib@label@year}{}%
                }%
            }%
        \@xp\endgroup
        \@tempa
    \else
        \@xp\bib@cite@check\@xp#1#1\@empty\@empty\@empty\@empty\@empty
    \fi
}
```

`\bib@cite@check` For the citation key we want to check if it is already defined. But there is a slight problem. There is already one control sequence in use for each bibliography entry, to store the label or the author/year information needed by `\cite`. If we introduce another control sequence to check whether a particular cite is multiply defined, then we double the number of control sequences used. For a large bibliography in a book this is fairly serious. This is addressed by using a `\citesel` function.

Arguments:
`#1 <- \b@`*citekey*.
`#2 <- \citesel`.
`#3 <-` *cited?*.
`#4 <-` *used?*.
`#5 <-` *label*.
`#6 <-` *year*.
`#7 <-` *backrefs*.

```
\def\bib@cite@check#1#2#3#4#5#6#7{%
    \ifx 1#4\relax
        \DuplicateBibKeyWarning
```

```
\else
```

This has gotten *way* out of hand.

```
        \begingroup
            \auto@protect\etaltext
            \@apply\auto@protect\amsrefs@textsymbols
            \@apply\auto@protect\amsrefs@textaccents
            \@tempswafalse
            \in@\CitePrintUndefined{#5}%
            \ifin@
                \let\@tempa\@empty
            \else
                \def\@tempa{#5}%
            \fi
            \ifx\@tempa\@empty
            \else
                \@xp\ifx\@xp\@currentlabel\cite@label
                    \edef\@tempb{\cite@label}%
                \else
                    \let\@tempb\cite@label
                \fi
                \ifx\@tempa\@tempb
                    \def\@tempa{#6}%
                    \ifx\@tempa\bib@label@year
                    \else
                        \@tempswatrue
                    \fi
                \else
                    \@tempswatrue
                \fi
            \fi
            \if@tempswa
                \@ifempty{#6}{%
                    \def\@tempa{#5}%
                    \let\@tempb\cite@label
                }{%
                    \def\@tempa{#5, #6}%
                    \def\@tempb{\cite@label, \bib@label@year}%
                }%
                \amsrefs@warning{Citation label for \extr@cite#1 is
                    changing from `\@tempa ' to `\@tempb '}%
            \fi
            \protected@edef\@tempa{%
                \gdef\@nx#1{%
                    \@nx\citesel #31{\cite@label}{\bib@label@year}{#7}%
                }%
            }%
        \@xp\endgroup
        \@tempa
    \fi
```

```
}
```

\bib@label@year

```
\let\bib@label@year\@empty
```

\DuplicateBibKeyWarning

```
\def\DuplicateBibKeyWarning{%
    \amsrefs@warning{%
        Duplicate \protect\bib\space key
        `\CurrentBib ' detected\MessageBreakNS}%
}
```

\DuplicateBibKeyWarning

```
\def\DuplicateBibLabelWarning{%
    \amsrefs@warning{%
        Duplicate biblabel stem `\current@stem ' detected.\MessageBreakNS
        This usually means the order of the bibitems\MessageBreakNS
        is incompatible with the style of labels\MessageBreakNS
        you are using}%
}
```

\bibcite@write

```
\def\bibcite@write#1{%
    \if@filesw
        \begingroup
            \let\citesel\citesel@write
            \csname b@#1\endcsname
        \endgroup
    \fi
}
```

\citesel@write

```
\def\citesel@write#1#2#3#4#5{%
    \toks@{{#3}{#4}}%
    \immediate\write\@auxout{\string\bibcite{\CurrentBib}{\the\toks@}}%
}
```

Because duplicate bibs are caught immediately, we don't need \bibcite to run \@testdef.

```
\AtEndDocument{\let\bibcite\@gobbletwo}
```

6.12.8 Printing the bibliography

\bibname

```
\providecommand{\bibname}{Bibliography}
```

\refname

```
\providecommand{\refname}{References}
```

`\bib@div@mark` The AMS document classes automatically take care of the page marks for `\section*` and `\chapter*`, but for the standard classes, we need to make sure that `\@mkboth` gets invoked.

```
\let\bib@div@mark\@gobble
```

This is verbose, but probably safer than any alternative.

```
\@ifclassloaded{amsbook}{}{%
    \@ifclassloaded{amsart}{}{%
        \@ifclassloaded{amsproc}{}{%
            \def\bib@div@mark#1{%
                \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
            }%
        }%
    }%
}
```

`bibchapter` We need to take a little extra trouble here to pre-expand the `\bibname`.

```
\newenvironment{bibchapter}[1][\bibname]{%
    \begingroup
        \protected@edef\@{%
                \endgroup
            \protect\chapter*{#1}%
            \protect\bib@div@mark{#1}%
        }%
        \@
}{\par}
```

`bibsection` And here to pre-expand the `\refname`.

```
\newenvironment{bibsection}[1][\refname]{%
    \begingroup
        \protected@edef\@{%
                \endgroup
            \ifx\@bibtitlestyle\undefined
                \protect\section*{#1}%
            \else
                \protect\@bibtitlestyle
            \fi
            \protect\bib@div@mark{#1}%
        }%
        \@
}{\par}
```

`bibdiv` Here we try to guess whether this is a book-like document or an article-like document.

```
\@ifundefined{chapter}{%
    \newenvironment{bibdiv}{\bibsection}{\endbibsection}
}{%
    \newenvironment{bibdiv}{\bibchapter}{\endbibchapter}
}
```

This is what the standard book class has for the bibliography title:

```
\newenvironment{thebibliography}[1]
     {\chapter*{\bibname
        \@mkboth{\MakeUppercase\bibname}{\MakeUppercase\bibname}}%
      \list{\@biblabel{\@arabic\c@enumiv}}%
```

thebibliography

```
\renewenvironment{thebibliography}[1]{%
    \bibdiv
    \biblist[\resetbiblist{#1}]%
}{%
    \endbiblist
    \endbibdiv
}
```

6.13 Name, journal and publisher abbreviations

The commands \DefineName, \DefinePublisher, and \DefineJournal are provided to make abbreviations a little easier.

\DefineName

```
\newcommand{\DefineName}[2]{%
    \bib*{#1}{name}{name={#2}}%
}
```

\DefineJournal

```
\newcommand{\DefineJournal}[4]{%
    \bib*{#1}{periodical}{
        issn={#2},
        journal={#4}
    }%
}
```

\DefinePublisher Note that an explicit address field in a \bib entry will override the address supplied as part of a \DefinePublisher.

```
\newcommand{\DefinePublisher}[4]{%
    \bib*{#1}{publisher}{%
        publisher={#3},
        address={#4}
    }%
}
```

6.14 Processing .ltb files

If you have a file that contains amsrefs-style \bib entries, you can use it as a database and extract items from it for use in another document. In typical relatively simple scenarios, the extraction can be done by LaTeX itself on the first pass, so that citations in the text will be successfully resolved on the second pass (possibly even the first, depending on what kind of bibliography sorting is used).

\bibselect

```
\newcommand{\bibselect}{%
    \@ifstar{%
        \let\@bibdef\copy@bibdef
        \BibSelect
    }{%
        \let\@bibdef\selective@bibdef
        \BibSelect
    }%
}
```

\BibSelect

```
\newcommand{\BibSelect}[2][\bblname]{%
    \if@filesw
        \typeout{Trying to create bbl file ‘#1.bbl’ ...}%
        \def\bibselect@msg{%
            \typeout{ ... rats. Unable to create bbl file.}%
        }%
        \let\@open@bbl@file\OpenBBLFile
        \@for\@tempa:=#2\do{\ReadBibData{\@tempa}}%
    \fi
    \@close@bbl@file
    \@apply\g@undef\bibdefer@list
    \global\let\bibdefer@list\@empty
```

Now read the `.bbl` file we just created.

```
    \let\@bibdef\normal@bibdef
    \@input@{#1.bbl}%
    \let\BibSelect\MultipleBibSelectWarning
}
```

\MultipleBibSelectWarning

```
\newcommand\MultipleBibSelectWarning[2][]{%
    \amsrefs@warning{%
        Multiple \string\bibselect ’s found (only one
        \string\bibselect\space per biblist environment is allowed)%
    }%
}
```

\bblname

```
\def\bblname{\jobname}
```

\bib@dbfile

```
\newread\bib@dbfile
```

\ReadBibData

```
\newcommand{\ReadBibData}[1]{%
    \IfFileExists{#1.ltb}{%
        \openin\bib@dbfile=\@filef@und \relax
```

```
    }{%
        \IfFileExists{#1.ltx}{%
            \openin\bib@dbfile=\@filef@und \relax
        }{%
            \IfFileExists{#1.tex}{%
                \openin\bib@dbfile=\@filef@und \relax
            }{%
                \begingroup
                    \NoBibDBFile{#1}%
                    \let\ReadBibData@a\endgroup
            }%
        }%
    }%
    \ReadBibData@a
}
```

\NoBibDBFile

```
\def\NoBibDBFile#1{%
    \amsrefs@warning{No data file #1.ltb (.ltx, .tex) found}%
}
```

\ReadBibData@a

```
\def\ReadBibData@a{%
    \ProvidesFile{\@filef@und}\relax
    \begingroup
        \let\star@bibdef\defer@bibdef
        \ReadBibLoop
    \endgroup
    \closein\bib@dbfile
}
```

\ReadBibLoop

```
\def\ReadBibLoop{%
    \ifeof\bib@dbfile
        \@xp\@gobble
    \else
        \read\bib@dbfile to\CurLine
```

The \@empty is in case \CurLine is empty.

```
        \@xp\ReadBibLoop@a\CurLine\@empty\@nil
    \fi
    \ReadBibLoop
}
```

\ReadBibLoop@e This traps top-level \bib commands. Note that:

- If \CurLine doesn't contain a complete \bib entry, the code chokes.
- I \bib is not the very first non-space token in a line, it will not be recognized.

```
\long\def\ReadBibLoop@a#1#2\@nil{%
    \ifx\bib#1%
        \CurLine % just exec it
    \else
```

We're not done yet. The line may contain something like `\DefineName`, so we need to expand the first macro in the line and see if it starts with `\bib`. But first we check to make sure that the token we're about to expand isn't `\endinput`.

```
        \ifx\endinput#1%
            \let\ReadBibLoop\@empty
        \else
```

And this `\@empty` is for the admittedly unlikely case that `\CurLine` isn't empty, but its expansion is.

```
            \@xp\ReadBibLoop@b#1#2\@empty\@nil
        \fi
    \fi
}
```

`\ReadBibLoop@b`

```
\long\def\ReadBibLoop@b#1#2\@nil{%
    \ifx\bib#1%
        \CurLine % just exec it
    \fi
}

\let\bbl@out=\relax
\let\bbl@write\@gobble
\let\@open@bbl@file\relax
\let\@close@bbl@file\relax
```

`\OpenBBLFile`

```
\def\OpenBBLFile{%
    \if@filesw
        % Just use the next unused output stream
        \count@\count17
        \advance\count@\@ne
        \ifnum\count@<\sixt@@n
            \global\chardef\bbl@out=\count@
            \immediate\openout\bbl@out=\bblname.bbl\relax
            \global\let\@close@bbl@file\CloseBBLFile
            \gdef\bbl@write{\immediate\write\bbl@out}%
        \else
            \ch@ck\count@\sixt@@n\write
        \fi
    \fi
    \global\let\@open@bbl@file\relax
}
```

`\CloseBBLFile`

```
\def\CloseBBLFile{%
    \immediate\closeout\bbl@out\relax
    \global\let\@close@bbl@file\relax
    \global\let\bbl@write\@gobble
    \global\let\bbl@out\relax
}
```

6.15 Citation processing

6.15.1 The `\citesel` structure

The information used by `\cite` for key `moo` is stored in `\b@moo` in the form

```
\citesel{status1}{status2}{label}{year}{backref-info}
```

The first status flag is 1 if this key has already been cited earlier in the same document; 0 otherwise. This is used in some bibliography schemes to print a full list of author names for the first citation and an abbreviated author list for subsequent citations.

The second status flag is 1 if this key has already been used by a define-cite command (such as `\bib`); 0 otherwise. This makes it possible to issue a warning message as soon as the conflict is seen, on the first LaTeX run, instead of on a subsequent run during the processing of the `.aux` file.

When an author/year citation scheme is in use, args 3 and 4 hold respectively author names and year. Otherwise arg 3 simply holds a cite label and arg 4 is empty.

And finally, arg 5 holds a list of backref pointers indicating the locations in the document where this entry has been cited.

`\citesel@update`

```
\def\citesel@update#1#2#3#4#5#6{%
    \gdef#6{\citesel 1#2{#3}{#4}{#5}}%
}
```

`\citesel@number`

```
\def\citesel@number#1#2#3#4#5{#3}
```

`\citesel@year`

```
\def\citesel@year#1#2#3#4#5{#4}
```

`\citesel`

```
\let\citesel\citesel@number
```

6.15.2 The basic `\cite` command

Here is the difference between the various optional forms of `\cite`:

```
\cite{xyz}        -> \cite@a\citesel{xyz}{}
                  -> \cite@bc\b@xyz\citesel{}

\cite{xyz}*{blub} -> \cite@a\citesel{xyz}{blub}
                  -> \cite@bc\b@xyz\citesel{blub}
```

```
\cite[blub]{xyz}  -> \cite@a\citesel{xyz}{blub}
                  -> \cite@bc\b@xyz\citesel{blub}
```

Canceling the old LaTeX definition of `\cite␣` prevents certain problems that could arise with the `showkeys` package.

```
\expandafter\let\csname cite \endcsname\relax
```

`\cite` Need to handle the standard [...] option for compatibility's sake.

```
\renewcommand{\cite}[2][]{%
    \if\cite@single#2,\@gobble \else\MultipleCiteKeyWarning{#2}{#1}\fi
    \@ifempty{#1}{%
        \cites@o{#2}%
    }{%
        \ObsoleteCiteOptionWarning
        \cites@a{*{#1}}{#2}%
    }%
}
```

`\MultipleCiteKeyWarning`

```
\def\MultipleCiteKeyWarning#1#2{%
    \amsrefs@warning{%
        Use of \string\cites\space is recommended instead of %
        \string\cite\space\MessageBreak
        for multiple cites '#1'}%
    \@ifnotempty{#2}{%
        \amsrefs@warning{Star option requires \string\citelist\space here}%
    }%
    \global\let\MultipleCiteKeyWarning\@gobbletwo
}
```

`\ObsoleteCiteOptionWarning`

```
\def\ObsoleteCiteOptionWarning{%
    \amsrefs@warning{%
        The form \string\cite{...}*{...} is recommended\MessageBreak
        instead of \string\cite[...]{...}}%
    \global\let\ObsoleteCiteOptionWarning\@empty
}
```

`\cite@single`

```
\edef\cite@single#1,#2{\iffalse{\fi\iffalse{\fi\string}#2.\string}}
```

`\cites@o`

```
\def\cites@o#1{\star@{\cites@oo{#1}}{}}
```

`\cites@oo`

```
\def\cites@oo#1#2{\@ifempty{#2}{\cites@a{}{#1}}{\cites@a{*{#2}}{#1}}}
```

\cites@a

```
\def\cites@a#1#2{%
    \begingroup
        \toks@{\endgroup \cites@b{#1}}%
        \vdef\@tempa{#2}%
        \edef\@tempa{%
            \the\toks@ \@firstofone{\@xp\zap@space\@tempa} \@empty
        }%
        \@tempa,\@empty
        \edef\@tempa{\endgroup\@nx\citelist{\the\toks@}}%
        \@tempa
}
```

\cites@b

```
\def\cites@b#1#2,#3{%
    \begingroup
        \toks@{\InnerCite{#2}#1}%
        \ifx\@empty#3\@xp\@gobble\fi
        \cites@c#3%
}
```

\cites@c

```
\def\cites@c#1,#2{%
    \add@toks@{\InnerCite{#1}}%
    \ifx\@empty#2\@xp\@gobble\fi
    \cites@c#2%
}
```

\citeleft
\citeright
\citemid
\citepunct

These variables are named to follow the precedent set by Arseneau's `cite` package. `\citemid` is used to separate a citation label from additional information such as "Theorem 4.9". `\citepunct` is used to separate multiple cites, unless one of the cites has additional associated information, in which case `\CiteAltPunct` is used.

```
\def\citeleft{[}
\def\citeright{]}
```

```
\def\citemid{,\penalty9999 \space}
\def\citepunct{,\penalty9999 \hskip.13em plus.1em minus.05em\relax}
```

\citeAltPunct

When a citation list contains one or more citations with optional arguments, we replace `\citemid` by `\CiteAltPunct`.

```
\def\citeAltPunct{;\ }
```

\citeform

This is used for formatting the citation label. It can be used, for example, to bolden the labels (as in `amsbook` and `amsproc`) or to do more elaborate things such as convert the numbers to roman numerals. By default, it's just a no-op.

Note that currently there is no corresponding macro for changing the formatting of `\cite`'s optional argument. This is probably a bug.

```
\providecommand{\citeform}{\@firstofone}
```

`\citelist` The `\@citelist` indirection turns out to be helpful in implementing the `\ocites` command for the `author-year` option.

```
\DeclareRobustCommand{\citelist}{\@citelist}
```

`\@citelist`

```
\def\@citelist#1{%
    \leavevmode
    \begingroup
        \@citestyle
        \citeleft\nopunct   % suppress first \citepunct
        \cite@begingroup
            \in@*{#1}%
            \ifin@
                \let\citepunct\citeAltPunct
            \fi
            \let\cite@endgroup\@empty
            \cites@init
            \def\citeleft{\@addpunct{\citepunct}}%
            \let\citeright\ignorespaces
            \def\cite{\InnerCite}%
            \process@citelist{#1}%
        \endgroup
        \citeright
    \endgroup
}
```

`\@citestyle` Reset the font to an upright, medium font (e.g. `cmr`), per AMS style. Also set `\mathsurround` = 0 pt just in case there are subscripts in the cite numbers (from `\etalchar`, for example).

```
\providecommand{\@citestyle}{\m@th\upshape\mdseries}
```

`\cite@begingroup` Grouping that encloses an entire cite block (a single cite or a list of cites).

```
\def\cite@begingroup{\begingroup\let\cite@begingroup\relax}
```

`\cite@endgroup`

```
\let\cite@endgroup\endgroup
```

`\cites@init` This needs to be called at the beginning of a list of cites to reset a few things.

```
\def\cites@init{%
    \gdef\prev@names{???}%
    \let\cites@init\@empty
}
```

`\InnerCite`

```
\newcommand{\InnerCite}[1]{\star@{\cite@a\citesel{#1}}{}}
```

`\cite@a` The job of `\cite@a` is to convert the cite key to all catcode-12 characters and remove any spaces it might contain before passing it on to `\cite@b`.

Arguments:
#1 <- \CITESEL.
#2 <- *citekey*.

```
\def\cite@a#1#2{%
    \BackCite{#2}%
    \cite@begingroup
        \cites@init
        \let\citesel#1\relax
        \ifx\citesel\citesel@author
            \let\citeleft\@empty
            \let\citeright\@empty
        \fi
        \begingroup
            \toks@{\endgroup \cite@b}%
            \vdef\@tempa{#2}%
            \edef\@tempa{%
                \the\toks@{\@firstofone{\@xp\zap@space\@tempa} \@empty}%
            }%
        \@tempa
}
```

\cite@b *Arguments*:
#1 <- *citekey*.
#2 <- *star-optional-arg*.

```
\def\cite@b#1#2{%
    \@xp\cite@bc\csname b@#1\@xp\endcsname {#1}{#2}%
}
```

\cite@bc If it's uninitialized, plug in an empty cite structure. \cite@bc should be executed only once for a given instance of a cite key. All further processing should go through \cite@cj.

```
\def\cite@bc#1#2{%
    \ifx#1\@@undefined \global\let#1\relax \fi
    \ifx#1\relax       \global\let#1\empty@cite \fi
    \@xp\cite@nobib@test#1{}{}{}{}\@nil#1%
    \cite@cj#1%
}
```

\empty@cite

```
\def\empty@cite{\citesel 00{}{}{}}
```

\cite@nobib@test If arg 4 is empty, it means there wasn't any \bib command that defined a valid label.

Arguments:
#1 <- \citesel.
#2 <- *cited?*.
#3 <- *used?*.

#4 <- *label*.
#5 <- *backrefs*.
#6 <- \b@*citekey*.

```
\def\cite@nobib@test#1#2#3#4#5\@nil#6{%
    \@ifempty{#4}{%
        \G@refundefinedtrue
        \UndefinedCiteWarning#6%
        \xdef#6{\@nx\citesel #2#3{%
            \@nx\CitePrintUndefined{\extr@cite#6}}{}{}}%
    }{}%
}
```

\UndefinedCiteWarning This is a copy of the standard warning from \@citex.

```
\def\UndefinedCiteWarning#1{%
    \@latex@warning{%
        Citation `\extr@cite#1' on page \thepage\space undefined}%
}
```

\CitePrintUndefined

```
\DeclareRobustCommand{\CitePrintUndefined}[1]{%
    \begingroup\fontshape{n}\fontseries\mddefault \ttfamily ?#1\endgroup
}
```

\CPU@normal This has to be a \let, not a \def.

```
\let\CPU@normal\CitePrintUndefined
```

\cite@cj *Arguments*:
#1 <- \b@*citekey*.
#2 <- *star-optional-arg*.

```
\def\cite@cj#1#2{%
        \leavevmode
            \begingroup
                \cite@cb#1% write info to aux file
                \ar@SK@cite#1%
                \@citeleft
                \ar@hyperlink{#1}%
                \@ifnotempty{#2}{\citemid{#2}}%
                \citeright
            \endgroup
            \ignorespaces % ignore spaces inside \citelist
        \cite@endgroup
}
```

\@citeleft The following definition provides some indirection that helps to deal with author-year object cites.

```
\def\@citeleft{\citeleft}
```

\cite@cb

```
\def\cite@cb#1{%
    \if@filesw
        \immediate\write\@auxout{\string\citation{\extr@cite#1}}%
    \fi
```

Define \citesel to make \b@whatever update itself.

```
    \begingroup
        \let\citesel\citesel@update
        #1#1%
    \endgroup
}
```

\extr@cite Extract *citekey* from \b@*citekey*.

```
\def\extr@cite{\@xp\@gobblethree\string}
```

6.15.3 Fancier \cite commands

\cites A list of simple cites. Make it robust in case used inside a figure caption. (But then also, by the way, listoffigures should provide special handling.)

```
\DeclareRobustCommand{\cites}{\cites@a{}}
```

\citen This is just to keep the showkeys package from clobbering the wrong part of our definition of \cite:

```
\providecommand{\citen}{\ocite}
```

\ycite \cite gets redefined inside of \citelist, so we need to \def \ycite here instead of just \letting everything to \cite.

```
\def\ycite{\cite}
```

\ycites

```
\let\ycites\cites
```

\ocite

```
\let\ocite\ycite
```

\ocites

```
\let\ocites\cites
```

\fullcite

```
\let\fullcite\cite
```

\fullocite

```
\let\fullocite\ocite
```

\citeauthor

```
\let\citeauthor\ycite
```

\citeauthory

```
\let\citeauthory\ycite
```

6.15.4 The \nocite command

\nocite

```
\renewcommand{\nocite}[1]{\othercites{#1}}
```

\othercites

```
\newcommand{\othercites}[1]{%
    \cite@begingroup
        \let\BackCite\@gobble
        \let\cite@endgroup\@empty
        \def\citelist{\othercitelist}%
        \cites{#1}%
}
```

\othercitelist

```
\newcommand{\othercitelist}[1]{%
    \cite@begingroup
        \let\cite@endgroup\@empty
        \cites@init
        \let\citeleft\relax
        \let\citeright\ignorespaces
        \def\InnerCite{\OtherCite}%
        \def\cite@cj ##1##2{%
            \begingroup
                \@xp\citesel##1%
                \cite@cb ##1%
            \endgroup
```

If we detect \nocite{*}, we globally alias \selective@bibdef to \copy@bibdef so that all succeeding \bibselect commands act like \bibselect*.

```
            \@xp\ifx\csname b@*\endcsname ##1%
                \global\let\selective@bibdef\copy@bibdef
            \fi
            \ignorespaces
            \cite@endgroup
        }%
    #1\relax
    \endgroup
}
```

\OtherCite

```
\def\OtherCite#1{\cite@a\citesel@other{#1}{}{}}
```

\citesel@other

```
\def\citesel@other#1#2#3#4#5#6{}
```

\b@* This provides a dummy definition to keep things like \nocite{*} from generating an error message.

```
\@namedef{b@*}{\citesel 11{*}{*}{*}}
```

6.15.5 Citation sorting

\process@citelist@sorted

```
\def\process@citelist@sorted#1{%
    \ifx\citesel\citesel@number
        \cite@sorted@s #1\cite@sorted@e
    \else
        \NonNumericCiteWarning
        \process@citelist@unsorted{#1}%
    \fi
}
```

\NonNumericCiteWarning

```
\def\NonNumericCiteWarning{%
    \amsrefs@warning{%
        Unable to confirm that cite keys are numeric: not sorting%
    }%
}
```

\process@citelist@unsorted

```
\def\process@citelist@unsorted#1{%
    \ignorespaces#1\relax
}
```

\process@citelist By default, citation lists will be sorted.

```
\let\process@citelist\process@citelist@sorted
```

\CPU@sort By defining this as TEX's maxint, undefined cites migrate to the end of a sorted list.

```
\def\CPU@sort#1{2147483647}
```

\cite@sorted@s Here's where we prepare to sort the citations and (optionally) compress ranges.

```
\def\cite@sorted@s{%
    \begingroup
        \let\CitePrintUndefined\CPU@sort
        \let\cite@cjs\cite@cj
        \let\cite@cj\cite@compress
        \begingroup
            \toks@\@emptytoks
            \let\cite@cj\cite@sort
            \ignorespaces
}
```

\cite@sorted@e

```
\def\cite@sorted@e{%
        \@xp\endgroup
        \the\toks@
        \cite@dash
        \prev@cite
    \endgroup
}
```

`\cite@sort` This is essentially an insertion sort. I think.

Arguments:
`#1 <- \b@`*citekey*.
`#2 <-` *optional arg*.

```
\def\cite@sort#1#2{%
    \safe@set\@tempcnta#1% highest number so far
    \toks@{\cite@cj#1{#2}}%
    \@temptokena\toks@
    \let\cite@cj\cite@sort@a
    \ignorespaces
}
```

`\cite@sort@a`

```
\def\cite@sort@a#1#2{%
    \safe@set\@tempcntb#1%
    \ifnum\@tempcntb > \@tempcnta
        \add@toks@{\cite@cj#1{#2}}%
        \@tempcnta\@tempcntb
    \else
        \let\cite@cj\cite@sort@b
        \toks@\@emptytoks
        \def\@tempb{\add@toks@{\cite@cj#1{#2}}}%
        \the\@temptokena
        \@tempb
        \let\cite@cj\cite@sort@a
    \fi
    \@temptokena\toks@
    \ignorespaces
}
```

`\cite@sort@b`

```
\def\cite@sort@b#1#2{%
    \safe@set\count@#1%
    \ifnum\@tempcntb < \count@
        \@tempb
        \let\@tempb\@empty
    \fi
    \add@toks@{\cite@cj#1{#2}}%
    \ignorespaces
}
```

6.15.6 Range compression

When the time comes to apply compression, we have at our disposal a list of internal cite calls that looks like this:

```
\cite@cj\b@aaa{opta}\cite@cj\b@bbb{optb}...\cite@cj\b@zzz{optz}
```

where

$$\texttt{\b@aaa} < \texttt{\b@bbb} < \cdots < \texttt{\b@zzz}$$

and the `opt` arguments are possibly null. To print the citations while collapsing sequences of 3 or more contiguous numbers into ranges of the form n–m, we bind `\cite@cj` to a suitably clever function and then execute the list. In the absence of optional arguments, here's the algorithm:

Begin. Enter state 0. This is done by `\cite@sorted@s`.

State 0. The current citation is the beginning of a range (possibly a singleton range). Print it. Then, set *prevnum* := *number* and enter state 1.

State 1. The current citation might be the second element of a range.

Case a) *number* = *prevnum* + 1. Then the current item is definitely the second element of a range. It might be the last element of the range, but we won't know until we examine the following citation. So, save the current citation in `\prev@cite`, set *prevnum* := *number*, and go to state 2.

Case b) *number* ≠ *prevnum* + 1. The current citation is the beginning of a new range. Print it, set *prevnum* := *number* and remain in state 1. (This is essentially identical to stage 0.)

State 2. The current citation might be the third (or later) element of a range.

Case a) *number* = *prevnum* + 1. The current element is definitely part of a range. It might be the last element of the range, but again we won't know until we examine the following citation. Save the current citation in `\prev@cite` and set *prevnum* := *number*. Remain in state 2.

Case b) *number* ≠ *prevnum* + 1. The previous citation was the end of a range and the current citation is the beginning of a new range. Print a dash followed by `\prev@cite`, then set *prevnum* := *number* and enter state 1.

End. If `\prev@cite` is not empty, print it, preceded by a dash if we were in the middle of a range. (This is done by `\cite@sorted@e`.)

The presence of optional arguments complicates things somewhat, since a citation with an optional argument should never participate in range compression. In other words, when we come across an optional argument, we should finish off the preceding range, print the current citation, and then return to the initial state. More precisely, here are the actions taken in each state when there is an optional argument:

State 0. Print the current citation and remain in state 0.

State 1. Print the current citation and return to state 0.

State 2. Print a dash followed by `\prev@cite`. Then print the current citation and return to state 0.

`\prev@cite`

```
1639 \let\prev@cite\@empty
```

\prev@cite@cb There's one further complication: Even though we're suppressing some of the citation numbers, we need to make sure that each citation is recorded in the `.aux` file. So, in case 2a, before we overwrite `\prev@cite`, we first invoke `\prev@cite@cb` to record the previous citation (if any).

```
\def\prev@cite@cb{%
    \ifx\@prev@cite\@empty
    \else
        \begingroup
            \def\cite@print##1##2{%
                \cite@cb##1%
            }%
            \prev@cite
        \endgroup
    \fi
}
```

\cite@print

```
\def\cite@print#1#2{%
    \begingroup
        \let\CitePrintUndefined\CPU@normal
        \cite@cjs#1{#2}%
    \endgroup
}
```

\cite@dash Ok, I lied. There was more than one further complication. Suppose that when we hit the end of the list, we're in state 2. We need to know whether to output a dash or a comma. (For example, both the sequences $[2, 3]$ and $[1, 2, 3]$ will end in state 2 with *prevcite* = 3, but in the former case we want a comma before the 3 and in the latter case we want a dash.) So, rather than printing the dash explicitly, we use `\cite@dash` to keep track of whether a dash is needed.

```
\let\cite@dash\@empty
```

\print@one@dash

```
\def\print@one@dash{%
    \textendash \nopunct
    \let\cite@dash\@empty
}
```

State 0, 1 and 2 each correspond to a different binding for `\cite@cj`. Here they are. The role of *prevnum* is played by `\@tempcnta`, with `\@tempcntb` assisting as *number* at times.

\cite@compress State 0:

```
\def\cite@compress#1#2{%
    \cite@print#1{#2}%
    \@ifempty{#2}{%
        \safe@set\@tempcnta#1%
        \let\cite@cj\cite@compress@a
```

```
    }{}%
}
```

`\cite@compress@a` State 1:

```
\def\cite@compress@a#1#2{%
    \@ifempty{#2}{%
        \advance\@tempcnta\@ne
        \safe@set\@tempcntb#1%
        \ifnum\@tempcnta=\@tempcntb
            \def\prev@cite{\cite@print#1{}}%
            \let\cite@cj\cite@compress@b
        \else
            \cite@print#1{}%
            \@tempcnta\@tempcntb
        \fi
    }{%
        \cite@print#1{#2}%
        \let\cite@cj\cite@compress
    }%
}
```

`\cite@compress@b` State 2:

```
\def\cite@compress@b#1#2{%
    \@ifempty{#2}{%
        \advance\@tempcnta\@ne
        \safe@set\@tempcntb#1%
        \ifnum\@tempcnta=\@tempcntb
            \let\cite@dash\print@one@dash
            \prev@cite@cb
            \def\prev@cite{\cite@print#1{}}%
        \else
            \cite@dash
            \prev@cite
            \let\prev@cite\@empty
            \cite@print#1{}%
            \@tempcnta\@tempcntb
            \let\cite@cj\cite@compress@a
        \fi
    }{%
        \cite@dash
        \prev@cite
        \let\prev@cite\@empty
        \cite@print#1{#2}%
        \let\cite@cj\cite@compress
    }%
}
```

6.15.7 Munging the `.aux` file

`\amsrefs@bibcite` When processing the `.aux` file at begin-document, this is what `\bibcite` will do:

```
\def\amsrefs@bibcite#1{\@xp\bibcite@a\csname b@#1\endcsname}
```

However, `hyperref` also redefines `\bibcite`, so to avoid conflicts and also ensure that it doesn't matter whether `amsrefs` or `hyperref` is loaded first, rather than redefining `\bibcite` directly, we redefine it inside the `.aux` file.

```
\AtBeginDocument{%
    \if@filesw
        \immediate\write\@auxout{%
            \string\@ifundefined{amsrefs@bibcite}{}{%
                \string\let\string\bibcite\string\amsrefs@bibcite
            }%
        }%
    \fi
```

For good measure, we'll redefine it here as well, even though it really shouldn't matter any longer.

```
    \let\bibcite\amsrefs@bibcite
}
```

`\bibcite@a` *Arguments*:
`#1 <-` `\b@`*citekey*.
`#2 <-` `{`*label*`}{}` *or* `{`*author*`}{`*year*`}`.

```
\def\bibcite@a#1#2{%
```

Most of the time arg 1 will already be defined, by an earlier `\citedest` command in the `.aux` file. Then we just need to change the number.

```
    \ifx\relax#1%
        \gdef#1{\citesel 00#2{}}%
    \else
        \begingroup
            \@xp\bibcite@b\@xp#1#1{#2}%
        \endgroup
    \fi
}
```

`\bibcite@b` *Arguments*:
`#1 <-` `\b@`*citekey*.
`#2 <-` `\citesel`.
`#3 <-` *cited?*.
`#4 <-` *used?*.
`#5 <-` *label*.
`#6 <-` *year*.
`#7 <-` *backrefs*.
`#8 <-` `{`*newlabel*`}{`*newyear*`}`.

```
\def\bibcite@b#1#2#3#4#5#6#7#8{\gdef#1{\citesel#3#4#8{#7}}}
```

`\citedest` The `\citedest` command goes into the `.aux` file to provide back-reference support.

```
\newcommand{\citedest}[1]{\@xp\cite@dest\csname b@#1\endcsname}
```

`\cite@dest`

```
\def\cite@dest#1{%
    \ifx\relax#1%
        \gdef#1{\citesel 00{}{}{}}%
    \fi
    \@xp\cite@dest@b\@xp#1#1%
}
```

`\cite@dest@b` *Arguments*:
`#1 <-` `\b@`*citekey*.
`#2 <-` `\citesel`.
`#3 <-` *cited?*.
`#4 <-` *used?*.
`#5 <-` *label*.
`#6 <-` *year*.
`#7 <-` *backrefs*.
`#8 <-` {*more backrefs*}.

```
\def\cite@dest@b#1#2#3#4#5#6#7#8{%
    \@ifempty{#7}{%
        \def#1{\citesel #3#4{#5}{#6}{{#8}}}%
    }{%
        \gdef#1{\citesel #3#4{#5}{#6}{#7,{#8}}}%
    }%
}
```

6.15.8 Back references

`\ifBR@verbose`

```
\@ifundefined{ifBR@verbose}{\let\ifBR@verbose\iffalse \let\fi\fi}{}%
```

`\BackCite`

```
\let\BackCite\@gobble
```

`\back@cite`

```
\def\back@cite#1{%
    \ifBR@verbose
        \PackageInfo{backref}{back cite \string `\extr@cite#1'}%
    \fi
    \Hy@backout{#1}%
}
```

`\print@backrefs` In an AMS-style bibliography, the backref info might follow the final period of the reference, or it might follow some *Mathematical Reviews* info, without a period.

```
\def\print@backrefs#1{%
    \space\SentenceSpace$\uparrow$\csname br@#1\endcsname
}
```

\PrintBackRefs

```
\let\PrintBackRefs\@gobble
```

6.15.9 hyperref, showkeys and shaderef support

\shade@cite

```
\newcommand{\shade@cite}{\printref}
```

\format@cite

```
\def\format@cite#1{\shade@cite{\citeform{#1}}}
```

\ar@hyperlink

```
\def\ar@hyperlink#1{%
    \hyper@@link[cite]{}{cite.\extr@cite#1}{\format@cite{#1}}%
}
```

\ar@SK@cite

```
\def\ar@SK@cite#1{\@bsphack\@xp\SK@\@xp\SK@@ref\@xp{\extr@cite#1}\@esphack}
```

Turn off hyperref and showkeys support if those packages don't appear to be loaded.

```
\AtBeginDocument{%
    \@ifpackageloaded{shaderef}{}{%
        \let\shade@cite\@firstofone
    }%
    \@ifpackageloaded{hyperref}{}{%
        \def\ar@hyperlink{\format@cite}%
        \let\hyper@anchorstart\@gobble
        \let\hyper@anchorend\relax
        \let\Hy@raisedlink\@firstofone
    }%
    \@ifpackageloaded{showkeys}{%
        \@ifpackagewith{showkeys}{notcite}{%
            \let\ar@SK@cite\@gobble
        }{}
    }{%
        \let\ar@SK@cite\@gobble
        \let\SK@@label\@gobble
        \let\SK@\@gobbletwo
    }%
}
```

6.16 Lexical structure of names

Before we can begin parsing names, we need to give some thought to the lexical structure of names. For the remainder of this document, when we refer to a "name" and especially when we speak of a name as a macro argument, we assume that the only tokens contained in the name are

- letters and punctuation (i.e., characters with catcode 11 or 12),
- ties (the token $\texttt{\~}_{13}$),
- accent commands, such as `\"` or `\k`,
- text symbol macros, such as `\i`, `\ae` or `\cprime`,
- grouping characters (braces).

In addition to their normal function of delimiting macro arguments, braces inside names have the following special functions:

1. They are used to indicate that multiple characters should be considered a single "compound" character when extracting initials. For example, `Yuri` becomes `Y.`, but `{Yu}ri` becomes `Yu.`

   An important aspect of this use of braces is that it only applies to the first characters of a given name. As we'll see below, this has important implications for our parsing code, which must preserve braces at the beginning of given names, but can be more cavalier with braces in other positions.
2. Spaces and commas are ordinarily interpreted as name separators, rather than name components. Similarly, periods and hyphens usually have a special interpretation. All these characters can be stripped of their special meanings by putting them within braces.

In practice, it might be possible to insert other tokens (such as macros) into names as long as they either (a) are non-expandable or (b) expand into a series of tokens of the above enumerated types. However, in such cases it will probably be safer to declare the macro in question as either a text accent or a text symbol.

6.16.1 Text accents

Syntactically, a text accent is a macro that takes a single, undelimited argument, i.e, it has a "prototype" of `macro:#1->`. Semantically, the implication is that it takes a letter (the *base*) as an argument and produces a glyph that for certain purposes can be considered equivalent to the base (see the discussion of stem comparison on page 86).[3]

`\amsrefs@textaccents` This will contain a list of accent commands in standard LaTeX format (i.e., separated by the token `\do`). For example, after registering the `\"` and `\'` accents, it will contain

```
\do \"\do \'
```

```
1782 \let\amsrefs@textaccents\@empty
```

[3]Note that this is meant to be a pragmatic definition for the purposes of this package. No claim is made to greater generality.

\DeclareNameAccent *Arguments*:
#1 <- *accent*.

```
\def\DeclareNameAccent{%
    \@lappend\amsrefs@textaccents
}
```

Here are all the standard LaTeX accents, as well as a few nonstandard accents from the `mathscinet` package.

```
\DeclareNameAccent\"
\DeclareNameAccent\'
\DeclareNameAccent\.
\DeclareNameAccent\=
\DeclareNameAccent\^
\DeclareNameAccent\`
\DeclareNameAccent\~%
\DeclareNameAccent\b
\DeclareNameAccent\c
\DeclareNameAccent\d
\DeclareNameAccent\H
\DeclareNameAccent\k
\DeclareNameAccent\r
\DeclareNameAccent\t
\DeclareNameAccent\u
\DeclareNameAccent\v
```

From `mathscinet`:

```
\DeclareNameAccent\utilde
\DeclareNameAccent\uarc
\DeclareNameAccent\dudot
\DeclareNameAccent\lfhook
\DeclareNameAccent\udot
\DeclareNameAccent\polhk
\DeclareNameAccent\soft
```

`\etalchar` and `\etaltext` are sort of accent-like if you look at them in the right light.

```
\DeclareNameAccent\etalchar
%\DeclareNameAccent\etaltext
```

6.16.2 Text symbols

Syntactically, a text symbol is a macro with a empty parameter text, i.e., a prototype of `macro:->`. Semantically, it's a letter-like glyph that should not be considered equivalent to any other glyph or group of glyphs. In addition, it may exist in both upper- and lowercase variants, unlike text accents, where we consider the case to be an attribute of the base letter, not of the accent.[4]

\amsrefs@textsymbols This is analogous to `\amsrefs@textaccents` but a little more complicated due to the need to store lowercase equivalents. It consists of a list of double entries of the form

[4] As with text accents, this is not intended as a fully general definition.

```
\do \symbol \do \lcsymbol
```

which means that `\symbol` is a text symbol whose corresponding lowercase version is `\lcsymbol`. (Note that nothing is implied about whether `\symbol` is to be considered as uppercase or lowercase.) For example, in

```
\do \ae \do \ae \do \OE \do \oe
```

the first four tokens indicate that `\ae` is a text symbol with lowercase equivalent `\ae`, while the last four tokens indicate that `\OE` is a text symbol with lowercase equivalent `\oe`. This scheme is somewhat redundant, but pleasingly simple.

This also duplicates some of the information in `\@uclclist`, but it seems safer to do this than to modify `\@uclclist`.

```
\let\amsrefs@textsymbols\@empty
```

`\DeclareNameSymbol` *Arguments*:
`#1 <-` *symbol*.
`#2 <-` *lowercase*.

```
\def\DeclareNameSymbol#1#2{%
    \@lappend\amsrefs@textsymbols#1%
    \@lappend\amsrefs@textsymbols#2%
    \ifx#1#2\else
        \@lappend\amsrefs@textsymbols#2%
        \@lappend\amsrefs@textsymbols#2%
    \fi
}
```

Here are the standard LaTeX and `mathscinet` text symbols.

Note that `\i` and `\j` are anomalous in being syntactically like text symbols, but semantically more like text accents.

```
\DeclareNameSymbol\i\i
\DeclareNameSymbol\j\j
\DeclareNameSymbol\AE\ae
\DeclareNameSymbol\OE\oe
\DeclareNameSymbol\O\o
\DeclareNameSymbol\DH\dh
\DeclareNameSymbol\DJ\dj
\DeclareNameSymbol\L\l
\DeclareNameSymbol\NG\ng
\DeclareNameSymbol\SS\ss
\DeclareNameSymbol\TH\th
```

From `mathscinet`:

```
\DeclareNameSymbol\Dbar\dbar
\DeclareNameSymbol\lasp\lasp
\DeclareNameSymbol\rasp\rasp
\DeclareNameSymbol\cprime\cprime
\DeclareNameSymbol\cdprime\cdprime
\DeclareNameSymbol\bud\bud
\DeclareNameSymbol\cydot\cydot
```

~ can be considered a text symbol in much the same way that \etalchar can be considered an accent.

```
\DeclareNameSymbol~~%
```

6.16.3 \edef-like macros for names

The following macros all behave sort of like \edef, in the sense that

```
\X@edef\foo{name}
```

defines \foo to be the result of expanding name and applying a certain transformation to it.

\normalize@edef This converts accents in the name to a normalized form where the accent and its argument are surrounded by braces. E.g., after

```
\normalize@edef\cs{P\'olya}
```

\cs will contain P{\'o}lya. (This might result in a redundant layer of braces if the original text contained, say, "P{\'o}lya", but that's ok.) This lets us extract the first n characters from a name by using TEX's macro argument-gobbling mechanism without worrying that an accent will be separated from its base letter. As a bonus, it also replaces ties (~) by spaces.

```
\def\normalize@edef#1#2{%
    \begingroup
        \@apply\auto@protect\amsrefs@textsymbols
        \@apply\wrap@accent\amsrefs@textaccents
```

Redefine \@tabacckludge in case someone wants to use this with the inputenc package.

```
        \let\@tabacckludge\use@accent
        \let~\space
        \edef\@tempa{\def\@nx#1{#2}}%
    \@xp\endgroup
    \@tempa
}
```

\use@accent This is identical to \@nameuse except for the addition of the \string, which, as per ltoutenc.dtx, guards against the eventuality that something like ' might be active at the point of use. We don't expect to find a \bib in the middle of a tabbing environment (do we?) so we

```
\def\use@accent#1{\csname\string#1\endcsname}
```

\wrap@accent Here's a wrapper macro that causes an accent to become auto-wrapping. E.g., after \wrap@accent\', \'o will expand to {\'o}.

```
\def\wrap@accent#1{%
    \def#1##1{{\@nx#1##1}}%
}
```

\lc@edef This converts all the characters in a name to all lowercase, using the mapping defined by \amsrefs@textsymbols. So, after

```
\lc@edef\cs{P\'olya}
```

`\cs` will contain `p\'olya`. Note that accents are not wrapped and ties are passed through unmolested.

```
\def\lc@edef#1#2{%
    \begingroup
        \let\@tabacckludge\use@accent %%??
        \@apply\auto@protect\amsrefs@textaccents
        \@apply\lc@do\amsrefs@textsymbols
        \edef\@tempa{\lowercase{\def\@nx#1{#2}}}%
    \@xp\endgroup
    \@tempa
}
```

`\lc@do` This is a slighly more complicated wrapper macro than previous ones. The first argument is a text symbol; the second argument is the lowercase variant of the symbol. If they're the same (i.e., the first argument is a lowercase text symbol), we `\auto@protect` it. Otherwise we define the first symbol to expand to the second.

```
\def\lc@do#1\do#2{%
    \ifx#1#2%
        \auto@protect#1%
    \else
        \def#1{#2}%
    \fi
}
```

`\purge@edef` Removes accents and braces from a name and converts ties to spaces, leaving only letters, punctuation and text symbols. For example,

```
\lc@edef\cs{P{\'o}lya}
```

will put `Polya` in `\cs`.

```
\def\purge@edef#1#2{%
    \begingroup
        \@apply\auto@protect\amsrefs@textsymbols
        \let~\space
        \@apply\purge@accent\amsrefs@textaccents
        \let\@tabacckludge\@gobble
```

As mentioned above (page 71), `\i` and `\j` are semantically like text accents; hence, they require special treatment here.

```
        \def\i{i}%
        \def\j{j}%
        \edef\@tempa{#2}%
        \toks@\@emptytoks
        \@xp\purge@edef@ \@tempa \@nil
        \edef\@tempa{\def\@nx#1{\the\toks@}}%
    \@xp\endgroup
    \@tempa
}
```

\purge@edef@ Peek ahead so \purge@edef@a will know whether its argument was originally surrounded by braces.

```
\def\purge@edef@{%
    \futurelet\@let@token
    \purge@edef@a
}
```

\purge@edef@a Process a single "chunk" (i.e., one macro-argument's worth) of the name.

```
\def\purge@edef@a#1{%
```

If we've run into the \@nil terminator, we're done.

```
    \ifx\@let@token\@nil
        \let\@tempa\@empty
    \else
```

Otherwise, if the argument was originally surrounded by braces, process it recursively before processing the remainder of the token stream.

```
        \ifx\@let@token\bgroup
            \def\@tempa{%
                \purge@edef@ #1\@nil
                \purge@edef@
            }%
        \else
```

If the argument is a single unbracketed token, just copy it into the output.

```
            \add@toks@{#1}%
            \let\@tempa\purge@edef@
        \fi
    \fi
    \@tempa
}
```

\purge@accent This is similar to \wrap@accent but it removes the accent command (and possibly a layer of braces surrounding the accent's argument).

```
\def\purge@accent#1{%
    \def#1##1{##1}%
}
```

6.17 Name parsing

Parsing names is somewhat complicated because parts of the name can (in principle) be empty (G=given, S=surname, J=jr)):

`author={Doe, John, Jr.}`: G={John} S={Doe} J={Jr.}
`author={Doe, John}`: G={John} S={Doe} J={}
`author={Doe, , Jr.}`: G={} S={Doe} J={Jr.}
`author={Doe}`: G={} S={Doe} J={}
`author={, John, Jr.}`: G={John} S={} J={Jr.}
`author={, John}`: G={John} S={} J={}
`author={, , Jr.}`: G={} S={} J={Jr.}
`author={}`: G={} S={} J={}

Not all of these forms are legal, of course, but that's no excuse for not parsing them correctly.

We also want to be somewhat lenient about the placement of spaces:

`author={ Doe,John,Jr.}`: G=`{John}` S=`{Doe}` J=`{Jr.}`

However, because one must have some standards, we assume there are no spaces in the following positions in the input:

1. before periods,
2. before commas,
3. at the end of the name,
4. before or after hyphens.

Thus, we make no attempt to compensate for the misplaced spaces in examples like these:

`author={Doe , J ., Jr.  }`: G=`{J .}` S=`{Doe }` J=`{Jr.  }`
`author={Doe, J. - M.}`: G=`{J. - M.}` S=`{Doe}` J=`{}`

Also, unless we are generating initials, we don't try to normalize spaces *after* periods:

`author={Doe, J.M.}`: G=`{J.M.}` S=`{Doe}` J=`{}`
(not G=`{J. M.}`)

Finally, since we allow authors to group together characters that should be treated as a single unit, we need to be careful to preserve the author's markup in cases like these:

`author={Doe, {Yu}ri}`: G=`{{Yu}ri}` S=`{Doe}` J=`{}`
`author={Doe, {Yu}}`: G=`{{Yu}}` S=`{Doe}` J=`{}`

This is harder than it seems. For example, consider a naive implementation that uses delimited arguments to pull the name apart:

```
\def\parsename#1,#2\@nil{%
    \def\bib'surname{#1}%
    \def\bib'given{#2}%
}

\parsename Doe, {Yu}ri\@nil
```

Unfortunately, this results in the space after the comma becoming part of `\bib'given`: " `{Yu}ri`".

Our next thought would be to modify the definition slightly to trick TEX into gobbling the space:

```
\def\parsename#1,#2#3\@nil{%
    \def\bib'surname{#1}%
    \def\bib'given{#2#3}%
}
```

Now the space is gone, but—surprise!—so are the braces: "`Yuri`". In addition, this approach makes it difficult to handle empty name parts correctly.

To sidestep these problems, instead of blindly gobbling macro arguments, we use `\futurelet` to look ahead at certain strategic moments so we can take the appropriate action (see `\get@namepart@d`–`f`). We only really care about preserving braces at the start of names (page 69), which simplifies things somewhat.

`\name@split` `\name@split` parses a name into its three parts and stores them in `\bib'surname`, `\bib'given` and `\bib'jr`. If the `initials` option is in force, it also extracts the initials from the given name and stores them in `\bib'initials`.

It expects the name to be parsed to be terminated by `\@nil` and to contain at least three commas. Thus the usual way to invoke it is

`\name@split` ⟨*name*⟩`,,,\@nil`

`\name@split` just uses `\get@namepart` to peal off the surname and then passes control to `\name@split@given`. (Note the spiffy continuation-passing programming style.)

```
\def\name@split{%
    \get@namepart\bib'surname\name@split@given
}
```

`\name@split@given` Pretty much the same, *mutatis mutandis*...

```
\def\name@split@given{%
    \get@namepart\bib'given\name@split@jr
}
```

`\name@split@jr` And again...

```
\def\name@split@jr{%
    \get@namepart\bib'jr\name@split@finish
}
```

`\name@split@finish` We have all three parts now. Do some consistency checking, extract the initials from the given name, and then call `\@nilgobble` to remove anything (such as extra commas) left on the stack.

```
\def\name@split@finish{%
    \ifx\bib'surname\@empty \EmptyNameWarning \fi
```

Theoretically, we could try to check for uninverted names here, but only at the risk of producing spurious warnings when the name really does only have one part (`author={Arvind}`).

A possible solution: Now that we have the `inverted` attribute, we could issue a warning if the given name is empty and the family name contains a space. I'm sure someone could find valid input that would still generate a spurious warning, but this would take care of the most common cases. This bears more thinking about.

```
%%      \ifx\@empty\bib'given
%%          \NameCheck \bib'surname ??\@nil
%%      \else
        \extract@initials\bib'given
```

```
%%      \fi
    \@nilgobble
}
```

`\get@namepart` Now for the fun part. `\get@namepart` takes two arguments. The first (the destination) should be a control sequence; the second (the continuation) will normally also be a control sequence, though technically we only require that it be a single token. `\get@namepart` scans everything up to the next level-0 comma, places it in the destination, and then calls the continuation.

```
\def\get@namepart#1#2{%
```

Save the destination in `\toks@` and the continuation in `\@temptokena`. It's unfortunate that this trashes the previous contents of those token lists (as well as the contents of `\@tempa` later on), but preliminary attempts to rewrite the code to leave the calling environment unchanged were not encouraging.

```
    \toks@{#1}%
    \@temptokena{#2}%
    \get@namepart@a
}
```

`\get@namepart@a` Now peek ahead at the next token in the stream and call `\get@namepart@b` to examine it.

```
\def\get@namepart@a{%
    \futurelet\@let@token
    \get@namepart@b
}
```

`\get@namepart@b` If the next token is a space token, we want to delete it. Otherwise we're ready to read the name.

```
\def\get@namepart@b{%
    \ifx\@let@token\@sptoken
        \@xp\get@namepart@c
    \else
        \@xp\get@namepart@d
    \fi
}
```

`\get@namepart@c` The next token is a space; we delete it and restart `\get@namepart@a`, in case there are multiple spaces.

```
\def\get@namepart@c{%
    \after@deleting@token\get@namepart@a
}
```

`\get@namepart@d` We're at the beginning of the name part. However, there are still two special cases we have to watch out for. First, the next token might be a comma, meaning that this name part is empty. Second, the next token might be an open brace (`{`), which we have to be sure to copy into the destination. So, we peek ahead again before proceeding.

```
\def\get@namepart@d{%
    \futurelet\@let@token
    \get@namepart@e
}
```

`\get@namepart@e` If the next token is a comma, it means the name part is empty; so, we set the destination to an empty list and then arrange to execute the continuation after deleting the comma. Otherwise we call `\get@namepart@f` to read a non-empty name, leaving `\@let@token` undisturbed so that `\get@namepart@f` knows what's coming up.

```
\def\get@namepart@e{%
    \ifx\@let@token,%
        \@xp\let\the\toks@\@empty
        \edef\@tempa{%
            \@nx\after@deleting@token\the\@temptokena
        }%
        \@xp\@tempa
    \else
        \@xp\get@namepart@f
    \fi
}
```

`\get@namepart@f` We know whether or not the name begins with a brace, but we don't know if the corresponding group contains the entire name or only part of it. By reading the name as two arguments, we can handle all cases correctly.[5]

Note that the arguments are not expanded.

```
\def\get@namepart@f#1#2,{%
    \ifx\@let@token\bgroup
        \@xp\def\the\toks@{{#1}#2}%
    \else
        \@xp\def\the\toks@{#1#2}%
    \fi
    \the\@temptokena
}
```

`\EmptyNameWarning` Or translator or contributor or...

```
\def\EmptyNameWarning{\amsrefs@warning{Empty contributor name}}
```

6.18 Extracting initials

Extracting initials from the author's given name is tricky because of the numerous special cases that need to be handled. Consider the following examples, some of which are admittedly contrived:

`author={Arvind}`: I={}
`author={Bing, R H}`: I={R H}
`author={Harish, \'Etienne}`: I={É.}

[5] More or less. If the second argument is brace-delimited, the braces will be lost. But as mentioned above (page 76), we don't really care.

`author={Harish, \'E.}`: I={É.}
`author={Harish, \'{E}.}`: I={É.}
`author={Harish, {\'E}.}`: I={É.}
`author={Harish, \'E}`: I={É}
`author={Harish, \'Etienne-P\^{\i }erre}`: I={É.-P.}
`author={Jones, David}`: I={D.}
`author={Jones, David-Michael}`: I={D.-M.}
`author={Katzenbach, Nicholas {deB}elleville}`: I={N. deB.}
`author={Katzenbach, Nicholas deB.}`: I={N. deB.}
`author={Matiyasevich, {Yu}ri}`: I={Yu.}
`author={Matiyasevich, {Yu}}`: I={Yu}
`author={Matiyasevich, Yu.}`: I={Yu.}

When processing initials, we loosen our strictures on spaces inside the given name by not requiring spaces after periods and tolerating them around hyphens and after the name:

`author={Jones, D.M.}`: I={D. M.}
`author={Jones, David - Michael}`: I={D.-M.}
`author={Jones, David , Jr.}`: I={D.}

(Strictly speaking, only the support for the first of these examples was a deliberate design decision; the other two are side-effects of the implementation. In any case, toleration of these quirks is in no way an endorsement of them, especially since they may make it more difficult for third-party software to correctly process bibliography entries.)

6.18.1 The algorithm

As a running example, consider the following contrived input:

`\'E.-P\^{\i}erre J.K. M`

which we want to turn into "É.-P. J. K. M".

We precede by stages.

1. Normalize the name by surrounding accents and their arguments by braces:

   `{\'E}.-P{\^\i }erre J.K. M`

   We also replace `~`s by spaces at this stage.
2. Replace each hyphen (`-`) by "␣`\ini@hyphen`␣":

   `{\'E}. \ini@hyphen  P{\^\i }erre J.K. M`
3. Add a space after each period:

   `{\'E}.  \ini@hyphen  P{\^\i }erre J. K.  M`
4. Now we have the name as a list of space-separated components. (In our example, the components are "`{\'E}.`", "`\ini@hyphen`", "`P{\^\i }erre`", "`J.`", "`K.`", and "`M`".) We loop through the compoments and replace each one by its "initialized" form. There are four cases:
   (a) The component ends in a period. Copy it and add the token `~`. (In our example, these are the components "`{\'E}.`", "`J.`" and "`K.`".)

(b) The component consists of a single (possibly compound) character without a period. Again, copy it and add `~`. (In our example, this is the component "`M`".)
(c) The component is the token `\ini@hyphen`. Copy it.
(d) The component consts of two or more (possibly compound) characters without a period (e.g., "`P{\^\i }erre`"). Copy the first character and add the tokens `.~`.

5. The token list generated above will end with an unwanted `~`. Delete it.

The end result is

```
{\'E}.~\ini@hyphen P.~J.~K.~M
```

which, when typeset, does indeed produce "É.-P. J. K. M".[6]

6.18.2 The implementation

`\extract@initials` This is pretty straightforward.

```
\def\extract@initials#1{%
    \begingroup
        \auto@protect\ini@hyphen
        \auto@protect\nobreakspace
        \let~\relax
        \@apply\auto@protect\amsrefs@textsymbols
        \@apply\auto@protect\amsrefs@textaccents
        \normalize@edef\@tempa{#1}%
        \ifx\@tempa\@empty
        \else
```

It would be nice if `\process@hyphens` and `\process@dots` commuted, and they almost do. However, suppose you have the (admittedly contrived) name `Yu.-{Yu}`, which should be turned into "Yu.-Yu". If `\process@dots` is applied first, the braces around the second "Yu" get removed, so the output is "Yu.-Y.". (Even worse would be `P.-\'E`, which would produce "P.-*?*")

```
            \process@hyphens\@tempa
            \process@dots\@tempa
            \process@names\@tempa
            \@chomp\@tempa{~}%
        \fi
        \edef\@tempa{\def\@nx\bib'initials{\@tempa}}%
    \@xp\endgroup
    \@tempa
}
```

`\ini@hyphen` The `\unskip` removes the space at the end of a potential (and probable) preceding `~`, but leaves the `\nobreak` penalty.

```
\def\ini@hyphen{\unskip-\nobreak}
```

[6] Tying all the characters together is potentially undesirable when, as in the example, there are a large number of pieces in the given name.

`\process@hyphens` This follows the same general pattern as `\get@namepart`, but with an extra layer of grouping to avoid unwanted side-effects. Otherwise, it uses the same parsing techniques.

One difference is that there is no explicit continuation: instead, we iterate by repeatedly calling `\process@one@hyphen@d` until we run into the `\@nil` marker.

```
\def\process@hyphens#1{%
    \begingroup
        \toks@\@emptytoks
        \@xp\process@one@hyphen #1-\@nil
        \edef\@tempa{\the\toks@}%
```

Because of the `-` we have to stick in as a delimiter above, `\process@one@hyphen` will always generate unwanted code at the end of the name. We now delete it. (This also has the necessary side-effect of expanding the `\space` macros into space characters.)

```
        \@chomp\@tempa{ \ini@hyphen\space}%
        \edef\@tempa{\def\@nx#1{\@tempa}}%
    \@xp\endgroup
    \@tempa
}
```

`\process@one@hyphen` Cf. `\get@namepart@a`.

```
\def\process@one@hyphen{%
    \futurelet\@let@token
    \process@one@hyphen@a
}
```

`\process@one@hyphen@a` Cf. `\get@namepart@b` and `\extract@initial@a`.

The tests for `\@nil` and `-` here are purely to supply better error recovery. Without them, a hyphen at the end of the given name (.e.g, `author={Doe, John-}`) would produce a very mysterious error message. Since it's unlikely the hyphen really belongs there, we delete it, but we also issue a warning to the author. (It will still show up as part of the full given name, though.)

We borrow `\fsa@n` from `rkeyval` to keep track of the appropriate next action.

```
\def\process@one@hyphen@a{%
    \ifx\@let@token\@nil
        \let\fsa@n\@gobble
    \else
        \ifx\@let@token -%
            \TrailingHyphenWarning
            \let\fsa@n\process@one@hyphen@b
        \else
            \ifx\@let@token\@sptoken
                \let\fsa@n\process@one@hyphen@b
            \else
                \let\fsa@n\process@one@hyphen@c
```

```
            \fi
        \fi
    \fi
    \fsa@n
}
```

\process@one@hyphen@b Cf. \get@namepart@c.

```
\def\process@one@hyphen@b{%
    \after@deleting@token\process@one@hyphen
}
```

\process@one@hyphen@c Cf. \get@namepart@f.

```
\def\process@one@hyphen@c#1#2-{%
    \ifx\bgroup\@let@token
        \add@toks@{{#1}#2 \ini@hyphen\space}%
    \else
        \add@toks@{#1#2 \ini@hyphen\space}%
    \fi
    \futurelet\@let@token
    \process@one@hyphen@d
}
```

\process@one@hyphen@d Here we just check for \@nil and terminate if we detect it. Otherwise, we start over.

```
\def\process@one@hyphen@d{%
    \ifx\@let@token\@nil
        \@xp\@gobble
    \else
        \@xp\process@one@hyphen
    \fi
}
```

\TrailingHyphenWarning Or translator or contributor or...

```
\def\TrailingHyphenWarning{%
    \amsrefs@warning{Trailing hyphen deleted from name}%
}
```

\process@dots This is almost completely parallel to \process@hyphens.

```
\def\process@dots#1{%
    \begingroup
        \toks@\@emptytoks
        \@xp\process@one@dot #1.\@nil
        \edef\@tempa{\the\toks@}%
        \@chomp\@tempa{. }%
```

Since it's legitimate for names to end in periods, we might still have an unwanted space at the end of the name, so we delete it too.

```
        \@chomp\@tempa{ }%
        \edef\@tempa{\def\@nx#1{\@tempa}}%
```

```
    \@xp\endgroup
    \@tempa
}
```

\process@one@dot

```
\def\process@one@dot{%
    \futurelet\@let@token
    \process@one@dot@a
}
```

\process@one@dot@a This is a bit different from \process@one@hyphen@a since we expect names sometimes to end in a period—or even two periods—not least because of the . we add as a delimiter when invoking \process@one@dot.

```
\def\process@one@dot@a{%
    \ifx\@let@token .%
        \def\fsa@n{\after@deleting@token\process@bare@dot}%
    \else
        \ifx\@let@token\@sptoken
            \let\fsa@n\process@one@dot@b
        \else
            \let\fsa@n\process@one@dot@c
        \fi
    \fi
    \fsa@n
}
```

\process@bare@dot

```
\def\process@bare@dot{%
    \add@toks@{. }%
    \futurelet\@let@token
    \process@one@dot@d
}
```

\process@one@dot@b

```
\def\process@one@dot@b{%
    \after@deleting@token\process@one@dot
}
```

\process@one@dot@c

```
\def\process@one@dot@c#1#2.{%
    \ifx\bgroup\@let@token
        \add@toks@{{#1}#2. }%
    \else
        \add@toks@{#1#2. }%
    \fi
    \futurelet\@let@token
    \process@one@dot@d
}
```

\process@one@dot@d

```
\def\process@one@dot@d{%
    \ifx\@let@token\@nil
        \@xp\@gobble
    \else
        \@xp\process@one@dot
    \fi
}
```

\process@names This is very similar to \process@hyphens and \process@dots, but with a couple of twists, as noted below.

```
\def\process@names#1{%
    \begingroup
        \toks@\@emptytoks
        \@xp\extract@initial #1 \@nil
        \edef\@tempa{\def\@nx#1{\the\toks@}}%
    \@xp\endgroup
    \@tempa
}
```

\extract@initial Scan through the token stream replacing words by their initials until we hit the terminating $'_{11}$

```
\def\extract@initial{%
    \futurelet\@let@token
    \extract@initial@a
}
```

\extract@initial@a As with \process@one@hyphen@a, the test for $'_{11}$ here is purely to provide better recovery, this time in case the given name has a trailing space (.e.g, author={Doe, John }). But since we're just deleting whitespace, we don't bother issuing a warning.

```
\def\extract@initial@a{%
    \ifx\@let@token\@nil
        \let\fsa@n\@gobble
    \else
        \ifx\@let@token\@sptoken
            \let\fsa@n\extract@initial@b
        \else
            \let\fsa@n\extract@initial@c
        \fi
    \fi
    \fsa@n
}
```

\extract@initial@b

```
\def\extract@initial@b{%
    \after@deleting@token\extract@initial
}
```

\extract@initial@c Here, instead of just copying the name, we extract its initials and copy those.

```
\def\extract@initial@c#1#2 {%
    \ifx\@let@token\bgroup
```

Note that we double-brace the first argument to avoid having to test \@let@token again inside \@extract@initial.

```
        \@extract@initial {{#1}}#2\@nil
    \else
        \@extract@initial #1#2\@nil
    \fi
    \futurelet\@let@token
    \extract@initial@d
}
```

\extract@initial@d

```
\def\extract@initial@d{%
    \ifx\@let@token\@nil
        \@xp\@gobble
    \else
        \@xp\extract@initial
    \fi
}
```

\@extract@initial This handles the four cases mentioned on page 79.

```
\def\@extract@initial#1#2\@nil{%
    \ifx\ini@hyphen#1%
        \add@toks@{\ini@hyphen}%
    \else
        \in@{.\@nil}{#1#2\@nil}% Look for a period at the end of the name
        \ifin@
            \add@toks@{#1#2~}%
        \else
            \count@chars\@tempcnta{#1#2}%
            \ifnum\@tempcnta > \@ne
                \add@toks@{#1.~}%
            \else
                \add@toks@{#1~}%
            \fi
        \fi
    \fi
}
```

\count@chars This sets its first argument (which is assumed to be a count register) to the number of characters in the second argument. Compound characters are counted as a single character.

```
\def\count@chars#1#2{%
    \begingroup
        \@tempcnta\z@
        \@count@chars#2\@nil
```

```
        \edef\@tempb{#1=\the\@tempcnta\relax}%
    \@xp\endgroup
    \@tempb
}
```

\@count@chars

```
\def\@count@chars#1{%
    \ifx #1\@nil
    \else
        \advance\@tempcnta\@ne
        \@xp\@count@chars
    \fi
}
```

6.19 Generating alphabetic labels

6.19.1 The algorithm

Like Gaul, an alphabetic label is divided into three parts.

1. The author part. In the simplest case, this is formed by extracting the first character of each word of each last name of each author. Thus, if there were two authors with last names "Vaughan Williams" and "Tallis", the author part would be "VWT".

   If there are more than four authors, only the first three names are used, and a superscript "+" is appended to represent the elided names. Similarly, if an author name is "others", it is replaced by a superscript "+" and any following author names (of which there shouldn't be any) are ignored.

   Finally, if there is only one author and the author's last name consists of a single word, the first three characters of that name are used.
2. The year part. If the `y2k` option is in force, or if the year is less than 1901, the entire year is used. Otherwise the last two digits of the year are used.[7] The combination of author part and year part will be referred to as the *stem*.
3. The suffix. If two or more items have the same stems, a suffix consisting of a lowercase latin letter will be appended to each label to make it unique.

This third part is more subtle than it might first appear. First, case is ignored when comparing stems, so that, for example, "Ahl1999" and "AHL1999" are considered identical. Second, existing practice (in English, at least), is to ignore diacritics so that, for example, "Ahl1999" and "Ähl1999" are considered identical.

Note that when checking for duplicate stems, we assume that bibliography items appear sorted by label, which means that all items with the same stem will be adjacent. This means we can use the naive algorithm (check to see if the current item has the same stem as the previous item and, if so, append a suffix) to detect clashes. This sorting will be done automatically by `amsxport`, but the document author is responsible for ensuring the appropriate order if

[7] Years with more than 4 digits are not currently handled correctly. *Caveat lector.*

amsxport is not used. This is why it's an error to mix the alphabetic and citation-order options.

6.19.2 The implementation

```
\let\previous@stem\@empty
\let\current@stem\@empty
\let\previous@year\@empty
\let\current@year\@empty
```

\amsrefs@label@prefix

```
\let\amsrefs@label@prefix\@empty
```

\thebib

```
\def\thebib{\amsrefs@label@prefix\@arabic\c@bib}
```

\append@to@stem

```
\def\append@to@stem{\global\@concat\current@stem}
```

\generate@alphalabel

```
\def\generate@alphalabel{%
```

If the user supplied an explicit label field, we use it. Otherwise, we generate our own.

```
    \ifx\bib'label\@empty
        \begingroup
```

We begin by saving the previous stem and initializing the current stem to the empty string.

```
            \global\let\previous@stem\current@stem
            \global\let\current@stem\amsrefs@label@prefix
```

The list of primary contributors is available to us in \current@primary in the form

$$\texttt{\textbackslash name\{}\mathrm{Last}_1,\mathrm{First}_1\texttt{\}}\ \texttt{\textbackslash name\{}\mathrm{Last}_2,\mathrm{First}_2\texttt{\}}\ \ldots\ \texttt{\textbackslash name\{}\mathrm{Last}_n,\mathrm{First}_n\texttt{\}}$$

We will be executing this list multiple times with various definitions of \name. So the first thing we want to do is establish a safe environment and normalize the names.

```
            \@apply\auto@protect\amsrefs@textsymbols
            \@apply\auto@protect\amsrefs@textaccents
            \auto@protect\name
            \auto@protect\etaltext
            \normalize@edef\@tempa\current@primary
```

Now we count the number of authors in the list and invoke the appropriate macro to calculate the author part of the reference label.

```
            \get@numberof\@tempcnta\name\@tempa
            \calc@author@part
```

Next append the year part.

```
            \append@label@year
```

At this point, the `\current@stem` is complete and we're ready to determine what (if any) suffix is needed to disambiguate it from the previous label.

```
            \calc@alpha@suffix
```

We have all the pieces now. Arrange to end the current group and then define `\bib@label` in the enclosing group. (This keeps `\bib@label` from being defined outside the group started by `\bib@start`. This isn't strictly necessary, but it provides a bit of compartmentalization.)

```
            \edef\@tempa{%
                \def\@nx\bib'label{%
                    \current@stem
                    \alpha@label@suffix
                }%
            }%
        \@xp\endgroup
        \@tempa
    \fi
}
```

`\calc@author@part@`

```
\def\calc@author@part@{%
    \ifnum \@tempcnta = 1
        \@xp\@oneauthorlabel\@xp{\@tempa}%
    \else
        \@xp\@multiauthorlabel\@xp{\@tempa}%
    \fi
}
```

`\calc@author@part@short`

```
\def\calc@author@part@short{%
    \@xp\@multiauthorlabel\@xp{\@tempa}%
}%
```

`\@firstone` This extracts the first character from a properly prepared author name (i.e., one in which accents are properly wrapped).

```
\def\@firstone#1{\@car#1\@empty\@nil}
```

`\@firstthree` And this extracts the first three characters.

```
\def\@firstthree#1{\@carcube#1\@empty\@empty\@empty\@nil}
```

`\@nametoken`

```
\let\@nametoken\@firstone
```

`\hyph@to@space`

```
\def\hyph@to@space#1-{#1 \hyph@to@space}
```

`\@marknames` Since we have a ' with funny catcode already, let's use it (being able to easily put a space after the ' makes things easier).

```
\def\@marknames#1{%
    \@ifnotempty{#1}{\surround@names#1 ' }%
}
```

\surround@names

```
\def\surround@names#1 {%
    \ifx '#1%
    \else
        \@nx\@nametoken{#1}%
        \@xp\surround@names
    \fi
}
```

\extract@surnames

```
\def\extract@surnames#1#2{%
    \get@namepart\@tempb\@nilgobble #2,\@nil
    \edef\@tempb{\@nx\@marknames{\@xp\hyph@to@space\@tempb\@gobble-}}%
    \edef#1{\@tempb}%
}
```

\@oneauthorlabel This is the easy case.

```
\newcommand{\@oneauthorlabel}[1]{%
    \def\name##1{%
        \extract@surnames\@tempa{##1}%
        \get@numberof\@tempcnta\@nametoken\@tempa
        \ifnum \@tempcnta = 1
            \let\@nametoken\@firstthree
        \fi
        \append@to@stem{\@tempa}%
    }%
    #1%
}
```

\@threeauthors

```
\def\@threeauthors\name#1\name#2\name#3#4\@empty{%
    \name{#1}\name{#2}\name{#3}%
    \append@to@stem{\etalchar{+}}%
}
```

\@multiauthorlabel

```
\newcommand{\@multiauthorlabel}[1]{%
    \def\name##1{%
        \ifx\etaltext ##1%
            \def\@tempa{\@nx\etalchar{+}}%
            \let\name\@gobble
        \else
            \extract@surnames\@tempa{##1}%
        \fi
        \append@to@stem{\@tempa}%
```

```
    }%
    \ifnum \@tempcnta > 4 \@xp \@threeauthors \fi
    #1\@empty
}
```

\etalchar

```
\newcommand{\etalchar}[1]{$^{#1}$}
```

\year@short For alphanumeric labels, we want to extract the last 2 digits of the year. Here's a way to do that, assuming a 4-digit year.

```
\def\year@short#1#2#3#4\@nil{#3#4}
```

\append@label@year@

```
\def\append@label@year@{%
    \safe@set\@tempcnta\bib@year
    \edef\bib@citeyear{\the\@tempcnta}%
    \append@to@stem{%
        \ifx\bib@year\@empty
        \else
            \@xp\year@short \bib@citeyear \@nil
        \fi
    }%
}

\let\alpha@label@suffix\@empty

\newcount\alpha@suffix
\alpha@suffix\@ne
\let\@suffix@format\@alph
```

\calc@alpha@suffix

```
\def\calc@alpha@suffix{%
    \@tempswafalse
    \compare@stems\previous@stem\current@stem
    \ifsame@stems
```

Under the `alphabetic` option, `\previous@year` and `\current@year` will always be the same (namely, both will be empty), but including the test allows this code to work with the `author-year` option as well.

```
        \ifx\previous@year\current@year
            \@tempswatrue
        \fi
    \else
        \begingroup
            \let\name\@firstofone
            \@ifundefined{amsrefs@stem@\current@stem}{%
                \expandafter\gdef\csname amsrefs@stem@\current@stem\endcsname{}%
            }{%
                \DuplicateBibLabelWarning
            }%
```

```
2287        \endgroup
2288    \fi
2289    \if@tempswa
2290        \global\advance\alpha@suffix\@ne
2291        \edef\alpha@label@suffix{\@suffix@format\alpha@suffix}%
2292        \ifnum\alpha@suffix=\tw@
2293            \immediate\write\@auxout{%
2294                \string\ModifyBibLabel{\prev@citekey}%
2295            }%
2296        \fi
2297    \else
2298        \let\alpha@label@suffix\@empty
2299        \global\alpha@suffix\@ne
2300        \@xp\ifx \csname b@\current@citekey @suffix\endcsname \relax
2301        \else
2302            \edef\alpha@label@suffix{\@suffix@format\alpha@suffix}%
2303        \fi
2304    \fi
2305 }
```

\ifsame@stems

```
2306 \newif\ifsame@stems
```

\compare@stems

```
2307 \def\compare@stems#1#2{%
2308     \begingroup
2309         \purge@edef\@tempa{#1}%
2310         \purge@edef\@tempb{#2}%
2311         \lc@edef\@tempa{\@tempa}%
2312         \lc@edef\@tempb{\@tempb}%
2313         \ifx\@tempa\@tempb
2314             \def\@tempa{\same@stemstrue}%
2315         \else
2316             \def\@tempa{\same@stemsfalse}%
2317         \fi
2318     \@xp\endgroup
2319     \@tempa
2320 }
```

\ModifyBibLabel

```
2321 \def\ModifyBibLabel#1{%
2322     \global\@xp\let\csname b@#1@suffix\endcsname\@empty
2323 }
```

6.20 Generating short alphabetic labels

This style for alphabetic labels is somewhat simpler than the regular alphabetic style. The stem consists only of an author part without a year part. The author part is formed in the same way, except that even when there is only a single author with a one-word last name, only the first letter of the name is used,

not the first three. Finally, the suffix used to disambiguate identical stems is numeric rather than alphabetic.

See section 6.26.2 on page 108 for the implementation.

6.21 Formatting series

The `\PrintSeries` command prints a list of objects in series form. The essential idea is to produce something like "A, B, and C" when we are given three elements "A", "B", and "C", with suitable variations in the punctuation and other intervening material depending on the number of elements.

More precisely, we can envision `\PrintSeries` being called as

`\PrintSeries{`S`}{`i_1`}{`i_2`}{`i_3`}{`E`}{\do{`T_1`}` ... `\do{`T_n`}}`

where S and E are material to be interpolated before the start and after the end of the list, respectively, i_1, ..., i_3 are material to be interpolated between the elements, and the final argument is a list of indeterminate length where each element consists of a macro and its argument. If there are exactly two elements, i_1 is inserted between them; otherwise, i_2 is inserted between each pair of items except for the last pair, where i_3 is inserted. Thus,

n	output
1	$S\ T_1\ E$
2	$S\ T_1\ i_1\ T_2\ E$
3	$S\ T_1\ i_2\ T_2\ i_3\ T_3\ E$
4	$S\ T_1\ i_2\ T_2\ i_2\ T_3\ i_3\ T_4\ E$

and so forth. For example, a standard comma-separated list could be formatted by

```
\PrintSeries{}{ and }{, }{, and }{}{...}
```

That is the simple case but in practice there are additional complications. What if user-supplied line breaks have to be supported at the boundaries between elements? What if in addition to adding material between elements we also want to apply some handy function to each element (e.g., `\textsc`)? Even worse, what if we want the function to be different depending on the position of the element in the list? Indeed if this did not happen to be the case with the current application I would not have gone to the extra trouble of supporting it. But if it must be so, then the output that we need from a list `\do{A}\do{B}...` is

```
f0{A}
f0{A} p1 i1 f1{B}
f0{A} p2 i2 f2{B} p3 i3 f3{B}
```

and so on, where

- f_n is a macro taking one argument,
- p_n is punctuation—material that must precede a line break if one occurs at this boundary,
- i_n other interpolated material, as before.

To reduce the number of distinct required objects we decree that each element will get braces wrapped around it as a matter of course; then it is possible for f_1, f_2, f_3 to be assimilated onto the tail end of i_1, i_2, i_3. Since we also have to specify the macro that delimits the elements of the list, we end up with the following rather formidable signature:

```
\PrintSeries{\m} {f_0} {p_1}{i_1 f_1} {p_2}{i_2 f_2} {p_3}{i_3 f_3}
    {S} {\m{T_1}...\m{T_n}} {E}
```

and our comma-separated list example becomes

```
\PrintSeries{\do}{} {}{ and } {,}{ } {,}{ and } {}{...}{}
```

`\series@index` First we define a dedicated count register to be used in tracking the ordinal number of the item currently being processed.

```
\newcount\series@index
```

`\PrintSeries`

```
\def\PrintSeries#1#2#3#4#5#6#7#8{%
    \begingroup
        \def\series@add@a{#2}%
        \def\series@add@b{\SwapBreak{#3}#4}%
        \def\series@add@c{\SwapBreak{#5}#6}%
        \def\series@add@d{\SwapBreak{#7}#8}%
        \def\series@add@e{\SwapBreak{#7}}%
        \PrintSeries@a{#1}%
}
```

`\PrinteSeries@a` For `\PrintSeries@a` the first arg is the iterator function present in the list which is arg 3. Args 2 and 4 are extra material to be added before and after the list that may require the use of `\Plural` or `\SingularPlural`.

```
\def\PrintSeries@a#1#2#3#4{%
        \get@numberof\@tempcnta#1{#3}%
        \chardef\series@total=\@tempcnta
        \ifnum\series@total=\@ne
            \let\SingularPlural\@firstoftwo
        \else
            \let\SingularPlural\@secondoftwo
        \fi
        \series@index=\z@
        \let#1\series@add
        #2#3#4\relax
    \endgroup
}
```

`\series@add` This is the inner function called by `\PrintSeries` that carefully distributes all the material stored previously in `\series@add@...` macros.

Note that the handling of "et al." cases is somewhat hardcoded. This seemed preferable to adding yet another argument (or two!) to `\PrintSeries`.

```
\def\series@add#1{%
```

```
    \advance\series@index\@ne
    \ifx\etaltext#1\relax
        \ifnum\series@index=\tw@
            \def\@tempa{\space\SubEtal}%
        \else
            \def\@tempa{\series@add@e\space\SubEtal}%
        \fi
```

We assume there are fewer than 20,000 items in the list.

```
        \series@index\@MM
    \else
        \ifcase\series@index
        \or
```

Material before name 1:

```
            \let\@tempa\series@add@a
        \or
```

Material before name 2:

```
            \ifnum\series@total<\thr@@
                \let\@tempa\series@add@b
            \else
                \let\@tempa\series@add@c
            \fi
        \else
```

Material before names 3, 4, 5,...

```
            \ifnum\series@index=\series@total
                \let\@tempa\series@add@d
            \else
                \ifnum\series@index<\series@total
                    \let\@tempa\series@add@c
                \else
                    \let\@tempa\@gobble
                \fi
            \fi
        \fi
    \fi
    \@tempa{#1}%
}
```

`\SwapBreak` This takes a single argument, which should begin with a punctuation character, and conditionally appends it to the current horizontal list after removing any preceding whitespace. If there was also a penalty at the end of the hlist (presumed to be the result of a `\linebreak` at the end of a field value), it moves the penalty to *after* the argument.

> *Known bug:* `\SwapBreak` interferes with TEX's kerning mechanism. For example, consider a field value that ends with a "y" and that should have a comma automatically appended. `amsrefs` generates the equivalent of `y\SwapBreak{,}`, which results in "y," (no kern before the comma) rather than "y,". Unfortunately, fixing this would

likely require a disproportionate effort. In cases where the lack of kerning is unacceptable, a workaround is to add the punctuation mark to the field value manually. For example, `title={...y,}` would generate the equivalent of `y,\SwapBreak{,}`, which in turn would produce "y," since `\SwapBreak` is careful not to add duplicate punctuation.

```
\def\SwapBreak#1{%
    \relax\ifvmode\leavevmode\fi
    \@tempcnta\@MM
    \toks@{#1}%
```

First, remove any preceding glue. (There usually shouldn't be any of this.)

```
    \unskip
```

There might be also be kern, typically an italic correction left there by a previous `TextFontCommand` like `\textit`. But don't remove the special 1 sp kern used to mark the beginning of a bibliography entry.

Known bug: Sometimes we want to keep the italic correction.

```
    \ifnum \lastkern>\@ne \unkern \fi
```

And now look for a penalty and stash it in a safe place.

```
    \ifnum\lastpenalty=\z@
    \else
        \@tempcnta\lastpenalty
        \unpenalty
    \fi
```

Now we add the punctuation, *unless* one of the following conditions is true:

1. The last item on the horizontal list was a kern of 1 sp, indicating that we're at the very beginning of a bibliography item.
2. The current space factor is equal to the `\sfcode` of the puncutation mark we are adding, meaning that the mark is already on the list.
3. The current space factor is equal to the special value `\@nopunctsfcode`, meaning that `\nopunct` was specified.

This relies on distinct punctuation marks having distinct space factors, as established by our definition of `\frenchspacing`.

```
    \edef\@tempa{%
        \@nx\deferredquoteslogical
        \ifnum\lastkern=\@ne
        \else
            \ifnum\spacefactor=\sfcode\@xp\@xp\@xp`\@xp\@car\string#1)\@nil
            \else
                \ifnum\spacefactor=\@nopunctsfcode
                \else
                    \the\toks@
                \fi
            \fi
        \fi
```

```
        \@nx\deferredquotes
        \ifnum\@tempcnta=\@MM \else \penalty\number\@tempcnta\space \fi
        \ifnum\lastkern=\@ne \ignorespaces \fi
    }%
    \@tempa
}
```

`\Plural` `\SingularPlural` `\Plural` takes one argument and prints it if there were two or more elements in the current list. So, to get "editors" instead of "editor" after printing a list of editor names, write `editor\Plural{s}`.

`\SingularPlural` takes two arguments and prints the first if there was only one element, otherwise prints the second arg.

```
\newcommand{\SingularPlural}[2]{#1}
\newcommand{\Plural}{\SingularPlural{}}
```

6.22 Formatting names and series of names

Now that we have a general mechanism for formatting series, we can easily specialize to the common case of a comma-separated list of names. First we provide specifications for the three most common name formats.

`\setbib@nameLE` This sets a name in standard western uninverted order, e.g., "John Doe Jr." (The "LE" stands for little-endian.)

```
\BibSpec{nameLE}{
    +{}{}{given}
    +{}{\IfEmptyBibField{given}{}{ }}{surname}
    +{}{ }{jr}
}
```

`\setbib@nameBE` Big-endian order, as used for example in traditional Chinese, Japanese, Vietnamese, and Hungarian names: "Doe John". Big-endian formatting can be requested for name by setting the "inverted" property to "yes."

```
\BibSpec{nameBE}{
    +{}{}{surname}
    +{}{ }{given}
```

I don't know what should happen if there's a suffix, so I'm going to just leave it out for now (although I should probably issue a warning). I suspect that either (a) it never comes up or (b) if it does come up, there's no set standard for how it should be handled.

```
%    +{}{ }{jr}
}
```

`\setbib@nameinverted` Inverted western-style names: "Doe, John, Jr."

```
\BibSpec{nameinverted}{
    +{} {}  {surname}
    +{,}{ } {given}
    +{,}{ } {jr}
}
```

Incidentally, it would probably be cleaner if names had their own namespace like properties do, i.e., something like

```
\DefineSimpleKey{name}{given}
\DefineSimpleKey{name}{initials}
\DefineSimpleKey{name}{surname}
\DefineSimpleKey{name}{jr}
```

followed by

```
\NameSpec{nameLE}{...}
```

or

```
\BibSpec[name]{nameLE}{...}
```

But this seems a little extravagant at this stage, so I've decided to leave things as-is for now.

`\PrintNames` `\PrintNames` is a simplified interface to `\PrintSeries` that takes only the last three arguments:

`\PrintNames {`S`} {`E`} {\name{`T_1`}...\name{`T_n`}}`

The order of the last two arguments is reversed to make it moderately easier to use; cf. `\PrintEditorsA`, etc.

The first name in a series is treated differently than the other names in the `author-year` style, so we use a separate formatting macro for it.

```
\newcommand{\PrintNames}{%
    \@ifstar{\PrintNames@a\set@othername}{\PrintNames@a\set@firstname}%
}
```

`\PrintNames@a`

```
\newcommand{\PrintNames@a}[4]{%
    \PrintSeries{\name}
        {#1}
        {}{ and \set@othername}
        {,}{ \set@othername}
        {,}{ and \set@othername}
        {#2}{#4}{#3}%
}
```

`\set@firstname` By default, the first name is formatted in little-endian format. The `author-year` option changes this to inverted order.

```
\def\set@firstname#1{%
    \set@name{#1}\setbib@nameLE
}
```

`\set@othername` The rest of the names are set in little-endian format by default.

```
\def\set@othername#1{%
    \set@name{#1}\setbib@nameLE
}
```

`\set@name` Parse the name into its components and then pass control to `\set@name@a`, which will decide what format to use for the name.

```
\def\set@name#1{%
    \name@split#1,,,\@nil
    \set@name@a
}
```

`\set@namea` Use the requested format unless the `order` property has been set to "inverted."

```
\def\set@name@a#1{%
    \begingroup
        \get@current@properties
        \select@auxlanguage
        \def\@tempa{yes}%
        \ifx\@tempa\prop'inverted
            \setbib@nameBE
        \else
            #1%
        \fi
    \endgroup
}
```

`\PrintPrimary`

```
\def\PrintPrimary{%
    \ifx\current@primary\@empty
        \EmptyPrimaryWarning
    \else
        \print@primary\current@primary
    \fi
}
```

`\EmptyPrimaryWarning`

```
\def\EmptyPrimaryWarning{%
    \amsrefs@warning{No authors, editors or translators}%
}
```

`\PrintAuthors` The comparison of `\previous@primary` and `\current@primary` doesn't look at auxiliary properties (see also `\PrintEditorsA` and `\PrintTranslatorsA`). This is probably ok.

```
\newcommand{\PrintAuthors}[1]{%
    \ifx\previous@primary\current@primary
        \sameauthors\@empty
    \else
        \def\current@bibfield{\bib'author}%
        \PrintNames{}{}{#1}%
    \fi
}
```

`\sameauthors`

```
\newcommand{\sameauthors}[1]{\bysame#1}
```

\bysame

```
\def\bysame{%
    \leavevmode\hbox to3em{\hrulefill}\thinspace
    \kern\z@
}
```

\PrintNameList This just prints the names without any additional information.

```
\newcommand{\PrintNameList}{\PrintNames{}{}}
```

\PrintEditorsC

```
\newcommand{\PrintEditorsC}[1]{%
    \PrintNames{Edited by }{}{#1}%
}
```

\PrintEditorsA When we consider editor names we have to think about some further complications. First, for the case of a book where editor names are listed in place of author names, just copy the same style with a bit of added text at the end.

```
\newcommand{\PrintEditorsA}[1]{%
    \ifx\previous@primary\current@primary
        \sameauthors{(ed\Plural{s}.)}%
    \else
        \def\current@bibfield{\bib'editor}%
        \PrintNames{}{ (ed\Plural{s}.)}{#1}%
    \fi
    \erase@field\bib'editor
}
```

\PrintEditorsB

```
\newcommand{\PrintEditorsB}{%
    \PrintNames*{(}{\SwapBreak{,}~ed\Plural{s}.)}%
}
```

\PrintContributions

```
\newcommand{\PrintContributions}[1]{%
    \PrintSeries
        {\fld@elt}
        {\print@contribution}
        {}{ and \print@contribution}
        {,}{ \print@contribution}
        {,}{ and \print@contribution}{}{#1}{}%
}
```

\print@contribution

```
\newcommand{\print@contribution}[1]{%
    \in@={#1}%
    \ifin@
        \ifnum\series@index=\@ne with \fi
        \RestrictedSetKeys{}{bib}{%
```

```
            \bib@print@inner\setbib@contribution{\the\rsk@toks}%
        }{#1}%
    \else
        #1%
    \fi
}
```

\resolve@inner

```
\def\resolve@inner#1#2{%
    \in@={#2}%
    \ifin@
        \RestrictedSetKeys{}{bib}{#1{\the\rsk@toks}}{#2}%
    \else
        \@ifundefined{bi@#2}{%
            \XRefWarning{#2}%
        }{%
            #1{\csname bi@#2\endcsname}%
        }%
    \fi
}
```

\PrintConference

```
\def\PrintConference{%
    \resolve@inner{\bib@print@inner\setbib@conference}
}
```

\PrintConferenceDetails

```
\def\PrintConferenceDetails#1{%
    \ifx\@empty\bib'address
        \ifx\@empty\bib'date
        \else
            \PrintConferenceDetails@
        \fi
    \else
        \PrintConferenceDetails@
    \fi
}
```

\PrintConferenceDetails@

```
\def\PrintConferenceDetails@{%
    \ifnum\lastkern=\@ne\else\space\fi(\kern 1sp
    \ifx\@empty\bib'address
    \else
        \bib'address
    \fi
    \ifx\@empty\bib'date
    \else
        \SwapBreak{,}\space
        \print@date
```

```
    \fi
    )%\spacefactor\sfcode`\,%
}
```

\PrintBook

```
\def\PrintBook{%
    \resolve@inner{\bib@print@inner\setbib@innerbook}
}
```

\PrintReprint

```
\def\PrintReprint{%
    \resolve@inner{\bib@reprint}
}
```

\bib@reprint

```
\def\bib@reprint#1{%
    \begingroup
        #1\relax            % execute definitions locally
        \bib@resolve@xrefs
        \bib@field@patches
        \bib'setup
        \IfEmptyBibField{copula}{reprinted in}{\bib'copula} \nopunct
        \let\bib'language\@empty
        \setbib@book
    \endgroup
}
```

\PrintTranslation

```
\def\PrintTranslation{%
    \resolve@inner{\bib@translation}
}
```

\bib@translation

```
\def\bib@translation#1{%
    \begingroup
        #1\relax            % execute definitions locally
        \bib@resolve@xrefs
        \bib@field@patches
        \bib'setup
        \let\PrintPrimary\@empty
        \bib@append{;}{ % keep this space!
            \IfEmptyBibField{language}{English}{\bib'language} transl.%
            \IfEmptyBibField{pages}{ in \kern\@ne sp}{, }%
        }\bib'transition
        \let\bib'language\@empty
        \setbib@@
    \endgroup
}
```

\PrintTranslatorsC

```
\newcommand{\PrintTranslatorsC}[1]{%
    \PrintNames{translated by }{}{#1}%
}
```

\PrintTranslatorsA

```
\newcommand{\PrintTranslatorsA}[1]{%
    \ifx\previous@primary\current@primary
        \sameauthors{(trans.)}%
    \else
        \def\current@bibfield{\bib'translator}%
        \PrintNames{}{ (trans.)}{#1}%
    \fi
    \erase@field\bib'translator
}
```

\PrintTranslatorsB

```
\newcommand{\PrintTranslatorsB}[1]{
    \PrintNames*{(}{\SwapBreak{,}~tran\Plural{s}.)}%
}
```

Some special handling for "et alii" or "and others".

```
\DefineName{alii}{\etaltext}
\DefineName{others}{\etaltext}
```

\etaltext
\SubEtal

The Chicago Manual of Style suggests that it is slightly better not to italicize 'et al' and some other extremely common abbreviations inherited from Latin. (Compare 'etc'.)

```
\newcommand{\etaltext}{et al.}
\newcommand{\SubEtal}[1]{\etaltext}
```

6.23 The partial field

\print@partial

```
\newcommand{\print@partial}{%
    \resolve@inner{\bib@print@inner\setbib@partial}
}
```

6.24 Special formatting for other fields

\parenthesize

The \parenthesize function adds parentheses around its argument, calling \upn to optionally prevent italic parentheses from being used.

```
\newcommand{\parenthesize}[1]{%
    \leavevmode\push@bracket)\upn{(}#1\pop@bracket
}
```

\upn

By default, \upn is a no-op, meaning that this refinement lies dormant unless the upref package or other activation is done. (Probably better done via special fonts, anyway.)

```
\providecommand{\upn}[1]{#1}
```

\push@bracket
\pop@bracket
```
\let\bracket@stack\@empty

\def\push@bracket#1{%
    \xdef\bracket@stack{#1\bracket@stack}%
}

\def\pop@bracket{%
    \iffalse{\fi
    \@xp\pop@bracket@a\bracket@stack \@empty}%
}

\def\pop@bracket@a#1{%
    \leavevmode\/\upn{#1}%
    \xdef\bracket@stack{\iffalse}\fi
}
```

\bibquotes
```
\newcommand{\bibquotes}[1]{%
    \textquotedblleft#1%
    \gdef\deferredquotes{%
        \global\let\deferredquotes\@empty
        \textquotedblright
    }%
}
```

\mdash
\ndash
Cf. `textcmds`, where there's also a penalty added.
```
\providecommand{\mdash}{\textemdash}
\providecommand{\ndash}{\textendash}
```

\strip@MRprefix
```
\def\strip@MRprefix#1#2#3#4\@nil{%
    \def\@tempa{#1#2#3#4}%
    \if#1M%
        \if#2R%
            \def\@tempa{#3#4}%
        \fi
    \fi
}
```

\MR
```
\def\MR#1{%
    \relax\ifhmode\unskip\spacefactor3000 \space\fi
    \begingroup
        \strip@MRprefix#1\@nil
        \edef\@tempa{MR\@nx\MRhref{\@tempa}{\@tempa}}%
    \@xp\endgroup
    \@tempa
}
```

\MRhref For older versions of some AMS document classes, this patch is needed.

```
\providecommand{\MRhref}[2]{#1}
```

\PrintReviews Reviews are handled as a list to support the theoretical possibility of multiple reviews.

```
\newcommand{\PrintReviews}[1]{%
    \PrintSeries{\fld@elt}{}{,}{ }{,}{ }{,}{ }{}{#1}{}%
}
```

\PrintPartials

```
\newcommand{\PrintPartials}[1]{%
    \PrintSeries
        {\fld@elt}
        {\print@partial}
        {;}{ \print@partial}
        {;}{ \print@partial}
        {;}{ \print@partial}{}{#1}{}%
}
```

\PrintISBNs And similarly for ISBNs. There seem to be a few different situations where one book might have two different ISBN numbers. Here are the ones I know of so far [mjd,2002-02-18]: separate ISBN numbers for hardback and paperback; separate ISBN numbers for U.S. edition and European edition.

```
\newcommand{\PrintISBNs}[1]{%
    \PrintSeries{\fld@elt}{}{,}{ }{,}{ }{,}{ }{ISBN }{#1}{}%
}
```

\voltext

```
\newcommand{\voltext}{\IfEmptyBibField{series}{Vol.~}{vol.~}}
```

\issuetext

```
\newcommand{\issuetext}{no.~}
```

\DashPages Scan the contents of a page value to see if it is a single page. Presence of \ndash or hyphen is taken to mean no. Probably should test also for spaces and commas. [mjd,2000/01/24]

```
\newcommand{\DashPages}[1]{%
    p\pp@scan@a#1@\ndash p@\ndash{\pp@scan#1@-p@-{}\@nil}\@nil.~#1%
}

\def\pp@scan#1-#2@-#3#4\@nil{#3}

\def\pp@scan@a#1\ndash#2@\ndash#3#4\@nil{#3}
```

\eprintpages If we have eprint info and pages info and no journal name, the pages information is presumably the number of pages in the eprint.

```
\newcommand{\eprintpages}[1]{%
    #1\IfEmptyBibField{eprint}{}{\IfEmptyBibField{journal}{ pp.}{}}%
}
```

\PrintThesisType

```
\def\PrintThesisType#1{%
    \thesis@type#1?\@nil{#1}%
}

\def\thesis@type#1#2\@nil#3{%
    \ifx p#1%
        Ph.D. Thesis%
    \else
        \ifx m#1%
            Master's Thesis%
        \else
            #3%
        \fi
    \fi
}
```

\PrintDOI Perhaps need to add allowbreak penalties at the parentheses in a DOI. Also what about prohibiting a break after the leading S?

```
\newcommand{\PrintDOI}[1]{%
    DOI #1%
}
```

\PrintDatePV Print date in different forms depending on DOI and volume information.

```
\newcommand{\PrintDatePV}[1]{%
    \IfEmptyBibField{doi}{%
        \let\@tempa\PrintDate
    }{%
        \IfEmptyBibField{volume}{%
            \let\@tempa\PrintDatePosted
        }{%
            \let\@tempa\PrintDate
        }%
    }%
    \@tempa{#1}%
}
```

\PrintDate The intent is to handle variations such as `1987`, `August 1987`, `1987-08`, and `1987-08-14`. If the month is present, print August or Aug. or 08 or nothing, at the behest of the bib style.

We've taken some special care to parse out the date info ahead of time, so this function just discards arg 1 and uses the already-parsed value.

```
\newcommand{\PrintDate}[1]{(\print@date)}
```

\PrintDateB The same, but without the parentheses.

```
\newcommand{\PrintDateB}[1]{\print@date}
```

\PrintDateField

```
2715 \newcommand{\PrintDateField}[1]{%
2716     \@ifempty{#1}{}{%
2717         \begingroup
2718             \bib@parsedate{#1}%
2719             \print@full@date
2720         \endgroup
2721     }%
2722 }
```

\print@full@date

```
2723 \def\print@full@date{%
2724     \ifx\bib@month\@empty
2725     \else
2726         \print@month@day
2727     \fi
2728     \bib@year
2729 }
```

\print@date

```
2730 \let\print@date\print@full@date
```

\print@month@day

```
2731 \def\print@month@day{%
2732     \bib@monthname
2733     \ifx\@empty\bib@day \else \nobreakspace\number 0\bib@day,\fi
2734     \space
2735 }
```

\bib@monthname With the Babel package, month names for a given language are typically available in a macro \month@*language*:

```
\def\month@german{\ifcase\month\or
  Januar\or Februar\or M\"arz\or April\or Mai\or Juni\or
  Juli\or August\or September\or Oktober\or November\or Dezember\fi}
```

However this is not true for English.

```
2736 \newcommand{\bib@monthname}{%
2737     \ifcase 0\bib@month
2738     \or January\or February\or March\or April\or May\or June\or
2739       July\or August\or September\or October\or November\or December\or
2740       Winter\or Spring\or Summer\or Fall\else Unknown Month%
2741     \fi
2742 }
```

\PrintYear You can use \PrintYear if you want to suppress month/day even when supplied in the data.

```
2743 \newcommand{\PrintYear}[1]{\bib@year}
```

\PrintDatePosted This one is special for AMS use.

```
2744 \newcommand{\PrintDatePosted}[1]{\unskip,  posted on \print@date}
```

\PrintEdition

```
\newcommand{\PrintEdition}[1]{%
    \afterassignment\print@edition
    \count@ 0#1\relax\@nil
}
```

\print@edition If the number assignment swept up all the contents, produce a cardinal number from \count@.

```
\def\print@edition#1#2\@nil{%
    \ifx\relax#1\relax
        \ifnum\count@>\z@
            \CardinalNumeric\count@
        \else
            ??th%
        \fi
        \ \editiontext
    \else
        \ifnum \count@>\z@ \number\count@ \fi
        #1#2\relax
    \fi
}
```

\editiontext

```
\newcommand{\editiontext}{ed.}
```

\CardinalNumber

```
\newcommand{\CardinalNumeric}[1]{%
    \number#1\relax
    \if
        \ifnum#1<14
            \ifnum#1>\thr@@ T\else F\fi
        \else
            F%
        \fi
        T%
            th%
    \else
        \@xp\keep@last@digit\@xp#1\number#1\relax
        \ifcase#1th\or st\or nd\or rd\else th\fi
    \fi
}
```

\keep@last@digit

```
\def\keep@last@digit#1#2{%
    \ifx\relax#2%
        \@xp\@gobbletwo
    \else
        #1=#2\relax
    \fi
```

```
    \keep@last@digit#1%
}
```

\SentenceSpace Note how careful we are here to preserve \frenchspacing.

```
\newcommand{\SentenceSpace}{\relax\ifhmode\spacefactor`\. \fi}
```

\eprint For now, this does nothing. Could do a url/hyperlink or something.

```
\newcommand{\eprint}[1]{\url{#1}}
```

The www.arXiv.org recommendations for citing their eprints are found at http://xxx.lanl.gov/help/faq/references, including these examples:

```
arXiv:hep-th/9910001
arXiv:math.AT/9910001
arXiv:physics.acc-ph/9911027
```

6.25 BibTEX support

\bibliographystyle Disable \bibliographystyle since we're going to handle that behind the scenes.

```
\let\bibliographystyle\@gobble
```

\bibtex@style

```
\def\bibtex@style{amsrn}
```

```
\AtBeginDocument{
    \if@filesw
        \immediate\write\@auxout{\string\bibstyle{\bibtex@style}}%
    \fi
}
```

6.26 Implementing package options

6.26.1 The alphabetic option

```
\IfOption{alphabetic}{%
    \def\bibtex@style{amsra}%
    \let\process@citelist\process@citelist@unsorted
    \amsrefs@option@alphabetic
}{}
```

6.26.2 The shortalphabetic option

```
\IfOption{shortalphabetic}{%
    \def\bibtex@style{amsrs}%
    \let\process@citelist\process@citelist@unsorted
    \amsrefs@option@shortalphabetic
}{}
```

6.26.3 The backrefs option

Rather than checking for the backrefs option *per se*, we check to see if the backref package is loaded. This accomodates authors who load the backref package explicitly but do not pass the backrefs option to amsrefs.

```
\AtBeginDocument{%
```

```
    \@ifpackageloaded{backref}{%
        \let\PrintBackRefs\print@backrefs
        \let\BackCite\back@cite
```

The `backref` package uses `\@starttoc` inside `\thebibliography` to open and read the `.brf` file. We could do something similar with `\biblist`, but it seems cleaner to use `\AtBeginDocument`. Unfortunately, `amsart` redefines `\@starttoc` in a way that interacts badly with this use. So, we inline the relevant parts of `\@starttoc` here. (The group and `\makeatletter` are unnecessary at present, but I'll leave them in as future-proofing.)

```
        \begingroup
            \makeatletter
            \@input{\jobname.brf}%
            \if@filesw
                \newwrite\tf@brf
                \immediate\openout\tf@brf \jobname.brf\relax
            \fi
        \endgroup
    }{}%
}
```

6.26.4 The `citation-order` option

```
\IfOption{citation-order}{%
    \IfOption{alphabetic}{%
        \amsrefs@warning@nl{%
            The citation-order and alphabetic options are
            incompatible%
        }%
    }{
        \def\bibtex@style{amsru}%
    }
}{}
```

6.26.5 The `initials` option

```
\IfOption{initials}{% TRUE:
    \BibSpec{nameLE}{
        +{}{}{initials}
        +{}{\IfEmptyBibField{initials}{}{ }}{surname}
        +{}{ }{jr}
    }

    \BibSpec{nameBE}{
        +{}{}{surname}
        +{}{ }{initials}
    %    +{}{ }{jr}
    }

    \BibSpec{nameinverted}{
        +{} {}  {surname}
        +{,}{ } {initials}
```

```
        +{,}{ } {jr}
    }
}{% initials? FALSE:
%    \let\extract@initials\@gobble
} % end conditional code for initials option
```

6.26.6 The logical-quotes option

\deferredquotes

```
\let\deferredquotes\@empty
```

\deferredquoteslogical

```
\IfOption{logical-quotes}{%
    \def\deferredquoteslogical{\deferredquotes}%
}{%
    \let\deferredquoteslogical\relax
}
```

6.26.7 The non-compressed-cites option

```
\IfOption{non-compressed-cites}{%
    \let\cite@compress\cite@print
}{}
```

6.26.8 The non-sorted-cites option

```
\IfOption{non-sorted-cites}{%
    \let\process@citelist\process@citelist@unsorted
}{}
```

6.26.9 The short-journals option

```
\IfOption{short-journals}{%
    \renewcommand{\DefineJournal}[4]{%
        \bib*{#1}{periodical}{
            issn={#2},
            journal={#3},
        }%
    }
}{}
```

6.26.10 The short-publishers option

```
\IfOption{short-publishers}{%
    \renewcommand{\DefinePublisher}[4]{%
        \bib*{#1}{publisher}{%
            publisher={#2},%
```

Maybe `short-publishers` should suppress the `address`? Or is that a separate option? I sense a combinatorial explosion coming on....

```
            address={#4},
        }%
    }%
}{}
```

6.26.11 The short-months option

```
\IfOption{short-months}{%
    \renewcommand{\bib@monthname}{%
        \ifcase 0\bib@month
        \or Jan.\or Feb.\or Mar.\or Apr.\or May\or June\or
          July\or Aug.\or Sep.\or Oct.\or Nov.\or Dec.\or
          Winter\or Spring\or Summer\or Fall\else Unknown Month%
        \fi
    }%
}{}
```

6.26.12 The y2k option

```
\IfOption{y2k}{%
    \IfOption{alphabetic}{%
        \def\year@short#1\@nil{#1}%
        \def\bibtex@style{amsry}%
    }{%
        \amsrefs@warning@nl{%
            The y2k option can only be used with the^^J%
            alphabetic option%
        }%
}
}{}
```

6.26.13 The bibtex-style option

```
\IfOption{bibtex-style}{%
    \RequirePackage{amsbst}
}{}
```

6.26.14 The msc-links option

```
\IfOption{msc-links}{%
```

Unless you're using pdfTEX, links cannot be broken across lines, which causes problems for long-form MR numbers such as "MR2149145 (2006d:01012)". To mitigate the problem, we manually break such numbers into two separate links.

```
    \@ifundefined{href}{}{%
        \def\parse@MR#1 (#2)#3\@nil{%
            \def\MR@url{http://www.ams.org/mathscinet-getitem?mr=#1}%
            \def\@tempd{#1}%
            \def\@tempe{#2}%
        }%
        \def\MRhref#1#2{%
            \begingroup
                \parse@MR#1 ()\@empty\@nil%
                \href{\MR@url}{\@tempd\vphantom{()}}%
                \ifx\@tempe\@empty
                \else
                    \ \href{\MR@url}{(\@tempe)}%
                \fi
            \endgroup
        }%
    }{}
```

```
}{}
```

6.26.15 The author-year option

Here ends the amsrefs package, unless the author-year option is in effect; then we want to use some different bibspecs.

```
\IfOption{author-year}{}{\PopCatcodes \endinput}
```

\@biblistsetup

```
\renewcommand{\@biblistsetup}[1]{%
    \RestrictedSetKeys{}{biblist}{\the\rsk@toks}{#1}%
    \rkvIfEmpty{biblist}{prefix}{}{%
        \amsrefs@warning{%
            The ‘prefix’ biblist option cannot be used\MessageBreak
            with the author-year option.%
        }%
    }%
    \rkvIfEmpty{biblist}{labels}{}{%
        \amsrefs@warning{%
            The ‘labels’ biblist option cannot be used\MessageBreak
            with the author-year option.%
        }%
    }%
}
```

\generate@label

```
\def\generate@label{%
```

If the user supplied an explicit label field, we use it. Otherwise, we generate our own.

```
    \ifx\bib’label\@empty
        \begingroup
```

We begin by saving the previous stem and initializing the current stem to the empty string.

```
            \global\let\previous@stem\current@stem
            \global\let\current@stem\@empty
            \global\let\previous@year\current@year
            \global\let\current@year\bib@year
```

The list of primary contributors is available to us in \current@primary in the form

$$\texttt{\textbackslash name\{}\mathrm{Last}_1,\mathrm{First}_1\texttt{\}}\ \texttt{\textbackslash name\{}\mathrm{Last}_2,\mathrm{First}_2\texttt{\}}\ \ldots\ \texttt{\textbackslash name\{}\mathrm{Last}_n,\mathrm{First}_n\texttt{\}}$$

We will be executing this list multiple times with various definitions of \name. So the first thing we want to do is establish a safe environment and normalize the names.

```
            \@apply\auto@protect\amsrefs@textsymbols
            \@apply\auto@protect\amsrefs@textaccents
            \def\name##1{\@nx\name{\lnscan@a##1,\@nil}}%
            \auto@protect\etaltext
            \normalize@edef\current@stem{\current@primary}%
```

```
            \xdef\current@stem{\current@stem}%
```

At this point, the `\current@stem` is complete and we're ready to determine what (if any) suffix is needed to disambiguate it from the previous label.

```
            \calc@alpha@suffix
```

We have all the pieces now. Arrange to end the current group and then define `\bib@label` in the enclosing group. (This keeps `\bib@label` from being defined outside the group started by `\bib@start`. This isn't strictly necessary, but it provides a bit of compartmentalization.)

```
            \edef\@tempa{%
                \def\@nx\cite@label{\current@stem}%
                \def\@nx\bib@label@year{%
                    \current@year
                    \alpha@label@suffix
                }%
            }
        \@xp\endgroup
        \@tempa
    \fi
}
```

`\lnscan@a`

```
\def\lnscan@a#1,#2\@nil{#1}
```

`\citesel@author`

```
\def\citesel@author#1#2#3#4#5{\PrintCiteNames{#3}}
```

`\citesel@authoryear`

```
\def\citesel@authoryear#1#2#3#4#5{\PrintCNY{#3}{#4}}
```

`\citesel@object`

```
\def\citesel@object#1#2#3#4#5{\PrintCiteNames{#3} \citeleft#4}
```

`\citesel`

```
\let\citesel\citesel@authoryear
```

`\citeleft`

```
\def\citeleft{(}%
```

`\citeright`

```
\def\citeright{)}%
```

`\@citeleft`

```
\def\@citeleft{\ifx\citesel\citesel@object\else\citeleft\fi}%
```

`\citepunct`

```
\def\citepunct{; }
```

\BibLabel

```
\def\BibLabel{%
    \Hy@raisedlink{\hyper@anchorstart{cite.\CurrentBib}\relax\hyper@anchorend}%
}
```

\process@citelist

```
\let\process@citelist\process@citelist@unsorted
```

\ycite

```
\DeclareRobustCommand{\ycite}[1]{%
    \star@{\cite@a\citesel@year{#1}}{}%
}
```

\ycites

```
\DeclareRobustCommand{\ycites}[1]{%
    \begingroup
        \def\citepunct{, }%
        \let\citesel\citesel@year
        \cites{#1}%
    \endgroup
}
```

\citeyear

```
\DeclareRobustCommand{\citeyear}[1]{%
    \begingroup
        \let\citeleft\@empty
        \let\citeright\@empty
        \star@{\cite@a\citesel@year{#1}}{}%
    \endgroup
}
```

\ocite

```
\DeclareRobustCommand{\ocite}[1]{%
    \star@{\cite@a\citesel@object{#1}}{}%
}
```

\ocites

```
\DeclareRobustCommand{\ocites}[1]{%
    \begingroup
        \let\@citelist\@ocitelist
        \cites{#1}%
    \endgroup
}
```

\ocitelist

```
\def\@ocitelist#1{%
    \PrintSeries{\InnerCite}%
        {\ocite}%
        {}{ and \ocite}%
```

For three or more names: print 'et al' instead of the last name. Have to putz around with the space factor a bit or the comma between name and year will not be applied.

```
        {,}{ \ocite}%
        {,}{ and \ocite}%
        {}%
        {#1}%
        {}%
}
```

\citeauthor

```
\DeclareRobustCommand{\citeauthor}[1]{%
    \star@{\cite@a\citesel@author{#1}}{}%
}
```

\citeauthory

```
\DeclareRobustCommand{\citeauthory}[1]{%
    \citeauthor{#1} \ycite{#1}%
}
```

\fullcite

```
\DeclareRobustCommand{\fullcite}[1]{%
    \begingroup
        \let\print@citenames\CiteNamesFull
        \star@{\cite@a\citesel@authoryear{#1}}{}%
    \endgroup
}
```

\fullocite

```
\DeclareRobustCommand{\fullocite}[1]{%
    \begingroup
        \let\print@citenames\CiteNamesFull
        \star@{\cite@a\citesel@object{#1}}{}%
    \endgroup
}
```

Invert the first author's name.

```
\def\set@firstname#1{%
    \set@name{#1}\setbib@nameinverted
}
```

\PrintCNY

```
\def\PrintCNY#1#2{%
    \PrintCiteNames{#1}%
    \@ifnotempty{#2}{\@addpunct{,} #2}%
}
```

\PrintCiteNames

```
\def\PrintCiteNames#1{%
    \leavevmode
    \def\@tempa{#1}%
    \ifx\@tempa\prev@names
    \else
        \gdef\prev@names{#1}%
        \@xp\ifx\@car#1.\@nil\CitePrintUndefined
            #1\relax
        \else
            \print@citenames{#1}%
        \fi
    \fi
}
```

\CiteNames

```
\newcommand{\CiteNames}[1]{%
    \PrintSeries{\name}%
        {}%
        {}{ and }%
```

For three or more names: print 'et al' instead of the last name. Have to putz around with the space factor a bit or the comma between name and year will not be applied.

```
        {}{\@gobble}%
        {}{ \etaltext\@\@gobble}%
        {}%
        {#1}%
        {}%
}
```

\print@citenames

```
\let\print@citenames\CiteNames
```

\CiteNamesFull

```
\newcommand{\CiteNamesFull}[1]{%
    \PrintSeries{\name}%
        {}%
        {}{ and }%
```

For three or more names: print 'et al' instead of the last name. Have to putz around with the space factor a bit or the comma between name and year will not be applied.

```
        {,}{ }%
        {,}{ and }%
        {}%
        {#1}%
        {}%
}
```

\PrintDate No parentheses around the year.

```
\renewcommand{\PrintDate}[1]{\bib@label@year}
```

\print@date Only print the year, not the month or day.

```
\def\print@date{%
    \IfEmptyBibField{date}{%
        \IfEmptyBibField{year}{\BibField{status}}{\bib@year}%
    }{%
        \bib@year
    }%
}

\BibSpec{article}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { \PrintDate}                  {date}
    +{.} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintContributions}         {contribution}
    +{.} { \PrintPartials}              {partial}
    +{,} { }                            {journal}
    +{}  { \textbf}                     {volume}
    +{,} { \issuetext}                  {number}
    +{,} { \eprintpages}                {pages}
    +{,} { }                            {status}
    +{,} { \PrintDOI}                   {doi}
    +{,} { available at \eprint}        {eprint}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{book}{%
    +{}  {\PrintPrimary}                {transition}
    +{.} { \PrintDate}                  {date}
    +{.} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintEdition}               {edition}
    +{}  { \PrintEditorsB}              {editor}
    +{,} { \PrintTranslatorsC}          {translator}
    +{,} { \PrintContributions}         {contribution}
    +{,} { }                            {series}
    +{,} { \voltext}                    {volume}
    +{,} { }                            {publisher}
    +{,} { }                            {organization}
    +{,} { }                            {address}
```

```
    +{,} { }                            {status}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{collection.article}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { \PrintDate}                  {date}
    +{.} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintContributions}         {contribution}
    +{,} { \PrintConference}            {conference}
    +{}  {\PrintBook}                   {book}
    +{,} { }                            {booktitle}
    +{,} { pp.~}                        {pages}
    +{,} { }                            {status}
    +{,} { \PrintDOI}                   {doi}
    +{,} { available at \eprint}        {eprint}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{report}{%
    +{}  {\PrintPrimary}                {transition}
    +{.} { \PrintDate}                  {date}
    +{.} { \textit}                     {title}
    +{.} { }                            {part}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintEdition}               {edition}
    +{,} { \PrintContributions}         {contribution}
    +{,} { Technical Report }           {number}
    +{,} { }                            {series}
    +{,} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \eprint}                     {eprint}
    +{,} { }                            {status}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
```

```
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{thesis}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { \PrintDate}                  {date}
    +{.} { \textit}                     {title}
    +{:} { \textit}                     {subtitle}
    +{,} { \PrintThesisType}            {type}
    +{,} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \eprint}                     {eprint}
    +{,} { }                            {status}
    +{}  { \parenthesize}               {language}
    +{}  { \PrintTranslation}           {translation}
    +{;} { \PrintReprint}               {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
    +{}  {\SentenceSpace \PrintReviews} {review}
}

\BibSpec{webpage}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { \PrintDate}                  {date}
    +{.} { \emph}                       {title}
    +{:} { \emph}                       {subtitle}
    +{,} { \url}                        {url}
    +{.} { Accessed \PrintDateField}    {accessdate}
    +{.} { }                            {note}
    +{.} {}                             {transition}
}

\PopCatcodes
⟨/pkg⟩
```

6.27 The amsbst package

```
⟨*bst⟩

\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesPackage{amsbst}[2013/01/16 v2.12]
%\RequirePackage{amsrefs}[2004/03/29]

\BibSpec{article}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { }                            {title}
    +{.} { }                            {part}
    +{:} { }                            {subtitle}
    +{.} { \PrintContributions}         {contribution}
    +{.} { \PrintPartials}              {partial}
    +{.} { \emph}                       {journal}
    +{}  { }                            {volume}
    +{}  { \parenthesize}               {number}
```

```
    +{:} {}                              {pages}
    +{,} { \PrintDateB}                  {date}
    +{,} { }                             {status}
    +{.} { \PrintTranslation}            {translation}
    +{.} { Reprinted in \PrintReprint}   {reprint}
    +{.} { }                             {note}
    +{.} {}                              {transition}
}

\BibSpec{partial}{%
    +{}  {}                              {part}
    +{:} { }                             {subtitle}
    +{.} { \PrintContributions}          {contribution}
    +{.} { \emph}                        {journal}
    +{}  { }                             {volume}
    +{}  { \parenthesize}                {number}
    +{:} {}                              {pages}
    +{,} { \PrintDateB}                  {date}
}

\BibSpec{book}{%
    +{}  {\PrintPrimary}                 {transition}
    +{.} { \emph}                        {title}
    +{.} { }                             {part}
    +{:} { \emph}                        {subtitle}
    +{.} { }                             {series}
    +{,} { \voltext}                     {volume}
    +{.} { Edited by \PrintNameList}     {editor}
    +{.} { Translated by \PrintNameList}{translator}
    +{.} { \PrintContributions}          {contribution}
    +{.} { }                             {publisher}
    +{.} { }                             {organization}
    +{,} { }                             {address}
    +{,} { \PrintEdition}                {edition}
    +{,} { \PrintDateB}                  {date}
    +{.} { }                             {note}
    +{.} {}                              {transition}
    +{.} { \PrintTranslation}            {translation}
    +{.} { Reprinted in \PrintReprint}   {reprint}
    +{.} {}                              {transition}
}

\BibSpec{collection.article}{%
    +{}  {\PrintAuthors}                 {author}
    +{.} { }                             {title}
    +{.} { }                             {part}
    +{:} { }                             {subtitle}
    +{.} { \PrintContributions}          {contribution}
    +{.} { \PrintConference}             {conference}
    +{.} { \PrintBook}                   {book}
```

```
3262     +{.} { In }                          {booktitle}
3263     +{,} { pages~}                       {pages}
3264     +{.} { \PrintDateB}                  {date}
3265     +{.} { \PrintTranslation}            {translation}
3266     +{.} { Reprinted in \PrintReprint}   {reprint}
3267     +{.} { }                             {note}
3268     +{.} {}                              {transition}
3269 }
3270
3271 \BibSpec{conference}{%
3272     +{}  {}                          {title}
3273     +{}  {\PrintConferenceDetails} {transition}
3274 }
3275
3276 \BibSpec{innerbook}{%
3277     +{.} { \emph}                        {title}
3278     +{.} { }                             {part}
3279     +{:} { \emph}                        {subtitle}
3280     +{.} { }                             {series}
3281     +{,} { \voltext}                     {volume}
3282     +{.} { Edited by \PrintNameList}     {editor}
3283     +{.} { Translated by \PrintNameList}{translator}
3284     +{.} { \PrintContributions}          {contribution}
3285     +{.} { }                             {publisher}
3286     +{.} { }                             {organization}
3287     +{,} { }                             {address}
3288     +{,} { \PrintEdition}                {edition}
3289     +{,} { \PrintDateB}                  {date}
3290     +{.} { }                             {note}
3291     +{.} {}                              {transition}
3292 }
3293
3294 \BibSpec{report}{%
3295     +{}  {\PrintPrimary}                 {transition}
3296     +{.} { \emph}                        {title}
3297     +{.} { }                             {part}
3298     +{:} { \emph}                        {subtitle}
3299     +{.} { \PrintContributions}          {contribution}
3300     +{.} { Technical Report }            {number}
3301     +{,} { }                             {series}
3302     +{.} { }                             {organization}
3303     +{,} { }                             {address}
3304     +{,} { \PrintDateB}                  {date}
3305     +{.} { \PrintTranslation}            {translation}
3306     +{.} { Reprinted in \PrintReprint}   {reprint}
3307     +{.} { }                             {note}
3308     +{.} {}                              {transition}
3309 }
3310
3311 \BibSpec{thesis}{%
```

```
    +{}  {\PrintAuthors}                {author}
    +{,} { \emph}                       {title}
    +{:} { \emph}                       {subtitle}
    +{.} { \PrintThesisType}            {type}
    +{.} { }                            {organization}
    +{,} { }                            {address}
    +{,} { \PrintDateB}                 {date}
    +{.} { \PrintTranslation}           {translation}
    +{.} { Reprinted in \PrintReprint}  {reprint}
    +{.} { }                            {note}
    +{.} {}                             {transition}
}

\BibSpec{webpage}{%
    +{}  {\PrintAuthors}                {author}
    +{.} { }                            {title}
    +{:} { }                            {subtitle}
    +{.} { \PrintDateB}                 {date}
    +{.} { \url}                        {url}
    +{.} { Accessed \PrintDateField}    {accessdate}
    +{.} { }                            {note}
    +{.} {}                             {transition}
}
```

`\PrintEditorsA` When we consider editor names we have to think about some further complications. First, for the case of a book where editor names are listed in place of author names, just copy the same style with a bit of added text at the end.

```
\renewcommand{\PrintEditorsA}[1]{%
    \def\current@bibfield{\bib'editor}%
    \PrintNames{}{, editor\Plural{s}}{#1}%
    \erase@field\bib'editor
}
```

`\PrintTranslatorsA`

```
\renewcommand{\PrintTranslatorsA}[1]{%
    \def\current@bibfield{\bib'translator}%
    \PrintNames{}{, translator\Plural{s}}{#1}%
    \erase@field\bib'translator
}
```

⟨/bst⟩

The usual `\endinput` to ensure that random garbage at the end of the file doesn't get copied by `docstrip`.

```
\endinput
```

References

[1] David M. Jones, *User's Guide to the* `amsrefs` *Package*. distributed with the `amsrefs` code.

[2] Ellen Swanson, Arlene O'Sean, and Antoinette Schleyer, *Mathematics into Type*, updated, American Mathematical Society, 1999.

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

D

E

F

G

H

I

R

S

T

U

V

W

X

Y

Z