The achicago BibTeX style
A Chicago Manual BibTeX style

Matt Swift <swift@alum.mit.edu>

Version: 0.8a Date: 1998/08/14
Documentation revision: 1999/03/01

Abstract

The Chicago Manual of Style bibliography style BibTeX macros. The style is mostly style A, but borrows the author-date reference system from style B.

Contents

Part I
Discussion

Warning: *The dox are spotty for this bibstyle—sorry! This is a pretty stable style, however. I've included almost my entire test cases. If you're puzzled, you may compare* `frankenstein.bib` *with what you see in the References section at the end of this document. Or please write me.*

Reference list ordering: alphabetical by author or whatever passes for author in the absence of one.

1 User documentation

No bibstyle is going to elgantly handle everything while using them in a way that's at all compatible with the majority of existing bibstyles out there. And it's not clear that such a bibstyle would be worth the effort to write and learn to use properly. The time would be better spent, I think, writing some macros to make inserting hand-done exceptions seamlessly into the bibliography, so that the whole process could remain clean and automatic.

For the sceptical, and for your entertainment, an example. I wanted to a cite a certain paper by Freud in my dissertation. The cited paper has a title, date of first publication, and chapter number; I read this paper in a volume of Freud's papers with a title, volume editor, volume date, publisher, reprint date, and reprint publisher; this volume has a number in a series of publications (The International Psycho-Analytic Library) with a series editor; and also this volume has a volume number within a multivolume work with its own title (*Collected Papers*) and (supervising) translator. Readers honestly don't need all that information, but each one of those 15 pieces of information might be highly important for another citation, for which only two or three pieces of information were relevant or available. No one needs to write a bibstyle that handles it all, so at several places, the fields are assumed to be used for either one thing or another. For example, '' series *either* contains the title of the inclusive multivolume work *or* the series. When, as rarely happens, a volume is part of both at once, you have to give up one or the other.

Warning: *This is a complex bibstyle and the number of potential cases is very high. Please check your output, and write me with cases that are not set according to The Chicago Manual percepts. Please refer to the The Chicago Manual whenever possible.*

STAR the new fields.

entry type	description
address	Address of publisher of *cited* edition, whether a reprint or not.
annotation	A complete paragraph or more.
author	
bookauthor	Book's author whenever '' author is referring to the author of a part of the book. Usually an 'inbook' or "incollection" entry refers to a book with an editor but no author, but not with introductions and the like.

Table 1: Entry types.

entry type	description
booktitle	Book's title whenever '' title is referring to a part of the book.
chapter	The '' title names the '' chapterth part of the book. One expects to have nonempty '' booktitle. See also '' type.
edition	Edition information to be given after the book's title, which is '' booktitle if that is nonempty and '' title otherwise. Example: `2d~ed.,2~vols.in1~book`
editor	
	NOTE all f* and o* fields are only going to apply to 'book', 'incollection', 'inbook' entries. FIX: check – booklet? etc.
faddress	Address of publisher of first-published edition. Not used unless '' fyear is nonempty.
flanguage	Language of first-published edition. Not used unless '' translator or '' fyear is nonempty.
fpublisher	Publisher of first-published edition. Not used unless '' fyear is nonempty.
ftitle	Title of first-published edition (probably in '' flanguage). Not used unless '' fyear is nonempty.
fyear	Date of first-published edition.
howpublished	
illustrator	For 'book' entries only. When the author or editor illustrates, use a construction like `{theeditor}andothers`.
institution	For thesis entries only. FIX check
journal	For 'article' entries only.
key	often default with empty author, editor, organization
month	For 'journal' entries, also season, e.g., Winter
note	Begins its own block, so capitalize. Concluding puctuation is supplied automatically if not given in the field.
number	For 'journal' entry, number of the journal (if no "" volume) or number within the volume. For "book" entry, number of the book in the '' series. Makes sense only when '' series is a series rather than the title of a multivolume work.
oaddress	For a reprint, address of ORIGINAL publisher.
opublisher	For a reprint, publisher of ORIGINAL edition. A nonempty value implies the cited edition IS a reprint (and so must have '' oyear).
organization	
oyear	For a reprint, date of publication of ORIGINAL edition A nonempty value implies the cited edition IS a reprint (and so OUGHT TO have '' opublisher). The value will almost always differ from '' year.
pages	
publisher	Publisher of the cited edition.
school	
series	Series to which '' title belongs, or title of a multivolume work where the volumes have their own titles given in '' booktitle or '' title.

Table 1: Entry types.

entry type	description
seriesedition	Edition information to be given after ‘’ series. Makes sense only when ‘’ series is the title of a multivolume work with titled volumes. In the entry for the multivolume work, that is, when ‘’ series and ‘’ title are identical, use ‘’ seriesedition not ‘’ edition. See also ‘’ edition. Consequence of disobeying last is when references are less than min-crossrefs (i.e., no separate entry for the multivolume work) you get the edition information for the multivolume work appearing after the title of the component volume. Note: you can't really get away with putting a series editor here. I think you'd have to use the ‘’ note field for that; or we could make yet another field.
title	
translator	For ‘book’ entries only, the translator(s). See also “” flanguage. When the author or editor translates, use a construction like `{theeditor}andothers`.
type	For ‘inbook’ entries, the value overrides `Chapter`. When “” type is empty you get `Chapter~<chapter>of`. FIX: CHECK: For *-thesis entries, gives thesis-type word, e.g. `Master'sThesis`.
volume	For ‘journal’ entry, it's volume. For “book” entry, its number in the ‘’ series, or its volume number in a multivolume book.
year	Year of publication of cited edition, whether a reprint or not.
yearcomp	Year work composed. For ‘book’ entry only (so far).

Table 1: Entry types.

Part II
Implementation

2 Version control

Version information:

```
1 ^^A Keep the /^ %% \def\file.../ beginning exact!
2
3 %% A BibTeX bibliography style
4 %%    conforming to the Chicago Manual's A style
5 %%       but with B style author-date citations
6 %%          by Matt Swift <swift@alum.mit.edu>
7 %%
8 %% \def\fileinfo{A Chicago Manual BibTeX style}
9 %% \def\DoXPackageS {longtable}
10 %% \def\initelyHavECitationS {}
11 %% \def\fileversion{v0.8a}
12 %% \def\filedate{1998/08/14}
13 %% \def\docdate{1999/03/01}
14 %%
15
```

3 Implementation

3.1 Preliminary

These are global variables. 'output.state' will always have one of the four following scalar values, telling us where we are in producing the entry.

```
16 INTEGERS { output.state before.all mid.sentence after.sentence after.block }
```

These are used for various internal calculations explained in the functions that use them.

```
17 INTEGERS { j nameptr namesleft numnames len last.year.tag.num }
```

These are more scratch variables.

Warning: *Suspect a conflict among these scratch variables if a strange problem occurs! I think 'u' is the only one that will hold its value past an 'output'.*

```
18 STRINGS { s t u }
```

These are label variables used in the calculation of the final 'long.label', 'short.label', and `sort.key$`

```
19 STRINGS { last.long.label last.long.year.label following.year.tag.label}
```

These are the fields used. Not every entry type uses every field; the ones which are not ignored are listed with the definition of each entry type. Any fields in the `bib` database not on this list will be ignored by this bibliography style.

```
20 ENTRY
21   { address
22     annotation
23     author
24     bookauthor
```

```
    booktitle
    chapter
    edition
    editor
    faddress
    flanguage
    fpublisher
    ftitle
    fyear
    howpublished
    illustrator
    institution
    journal
    key
    month
    note
    number
    oaddress
    opublisher
    organization
    oyear
    pages
    publisher
    school
    series
    seriesedition
    title
    translator
    type
    volume
    year
    yearcomp
}
```

Integer entry variables. See the discussion of queue functions below and the string entry variable 'printed.name'.

```
{ printed.name.type
}
```

String entry variables.

'long.label' and 'short.label' hold the strings that will appear in the running text of a citation, preceding 'year.label' + 'year.tag.label'. 'long.label' is used by the regular citing commands; 'short.label' is used by the short citing commands (`\shortcite` etc.).

```
{ long.label short.label
```

'year.label' is a string representing the date (the actual *''* year field can be longer than 4 characters and have special characters in it, such as the `\noop` often given in certain examples as a tweak to sort order). 'year.tag.label' is either empty or a lower-case letter. The catenation of the two is used in the running text of the citation.

```
    year.label year.tag.label
```

'long.label' and 'short.label' contain a formatted version of one of the entry's fields (such as *''* author, *''* editor, etc.), and is used in the running text at the location

of the user's \cite command. 'printed.name' contains the string that appears at the front of the bibliography entry. Usually it is going to be the same thing as 'long.label', but sometimes we will be labelling it as a duplicate entry.

'printed.name' is going to be set in the sorting passes done before the entry-type functions are called, so, e.g., you might find some "make" functions branching on what's in 'printed.name'.

```
    printed.name
}
```

stop with the \end{document}.

This bibstyle does not distinguish between sentences and blocks. There are no sentences here, only blocks. In the *achicago* package, \newblock adds a very small and rubber amount of horizontal space, relaxing somewhat the \frenchspacing in effect. I have not used 'new.sentence in this bibstyle, but I have not removed the capability of telling the distinction between sentences and blocks. Someone would need to merely put 'new.sentence somewhere in the bibstyle here, probably replacing a 'new.block, and then define \newblock to do something more interesting than it presently does. Someone wishing to take advantage of the distinction ought to review all relevant functions carefully.

init.state.constants

This should be called before any processing begins.

```
FUNCTION {init.state.consts}
{ #0 'before.all :=
  #1 'mid.sentence :=
  #2 'after.sentence :=
  #3 'after.block :=
}
```

3.2 Macros

These are required macro abbreviations. The user can redefine them. The billions of macros for obscure journals in all the standard styles just waste memory. They can be added by the user as needed.

```
MACRO {jan} {"January"}
MACRO {feb} {"February"}
MACRO {mar} {"March"}
MACRO {apr} {"April"}
MACRO {may} {"May"}
MACRO {jun} {"June"}
MACRO {jul} {"July"}
MACRO {aug} {"August"}
MACRO {sep} {"September"}
MACRO {oct} {"October"}
MACRO {nov} {"November"}
MACRO {dec} {"December"}
```

3.3 Basic logic

not
and
or

```
FUNCTION {not}
{   { #0
    }
```

```
    { #1
    }
  if$
}

FUNCTION {and}
{   'skip$
    { pop$
      #0
    }
  if$
}

FUNCTION {or}
{   { pop$
      #1
    }
    'skip$                      %$
  if$
}
```

3.4 Punctuation and abbreviations

comma
space
tie
colon
etal
unidentified

These definitions are made so that later functions are easier to read. The reason these are functions not macros is so the user cannot change them. They need to come before the output functions.

```
FUNCTION {comma}
{ ","
}

FUNCTION {colon}
{ ":"
}

FUNCTION {space}
{ " "
}

FUNCTION {tie}
{ "~"
}

FUNCTION {etal}
{ " et~al."
}
```

Below, we need to be able to set a string to a value which would never be generated from any string in a bibliography database. It's safe to assume that this string is such a one.

```
FUNCTION {unidentified}
{ "HiGHlY*Peco@lIEr"
}
```

3.5 Output functions

The 'output' function automatically handles interunit punctuation.

A call to 'output' should be made with a string and a punctuation mark on the stack. The punctuation mark is what should *precede* the string in the case the string begins in the middle of a sentence. Strings which begin sentences are preceded by periods. Strings which begin a new block are preceded by periods and the `\newblock` command. (It makes no sense for a string to begin at the end of a sentence.)

Calls to 'output.internal' leave the string on the stack until the next time 'output.internal' is called, when it will get printed out. The function 'output.begin' puts an empty string on the stack, and should be called before beginning any series of calls to 'output'. 'output.end' cleans up and should be called at the conclusion of a series of calls to 'output'.

`output.begin` Prepare for a series of calls to 'output'. Put an empty string on the stack and set 'output.state to 'before.all'.

```
FUNCTION{output.begin}
{ ""
  before.all 'output.state :=
}
```

`output.end` Conclude a series of calls to 'output'. Simply calls `write$`.

```
FUNCTION {output.end}
{ write$
}
```

`new.block` Begin a new block.

```
FUNCTION {new.block}
{ output.state before.all =
    'skip$
    { after.block 'output.state :=
    }
  if$
}
```

`new.sentence` Begin a new sentence. FIX: hmm, wouldn't we want to put a TeX command here for intersentence space now? Why do we mess with the usual spacefactor for periods anyway? When in bibliogrphies are the usual TeX rules not going to be what we want, and give too much space after a period? answer: when a lowercase letter followed by period, in e.g. sec. 2 and so on. often bibstyle will put tie there, but sometimes not I think, so need to have that be not an intersentence space. well, if i start using sentences instead of blocks, I think perhaps doing the same hskip after a sentence with a new command `\newsentence` perhaps as in `\newblock` would be the right thing to do.

```
FUNCTION {new.sentence}
{ output.state before.all = output.state after.block = or
    'skip$
    { after.sentence 'output.state :=
    }
  if$
}
```

output.internal \stackpunct string/entry(nonnull)null

```
FUNCTION {output.internal}
{ 's :=     % s := string/entry(nonnull)
  space * 't :=     % t := punctuation + space
  output.state mid.sentence =
    { t * write$
    }
    { output.state after.block =
        { add.period$ write$
          newline$
          "\newblock " write$
        }
        { output.state before.all =
            'write$
```

Final case is 'output.state' = 'after.sentence', which means we just add a period, but no newline or \newblock command.

```
            { add.period$ space * write$
            }
          if$
        }
      if$
```

We are now mid-sentence, whether we were before or not.

```
      mid.sentence 'output.state :=
    }
  if$
```

The actual string given to 'output.internal' is left on the stack here for next time we call 'output.internal' or 'output.end', whichever comes next.

```
  s
}
```

output \stackstring/entry puncteither null OR punct string/entry(nonnull)

Check if string/entry is null. If it's null do nothing. If it's nonempty call 'output.internal' on it with the given punctuation.

```
FUNCTION {output}
{ swap$ duplicate$ empty$
    { pop$
      pop$
    }
    'output.internal
  if$
}
```

output.bibitem The optional argument to \bibitem will be made the definition of \b @foo where foo is the cite$ key. \b @foo will be executed by all the various citing commands with a definition of \SCcite that is appropriate to each citing command.

The thebibliography environment actually ignores the optional argument to \bibitem. The environment sets the formatting so that the first part of the main entry hangs out to the left and looks like a label.

The unexpandable protection is necessary, instead of a simple protection, in order to work with *newclude*'s tag option.

I guess we use `write$` for each part instead of catenating them and using a single `write$` in case the catenation gets longer than BibTeX can handle. I don't know, this is how I found it. It seems like catenation and a single call to `write$` would be faster.

```
FUNCTION {output.bibitem}
{ newline$
  "\bibitem[\UnexpandableProtect\SCcite{" write$
  long.label write$
  "}{" write$
  short.label write$
  "}{" write$
  year.label year.tag.label * write$
  "}]{" write$
  cite$ write$
  "}" write$
  newline$            %$
}
```

3.6 Simple auxiliary functions

`begin.entry` Begin an entry.

```
FUNCTION {begin.entry}
{ output.begin
  output.bibitem
}
```

This "make" function needs to precede 'finish.entry'; see later for a general discussion of "make" functions.

`make.annotation` Annotations can be omitted simply by nullifying the `SCannotation` environment. The package or class file should take care of this.

```
FUNCTION {make.annotation}
{ annotation empty$
    { ""
    }
    { "\begin{SCannotation}"
      annotation *
      "\end{SCannotation}" *
    }
  if$
}
```

`finish.entry` Conclude an entry. `\stack`stringnull

```
FUNCTION {finish.entry}
{ add.period$
  output.end
  newline$
  make.annotation write$            %$

}
```

`field.or.null` `\stack`field literalfield contents or null string if the field was missing

```
FUNCTION {field.or.null}
```

```
{ duplicate$ empty$
    { pop$
      ""
    }
    'skip$
  if$                          %$
}
```

italicize \stackstring or entrystring

Italicize the top string on the stack and provide italic corrections. Null goes to null. We want to explicitly italicize rather than use `\emph`, since semantic emphasis might be indicated in some other way in the document, e.g., slanted type.

```
FUNCTION {italicize}
{ duplicate$ empty$
    { pop$
      ""
    }
    { "\textitswitch{" swap$ * "}" *
    }
  if$                            %$
}
```

parenthesize \stackstring-or-entry:S"(" + S + ")"

```
FUnCTION {parenthesize}
{ duplicate$ empty$
    { pop$
      ""
    }
    { "(" swap$ * ")" *
    }
  if$                            %$
}
```

entry.clip \stackstringstring FIX: `btxhak.dvi` does not tell us the difference between `entry.max$` and `global.max$`. Guess: `global.max$` is the biggest thing BibTeX can deal with internally; `entry.max$` is the biggest thing that can be sent to a `write$`.

entry.clip \stackstring-Sstring-S-possibly-shortened

Clips a string down to `entry.max$`, if it's longer.

```
FUNCTION {entry.clip}
{ #1 entry.max$ substring$
}
```

check \stackS warningS Issues warning if the string is empty.

```
FUNCTION {check}
{ 't :=
  duplicate$ empty$
    { "empty " t * " in " * cite$ * warning$
    }
    'skip$
  if$
}
```

multi.page.p \stacknothing#0 or #1

Set 't to ‘’ pages. Look at 't character by character for a hypen, comma, or plus. If we find one, return true immediately; else return false.

```
FUNCTION {multi.page.p}
{ pages 't :=
  #0 'j :=
    { j not
      t empty$ not
      and
    }
    { t #1 #1 substring$
      duplicate$ "-" =
      swap$ duplicate$ "," =
      swap$ "+" =
      or or
        { #1 'j :=
        }
        { t #2 global.max$ substring$ 't :=
        }
      if$
    }
  while$
  j
}
```

n.dashify \stackstringstring

Make single-dashes into double-dashes.

```
FUNCTION {n.dashify}
{ 't :=
  ""
    { t empty$ not
    }
    { t #1 #1 substring$ "-" =
        { t #1 #2 substring$ "--" = not
            { "--" *
      t #2 global.max$ substring$ 't :=
    }
    {   { t #1 #1 substring$ "-" =
                }
{ "-" *
  t #2 global.max$ substring$ 't :=
}
      while$
    }
  if$
        }
        { t #1 #1 substring$ *
          t #2 global.max$ substring$ 't :=
        }
      if$
    }
  while$                  %$
}
```

tie.or.space.connect \stackstring stringstring

If the second string has less than 5 characters, the two are tied together, otherwise they are catenated with a space between them.

```
FUNCTION {tie.or.space.connect}
{ duplicate$ text.length$ #5 <
    { tie
    }
    { space
    }
  if$
  swap$ * *
}
```

either.or.check \stackmsg fieldnull If field is not empty, give a warning appropriate for saying this field and an already-existing field can't be both used in the same entry.

```
FUNCTION {either.or.check}
{ empty$
    'pop$
    { "can't use both " swap$ * " fields in " * cite$ * warning$
    }
  if$
}
```

3.7 Queue functions

Queue functions expect a string on the stack. If it is empty, and the field for which the function is named is noempty, the string will be replaced with that field. (e.g., 'author.q' will put `` author on the stack if `` author is nonempty and the 'author.q' was passed an empty string). A chain of queue functions must begin with something on the stack, adn will leave the first nonempty field named in the chain on the stack.

Queue functions also increment an integer variable, 'j, whenever they are passed an empty string. At the end of the queue, 'j tells us which function in the queue contributed the string that results at the end of the queue, starting with zero for the first queue function after 'start.a.q'. This value will be stored in the integer entry variable 'printed.name.type'. In order for this to work, queue functions must always be called in the same order: `` author ('#0) `` editor ('#1'), `` organization ('#2'), and `` key ('#3'). For entry types where a certain field should not appear in the queue, use 'blank.q' in its place in the order.

Queues are used the above way to generate the string entry variables 'long.label', 'short.label', and 'printed.name', but queue functions are sometimes used individually in an entry-type function, without interest in the value of 'j.

'start.a.q' must begin a queue. It initializes 'j and passes an empty string to the first queue function.

start.a.q

```
FUNCTION {start.a.q}
{ #-1 'j :=
  ""
}
```

```
         inc.j
         key.q 317 FUNCTION {inc.j}
organization.q 318 { j #1 + 'j :=
      editor.q 319 }
      author.q 320
       blank.q 321 FUNCTION {key.q}
               322 { duplicate$ empty$
               323     { pop$
               324       key field.or.null
               325       inc.j
               326     }
               327     'skip$
               328   if$
               329 }
               330
               331 FUNCTION {organization.q}
               332 { duplicate$ empty$
               333     { pop$
               334       organization field.or.null
               335       inc.j
               336     }
               337     'skip$
               338   if$
               339 }
               340
               341 FUNCTION {editor.q}
               342 { duplicate$ empty$
               343     { pop$
               344       editor field.or.null
               345       inc.j
               346     }
               347     'skip$
               348   if$
               349 }
               350
               351 FUNCTION {author.q}
               352 { duplicate$ empty$
               353     { pop$
               354       author field.or.null
               355       inc.j
               356     }
               357     'skip$
               358   if$
               359 }
               360
               361 FUNCTION {blank.q}
               362 { duplicate$ empty$
               363     'inc.j
               364     'skip$
               365   if$
               366 }
```

`check.q` \stackstring-S warning-stringstring-S

This should be the final queue command, though it is not a true queue com-

mand, since an extra warning string should be on the top of the stack above the string that has passed along the queue. The warning string should be an English disjunctive list of the queue commands that have come before it, so the user can tell which fields are missing. For example "author, organization, or key" if the queue has been 'author.q' 'blank.q' 'organization.q' 'key.q'.

If the queue string is empty, 'check.q' issues the warning and puts on the stack an emergency string of the first three letters of the LaTeX key (i.e., `cite$`), which should allow BibTeX to proceed.

```
FUNCTION {check.q}
{ 't :=
  duplicate$ empty$
    { pop$
      cite$ #1 #3 substring$ 'u :=
      "Need " t * " in " * cite$ * "; using:  " * u * warning$
      u
    }
    'skip$
  if$
}
```

3.8 Format queue functions

Format queue functions may be inserted into a queue with queue functions. If they find an empty string on the stack, they do nothing. If they find a nonempty string, they modify it.

There are 4 cases for the labels, where n is some constant:

one author	Foo
one to n	Foo, Bar, and Baz
use of "and others"	Foo, Bar et al.
more than n	Foo et al.

`format.short.label.names.q` If the string on the stack is empty, do nothing, leaving the empty string there. If passed a nonempty string, format it to produce something appropriate for 'short.label'.

```
FUNCTION {format.short.label.names.q}
{ duplicate$ empty$
    'skip$                          %$ for emacs
    { 's :=
      s num.names$ 'numnames :=     %$ for emacs
      s #1 "{vv~}{ll}" format.name$ %$ for emacs
```

This constant is the number of authors over which we use "et al.". If you change this constant, change some logic below.

```
      numnames #2 >
{ etal *
}
{ numnames #1 =
    'skip$
    { s #2 "{ff}{vv}{ll}{ jj}" format.name$ "others" =
```

Above, I'm trying a simplification that I think should work. OLD: `snameptr"{ff}{vv}{ll}{jj}"format.name$"others"=`

```
390       { etal *
391       }
392       { " and " * s #2 "{vv~}{ll}" format.name$ *
393       }
394     if$
395           }
396   if$
397 }
398     if$
399    }
400   if$                       %$
401 }
```

If you want the short label to have 3 (or more) names in it, use this function. The constant is 3 here.

```
FUNCTION {format.short.label.names.q}
{ duplicate$ empty$                     %$
    'skip$                              %$
    { 's :=
      s num.names$ 'numnames :=         %$
      s #1 "{vv~}{ll}" format.name$     %$
%    \end{macrocode}
%  This constant is the number of authors over which we use "et~al.".
%    \begin{macrocode}
      numnames #3 >
        { etal *
        }
        { numnames #1 - 'namesleft :=
          #2 'nameptr :=
            { namesleft #0 >
            }
            { nameptr numnames =      % i.e, nameptr points to the last name
                { s nameptr "{ff}{vv}{ll}{ jj}" format.name$ "others" =
                    { etal *
                    }
                    { numnames #2 >
                        { comma *
                        }
                        'skip
                      if$
                      " and " * s nameptr "{vv~}{ll}" format.name$ *
                    }
                  if$
                }
                { ", " * s nameptr "{vv~}{ll}" format.name$ *
                }
              if$
              nameptr #1 + 'nameptr :=
              namesleft #1 - 'namesleft :=
            }
          while$             %$
        }
      if$
    }
```

```
  if$                           %$
}
```

format.long.label.names.q If the string on the stack is empty, do nothing, leaving the empty string there. If it is nonempty, format it to produce something appropriate for ‘long.label’.

```
FUNCTION {format.long.label.names.q}
{ duplicate$ empty$
    'skip$
    { 's :=
      #1 'nameptr :=                % nameptr = 1;
      s num.names$ 'numnames :=     % numnames = num.names$(s);
      numnames 'namesleft :=
      % begin a WHILE loop
        { namesleft #0 >
        }
        { s nameptr "{vv~}{ll}" format.name$ 't :=  % get the next name
          nameptr #1 >
            { namesleft #1 >
                  { comma * space * t *
                  }
                  { numnames #2 >
                      { comma *
                      }
                      'skip$
                    if$
                    t "others" =
                        { etal *
                        }
                        { " and " * t *
                        }
                      if$
                   }
                 if$
            }
            't
          if$
          nameptr #1 + 'nameptr :=          % nameptr += 1;
          namesleft #1 - 'namesleft :=      % namesleft -= 1;
        }
      while$
    }
  if$                   %$
}
```

3.9 Format names functions

The format functions modify a string on the stack. They make no warnings, produce no output, and place no spaces or punctuation before or after their string, just modify it somehow.

format.names.lastfirst Format a name list. Output first author last name first, and subsequent authors with first name first.

```
FUNCTION {format.names.lastfirst}
{ 's :=
  #1 'nameptr :=               % nameptr = 1;
  s num.names$ 'numnames :=    % numnames = num.name$(s);
  numnames 'namesleft :=
  % begin WHILE loop
    { namesleft #0 >
    }
    { nameptr #1 =
        { s nameptr "{vv~}{ll}{, jj}{, ff}" format.name$ 't :=
        }
        { s nameptr "{ff~}{vv~}{ll}{, jj}" format.name$ 't :=
        }
      if$
      nameptr #1 >
        { namesleft #1 >
              { comma * space * t *
              }
              { numnames #2 >
                  { comma *
                  }
                  'skip$
                if$
                t "others" =
                    { etal *
                    }
                    { " and " * t *
                    }
                  if$
               }
             if$
        }
        't
      if$
      nameptr #1 + 'nameptr :=          % nameptr += 1;
      namesleft #1 - 'namesleft :=      % namesleft =- 1;
    }
  while$                      %$
}
```

`format.names.firstfirst` Same as 'format.names.lastfirst' but all names are first name first.

```
FUNCTION {format.names.firstfirst}
{ 's :=
  #1 'nameptr :=               % nameptr = 1;
  s num.names$ 'numnames :=    % numnames = num.name$(s);
  numnames 'namesleft :=
    { namesleft #0 >
    }
    { s nameptr "{ff~}{vv~}{ll}{, jj}" format.name$ 't :=
      nameptr #1 >
        { namesleft #1 >
            { comma * space * t *
            }
            { numnames #2 >
```

```
                { comma *
                }
                'skip$
              if$
              t "others" =
                  { etal *
                  }
                  { " and " * t *
                  }
                if$
              }
            if$
          }
          't
        if$
        nameptr #1 + 'nameptr :=          % nameptr += 1;
        namesleft #1 - 'namesleft :=      % namesleft =- 1;
      }
    while$                      %$
}
```

3.10 Make functions

Make functions expect nothing on the stack, and leave one string there. Any side effects of setting global variables are noted in the comments. If a nonempty string cannot be made due to lack of the right fields, we simply leave an empty on on the stack.

Notice that 'make.annotation' has to be above, before 'finish.entry'.

`make.edition` We use some logic here because we want to use the `''` seriesedition for the `''` edition in certain cases.

```
FUNCTION {make.edition}
{ edition empty$
    { seriesedition empty$ not series field.or.null title field.or.null = and
        { seriesedition
        }
        { ""
        }
      if$                   %$
    }
    { edition
    }
  if$                       %$
}
```

`make.year.month` `make.year.nomonth` Make a year string. We don't we use 'year.label' because that string is sterilized to make it appropriate for use in running text in citations (and for sorting I think); in the bibliography entry, we can be a little more lax, accepting special characters, and text that is not simply four numerals.

```
FUNCTION {make.year.month}
  { year empty$
      { ""
      }
```

```
      { year year.tag.label *
month empty$
  'skip$
  { comma * space * month * }
if$
      }
    if$                    %$
}
FUNCTION {make.year.nomonth}
{ year empty$              %$
    { ""
    }
    { year year.tag.label *
    }
  if$                      %$
}
```

`make.year.or.oyear.month` Gives a year string. If ‘’ oyear is empty, calls ‘make.year.month’. Warns in almost-certainly-erroneous case of an ‘’ oyear without a ‘’ year.

```
FUNCTION {make.year.or.oyear.month}
{ oyear empty$
    { make.year.month
    }
    { year empty$
        { "oyear without year in " cite$ * warning$
        }
        'skip$
      if$
      oyear year.tag.label *
      month empty$
        'skip$
{ comma * space * month *
        }
      if$
    }
  if$
}
```

`make.authors` This is for clarity below. Really we could say ‘printed.name’.

```
FUNCTION {make.authors}
{ printed.name empty$
    { "empty ‘printed.name’ in ‘make.authors in " cite$ * warning$
      ""
    }
    { printed.name
    }
  if$
}
```

`make.editors.primary` This is for when the editor is the leading thing in the bibentry. If ‘printed.name’ is empty, we put an empty string on the stack. If ‘printed.name’ is nonempty and differs from ‘’ editor then ‘printed.name’ is a duplicate editor. We *do* want to follow this by a suffix. If they’re the same, then we format the ‘’ editor field, add a suffix, and leave it on the stack.

FIX: I don't think 'printed.name' should ever be empty.

```
FUNCTION {make.editors.primary}
{ printed.name empty$
    { "empty `printed.name' in `make.editors.primary' in " cite$ * warning$
      ""
    }
    { printed.name
      editor num.names$ #1 >
        { ",~eds." *
        }
        { ",~ed." *
        }
      if$
    }
  if$
}
```

`make.editors.secondary` This is for making editors' names that follow a title.

```
FUNCTION {make.editors.secondary}
{ editor empty$
    { ""
    }
    { output.state after.sentence =
      output.state after.block = or
        { "E" }
        { "e" }
      if$
      "dited by " * editor format.names.firstfirst *
    }
  if$                       %$
}
```

`make.bookauthors.secondary` This is for making authors' names that follow a title. When we use this, we are only going to be interested in the `'' bookauthor.

```
FUNCTION {make.bookauthors.secondary}
{ bookauthor empty$
    { ""
    }
    { output.state after.sentence =
      output.state after.block = or
        { "B" }
        { "b" }
      if$
      "y " * bookauthor format.names.firstfirst *
    }
  if$
}
```

`make.translators` `make.illustrators` Translators and illustrators always follow the title.

```
FUNCTION {make.translators}
{ translator empty$
    { ""
    }
```

```
    { output.state after.sentence =
      output.state after.block = or
        { "T" }
        { "t" }
      if$
      "ranslated " *
      flanguage empty$
        'skip$
        { "from the " * flanguage * space *
        }
      if$
      "by " * translator format.names.firstfirst *
    }
  if$
}

FUNCTION {make.illustrators}
{ illustrator empty$
    { ""
    }
    { output.state after.sentence =
      output.state after.block = or
        { "I" }
        { "i" }
      if$
      "llustrated by " * illustrator format.names.firstfirst *
    }
  if$
}
```

make.booktitle.ital
make.title.ital
make.as.ftitle.ital

```
FUNCTION {make.title.ital}
{ title field.or.null italicize
}

FUNCTION {make.booktitle.ital}
{ booktitle field.or.null italicize
}

FUNCTION {make.as.ftitle.ital}
{ ftitle empty$
    { ""
    }
    { "as " ftitle italicize *
    }
  if$
}
```

make.title.doublequote

Put the title in quotes on the stack. `\Wrapquotes` is going to properly handle punctuation that follows the closing quotation marks by sucking it inside the quotes.

```
FUNCTION {make.title.doublequote}
{ title empty$
    { ""
```

```
    }
    {
      "\Wrapquotes{" title * "}" *
    }
  if$
}

FUNCTION {make.pages}
{ pages empty$
    { ""
    }
    { multi.page.p
        { pages n.dashify
        }
        { pages
        }
      if$
    }
  if$                       %$
}
```

ke.vol.series.num.month.pages For articles, the number of cases to handle these fields is very large: 32 to do it properly! These cases could perhaps be collapsed, but it is going to be much easier to read here and to maintain if we do it brutishly by elaborating each case. The only exception I've made is when two adjacent leaves are very similar. The way I've laid out this funciton here, adjacent leaves are going to be the pairs of cases with and without a nonempty `'' pages.

There are 5 relevant fields, each of which can be empty or not ($2^5 = 32$). In the comments below, I mark each case with a series of 1 to 5 letters represent the cases where that field is nonempty: `V` for `'' volume, `S` for `'' series, `N` for `'' number, `M` for `'' month, and `P` for `'' pages. Thus, the case marked with the comment `SMP` is the one where the `'' series, `'' month, and `'' pages fields are nonempty, and `'' volume and `'' number are empty.

FIX: when I use 'make.pages' in this function, it's slightly inefficient, since 'make.pages' does a check for empty pages which is unnecessary here.

FIX: this does not yet handle entries with nonempty `'' series.

FIX: go through and combine adjacent leaves.

We could really put the punctuation that should precede this unit right at the beginning of the string we're making here, but let's conform better to the conventions and put this in the scratch variable 'u instead. I don't think this function will ever get called at any time other than mid-sentence, after the journal title, but who knows. So a call to this function should look like:

```
make.vol.series.num.month.pages u output
```

Remember that when 'output.internal' writes the punctuation, it adds a space following, so it is unnecessary to begin the string this function returns with a space. (Occasionally below, we want (and produce) a colon followed by no space; we can do this because we are directly creating this string, not using the 'output' routine.)

```
FUNCTION {make.vol.series.num.month.pages}
{ volume empty$
    { series empty$
```

```
        { number empty$
    { month empty$
{ pages empty$
                    % none of VSNMP
    { space 'u :=
                      ""
    }
                    % P
    { comma 'u :=
                      multi.page.p
                        { "pp." pages n.dashify
}
                        { "p." pages
}
                      if$
                      tie.or.space.connect
    }
  if$
}
{ pages empty$
                    % M
    { comma 'u :=
                      month
    }
                    % MP
    { comma 'u :=
                      month comma * space * make.pages *
    }
  if$
}
      if$
    }
    { month empty$
{ pages empty$
                    % N
    { comma 'u :=
                      "no." number tie.or.space.connect
    }
    % NP
    { comma 'u :=
                      "no." number colon * make.pages * tie.or.space.connect
    }
  if$
}
{ pages empty$
    % NM
    { comma 'u :=
                      "no." number tie.or.space.connect
                      space * month parenthesize *
    }
    % NMP
    { comma 'u :=
                      "no." number tie.or.space.connect
                      space * month parenthesize *
```

```
                      colon * space * make.pages *
    }
  if$
}
      if$
    }
  if$
        }
        { number empty$
    { month empty$
{ pages empty$
    % S
    {
    }
    % SP
    {
    }
  if$
}
{ pages empty$
    % SM
    {
    }
    % SMP
    {
    }
  if$
}
      if$
    }
    { month empty$
{ pages empty$
    % SN
    {
    }
    % SNP
    {
    }
  if$
}
{ pages empty$
    % SNP
    {
    }
    % SNMP
    {
    }
  if$
}
      if$
    }
  if$
        }
      if$
```

```
    }
    { series empty$
        { number empty$
    { month empty$
{ pages empty$
    % V
    { comma 'u :=
                        "vol." volume tie.or.space.connect
    }
    % VP
    { space 'u :=
                        volume colon * make.pages *
    }
  if$
}
{ pages empty$
    % VM
    { comma 'u :=
                        "vol." volume tie.or.space.connect
                        space * month parenthesize *
    }
    % VMP
    { space 'u :=
                        volume space *
                        month parenthesize * colon * space * make.pages *
    }
  if$
}
      if$
    }
    { month empty$
{ pages empty$
    % VN
    { space 'u :=
                       volume ", no." number tie.or.space.connect *
    }
    % VNP
    { space 'u :=
                        volume space *
                        number parenthesize * colon * space * make.pages *
    }
  if$
}
{ pages empty$
    % VNM
    { space 'u :=
                        volume ", no." number tie.or.space.connect *
                        space * month parenthesize *
    }
    % VNMP
    { space 'u :=
                        volume space * number parenthesize * colon * space *
                        make.pages * space * month parenthesize *
    }
```

```
  if$
}
      if$
    }
  if$
        }
        { number empty$
    { month empty$
{ pages empty$
    % VS
    {
    }
    % VSP
    {
    }
  if$
}
{ pages empty$
    % VSM
    {
    }
    % VSMP
    {
    }
  if$
}
      if$
    }
    { month empty$
{ pages empty$
    % VSN
    {
    }
    % VSNP
    {
    }
  if$
}
{ pages empty$
    % VSNM
    {
    }
    % VSNMP
    {
    }
  if$
}
      if$
    }
  if$
        }
      if$
    }
  if$                          %$
```

```
}
```

make.crossref

```
FUNCTION {make.crossref}
{ "\citeNP{" crossref * "}" *
}
```

output.series.number Outputs the number with appropriate capitalization. Warns if we also have nonempty ‘’ seriesedition.

```
FUNCTION {output.series.number}
{ series empty$
    { number empty$
        { ""
        }
        { "Can't use `number' without `series' in " cite$ * warning$
          ""
        }
      if$
    }
    { number empty$
{ series
          seriesedition empty$
            'skip$
            { comma output
              new.block
              seriesedition
            }
          if$
}
{ series space *
          "no." *
  number tie.or.space.connect
 "number and seriesedition" seriesedition either.or.check
}
      if$
    }
  if$                         %$
  comma output
}
```

output.volume.series The ‘’ series is not required.

```
FUNCTION {output.volume.series}
{ volume empty$
    { ""
    }
    { "Volume" volume tie.or.space.connect
      series empty$
        'skip$
        { " of " * series italicize *
          seriesedition empty$
            'skip$
            { comma output
              new.block
              seriesedition
```

```
            }
          if$
        }
      if$
      "volume and number" number either.or.check
    }
  if$
  comma output
}
```

make.chapter.of To say for example "Part 3" instead of "Chapter 3", put "Part" into the '' type field, and the number and the preposition into the '' chapter field, e.g., "3 of". A tie is automatically put in between the '' type field and the '' chapter field in this case. We can't make a decision based on the length of the '' chapter field when it's going to contain the preposition.

OK, putting the preposition in the chapter field doesn't work out when you do crossreferences. So let's try "of" as a general preposition.

```
FUNCTION {make.chapter.of}
{ chapter empty$
    { ""
    }
    { type empty$
        { "Chapter"
        }
        { type "t" change.case$
        }
      if$
      chapter tie.or.space.connect
      " of " *
    }
  if$
}
```

make.chapter

```
FUNCTION {make.chapter}
{ chapter empty$
    { ""
    }
    { type empty$
        { "Chapter"
        }
        { type "t" change.case$
        }
      if$
      chapter tie.or.space.connect
    }
  if$
}
```

make.number.tr

```
FUNCTION {make.number.tr}
{ type empty$
    { "Technical Report"
```

```
    }
    { type "t" change.case$
    }
  if$
  number empty$
    'skip$
    { number tie.or.space.connect
    }
  if$
}
```

3.11 Label functions

calc.labels
set.printed.name.type

Sets 'short.label', 'long.label' and 'year.label' from the original database. Should only need to be called once. We also set 'printed.name.type' to an appropriate value, but we do not set 'printed.name', that's done later.

Warning: *If this logic with regard to what is used as the printed name does not correspond to what's actually implemented in each entry-type function, I think you are going to get some weird errors, like 'make.author' and 'make.editors' are not going to give you what you expect.*

```
FUNCTION {set.printed.name.type}
{ j 'printed.name.type :=
}

FUNCTION {calc.labels}
{ type$ "book" =
  type$ "inbook" =
  or
    { start.a.q
      author.q
      editor.q
      format.short.label.names.q
      blank.q
      key.q
      set.printed.name.type
      "author, editor, or key to make 'short.label'" check.q
    }
    { type$ "proceedings" =
        { start.a.q
          blank.q
          editor.q
          format.short.label.names.q
          organization.q
          key.q
          set.printed.name.type
          "editor, organization, or key to make 'short.label'" check.q
        }
        { type$ "manual" =
            { start.a.q
              author.q
              format.short.label.names.q
              blank.q
              organization.q
```

```
              key.q
              set.printed.name.type
              "author, organization, or key to make `short.label'" check.q
            }
```

Default case. FIX: have I got all the right cases covered?

```
            { start.a.q
              author.q
              format.short.label.names.q
              blank.q
              blank.q
              key.q
              set.printed.name.type
              "author or key to make `short.label'" check.q
            }
          if$
        }
      if$
    }
  if$
  'short.label :=

  start.a.q
  author.q
  editor.q
  format.long.label.names.q
  organization.q
  key.q
  "author, editor, organiztion, or key to make `long.label'" check.q
  'long.label :=

  oyear empty$
    { year
    }
    { oyear
    }
  if$
  field.or.null purify$ #-1 #4 substring$             %$
  'year.label :=
}
```

3.12 Entry types

3.12.1 Article

`article` An article from a journal or magazine. Required: ‘’ author, ‘’ title, ‘’ journal, ‘’ year. Optional: ‘’ volume, ‘’ number, ‘’ pages, ‘’ month, ‘’ note. Title quoted.

FIX: this is not true any more. hmm, which warnings to give? Warnings given for no ‘’ pages; no ‘’ volume or ‘’ number; and ‘’ number but no ‘’ volume. It's a drag, but the latter warning should remain there. The reason is that this bibstyle will produce a very misleading entry if you mistakenly omit the volume and give a number for a journal.

FIX: this is the old def of article, following output of title through end of

crossref conditional.

```
      volume field.or.null
%    \end{macrocode}
%  One item (the volume) is on the stack now. Each of the four cases in the
%  following conditionals should leave a second item on the stack.
%    \begin{macrocode}
      number empty$
        { volume empty$
            { "empty volume and number in " cite$ * warning$
              ""
            }
            { colon
            }
          if$
        }
        { volume empty$
            { "number but no volume in " cite$ * warning$
%    \end{macrocode}
% FIX when a journal has no volume but a number, there should be a comma after
% the journal, a space and then ``no. n:ppp''.  But we have already said
% ``space output'' after the journal name, no doubt.  So we have to make this
% decision earlier.
%    \begin{macrocode}
              "\unskip, no." tie * number * colon *
            }
            { "(" number * ")" * tie.or.space.connect
%    \end{macrocode}
% FIX: I don't think this is handling the case without pages right -- there
% will be a colon with nothing following.  I could save either colon or
% colon+space in a variable, and slap it on before the pages if there are pages
% and leave it off if there are no pages.  Hmm, no: see chicago:14, item 15.215
% for how to handle missing pages.  Not easy.
%
% Now there's just one item (the volume) on the stack, so leave the following
% as the second item.
%    \begin{macrocode}
              colon space *
            }
          if$
        }
      if$ *
      make.pages "pages" check *
      space output
      month parenthesize space output
```

A crossref from an article says "See this, pages xxx." See the *Chicago Manual* 16:107 for this peculiar bit about the spacing after the colon.

```
FUNCTION {article}
{ begin.entry
  make.authors "author" check key.q space output
  new.block
  make.year.nomonth "year" check space output
  new.block
```

```
  make.title.doublequote "title" check space output
  new.block
  crossref empty$
    { journal italicize "journal" check space output
      series empty$
        'skip$
```

FIX: take this out when we finish 'make.vol.series...'

```
        { "ARTICLE NOT SET UP FOR NONEMPTY SERIES! in " cite$ * warning$
        }
      if$
      volume empty$ number empty$ month empty$ pages empty$ and and and
        { "empty volume, number, month, and pages in " cite$ * warning$
        }
        'skip$
      if$
```

See definition of 'make.vol.series.num.month.pages' for why we set 'u in that function.

```
      make.vol.series.num.month.pages u output
    }
```

Unusual case where you are probably citing one journal issue and numerous articles within it, so you have one entry for the issue itself, and others referring to it. Perhaps there are other cases, but I think what's right here is to give the crossref and only the pages.

```
    { "In " make.crossref * space output
      make.pages "pages" check comma output
    }
  if$
  new.block
  note space output
  finish.entry
}
```

3.12.2 Book

book

```
FUNCTION {book}
{ begin.entry
  author empty$
    { make.editors.primary "author and editor" check comma output
    }
    { make.authors comma output
    }
  if$
  new.block
  make.year.or.oyear.month "year" check comma output
  new.block
  make.title.ital "title" check comma output
  new.block
  make.edition comma output
  new.block
  crossref empty$
    { volume empty$
```

```
        { output.series.number
        }
        { output.volume.series
        }
      if$
      editor empty$ author empty$ or
        'skip$
        { new.block
          make.editors.secondary comma output
        }
      if$
      new.block
      make.translators comma output
      new.block
      make.illustrators comma output
      new.block
      oyear empty$
        { address comma output
          publisher "publisher" check colon output
        }
        { oaddress comma output
          opublisher "opublisher" check colon output
          new.block
          "Reprint" comma output
          address field.or.null comma output
          publisher "publisher" check
          address empty$
            { comma
            }
            { colon
            }
          if$
          output
          year "year" check comma output
        }
      if$
    }
    { volume empty$
        { "In "
        }
        { "Volume" volume tie.or.space.connect
          " of " *
        }
      if$
      make.crossref * comma output
    }
  if$
  new.block
  yearcomp empty$
    'skip$
    { "Composed in " yearcomp *
      comma output
      new.block
    }
```

```
  if$
```

The ‘’ fyear must be nonempty to get any information about the first-published edition,

The ‘’ fpublisher, ‘’ faddress, and ‘’ flanguage can be empty in any combination. They are all ignored if ‘’ fyear is empty, except for possibly ‘’ flanguage, which will be given with the translator if ‘’ translator is nonempty.

```
  fyear empty$
    'skip$
    { "First published "
      translator empty$ flanguage empty$ not and
        { "in " * flanguage * space *
        }
        'skip$
      if$
```

If ‘’ ftitle is empty, there will be a harmless? extra space.

```
      make.as.ftitle.ital * space output
      faddress empty$ fpublisher empty$ not and
        { "by "
        }
        { "in "
        }
      if$
      space output
      faddress space output
      fpublisher
      faddress empty$
        { space
        }
        { colon
        }
      if$
      output
      fyear
      faddress empty$ fpublisher empty$ and
        { space
        }
        { comma
        }
      if$ output
      new.block
    }
  if$                       %$
  note comma output
  finish.entry
}
```

3.12.3 Inbook

inbook The *Chicago Manual* doesn't seem to recognize parts of a book without their own titles. We make up our own version, moving the chapter and page information from the end of the ‘’ booktitle to the front.

```
FUNCTION {inbook}
```

```
{ begin.entry
  author empty$
    { make.editors.primary "author and editor" check comma output
    }
    { make.authors comma output
    }
  if$
  new.block
  make.year.or.oyear.month "year" check comma output
  new.block
```

We store an extra copy of 'make.chapter.of' in the variable 'u' here, so that we know whether to put pages or not after the '' title sentence (with editors, and so on).

```
  crossref empty$
    { make.chapter.of duplicate$ 'u :=
      duplicate$ empty$
        { pop$
          pages empty$
            { "empty chapter, type, and pages in " cite$ * warning$
              "In "
            }
            { "Pages " make.pages * " in " *
            }
          if$
        }
        'skip$
      if$
      comma output
      make.title.ital "title" check space output
      make.bookauthors.secondary comma output
      make.editors.secondary comma output
      volume empty$
        { output.series.number
        }
        { output.volume.series
        }
      if$
      make.edition comma output
```

Here we branch again on the contents of 'make.chapter.of'. If it is not empty, then we go ahead and 'make.pages'. If it is empty we don't, because this means that if '' pages were given, they have already been put earlier.

```
      u empty$
        'skip$
        { make.pages comma output
        }
      if$
      new.block
      make.translators comma output
      new.block
      make.illustrators comma output
      new.block
      oyear empty$
        { address comma output
```

```
          publisher "publisher" check colon output
        }
        { oaddress comma output
          opublisher "opublisher" check colon output
          new.block
          "Reprint" comma output
          address field.or.null comma output
          publisher "publisher" check
          address empty$
            { comma
            }
            { colon
            }
          if$
          output
          year "year" check comma output
        }
      if$
    }
    { "In " make.crossref * comma output
      make.chapter comma output
      make.pages comma output
      chapter empty$ pages empty$ and
        { "empty chapter and pages in " cite$ * warning$
        }
        'skip$
      if$
    }
  if$
  new.block
  note comma output
  finish.entry
}
```

3.12.4 Incollection

incollection

```
FUNCTION {incollection}
{ begin.entry
  author empty$
    { make.editors.primary "author and editor" check space output
    }
    { make.authors space output
    }
  if$
  new.block
  make.year.or.oyear.month "year" check space output
  new.block
  make.title.doublequote "title" check space output
  new.block
  crossref empty$
    { make.chapter.of duplicate$ empty$
        { pop$
          "In "
```

```
        }
        'skip$
      if$
      space output
      make.booktitle.ital "booktitle" check space output
      make.bookauthors.secondary comma output
      make.editors.secondary comma output
      volume empty$
        { output.series.number
        }
        { output.volume.series
        }
      if$
      make.edition comma output
% FIX: do I need this?  I don't for copleston:cusa
%      make.year.or.oyear.month "year" check comma output
      make.pages comma output
      chapter empty$ type empty$ and pages empty$ and
        { "empty chapter, type, and pages in " cite$ * warning$
        }
        'skip$
      if$
      new.block
      make.translators space output
      new.block
      make.illustrators space output
      new.block
      oyear empty$
        { address space output
          publisher "publisher" check colon output
        }
        { oaddress space output
          opublisher "opublisher" check colon output
          new.block
          "Reprint" comma output
          address comma output
          publisher "publisher" check
          address empty$
            { comma
            }
            { colon
            }
          if$
          output
          year comma output
        }
      if$
    }
    { new.block
      "In " make.crossref * space output
      make.chapter comma output
      make.pages comma output
      chapter empty$ pages empty$ and
        { "empty chapter and pages in " cite$ * warning$
```

```
        }
        'skip$
      if$
    }
  if$
  new.block
  note space output
  finish.entry
}
```

3.12.5 Majorthesis

format.thesis
thesis
phdthesis
mastersthesis
majorthesis
minorthesis

Theses are all set in the same way, with a different type name. An entry with entry type 'thesis' ought to have a nonempty "" type. The other four alias entry types simply supply an explicit value for "" type, which are overridden by a nonempty "" type.

FIX The "" type field should be hooked into a variable and done with the *babel* package: `\majorthesisname` and so on.

```
FUNCTION {format.thesis}
{ 'u :=
  begin.entry
  make.authors "author" check key.q space output
  new.block
  make.year.month "year" check space output
  new.block
  make.title.doublequote "title" check space output
  new.block
  u "type" check space output
  school "school" check comma output
  address comma output
  new.block
  note space output
  finish.entry
}
FUNCTION {thesis}
{ type field.or.null format.thesis
}
FUNCTION{mastersthesis}
{ type empty$
    { "Master's thesis"
    }
    { type
    }
  if$
  format.thesis
}
FUNCTION {minorthesis}
{ mastersthesis
}
FUNCTION {phdthesis}
{ type empty$
    { "Ph.D.~diss."
    }
```

```
    { type
    }
  if$
  format.thesis
}
FUNCTION {majorthesis}
{ phdthesis
}
```

3.12.6 Manual

`manual` Technical documentation. Required: ‘’ title (italicized). Optional: ‘’ author or ‘’ organization or ‘’ key, ‘’ address, ‘’ edition, ‘’ month, ‘’ year, ‘’ note. Logic: lead item is first nonempty field of ‘’ author, ‘’ organization, and ‘’ key. Subsequent nonempty fields in that list are ignored.

```
FUNCTION {manual}
{ begin.entry
  printed.name space output
  new.block
  make.year.month space output
  new.block
  make.title.ital "title" check comma output
  new.block
  make.edition comma output
  new.block
  address space output
  organization colon output
  new.block
  note space output
  finish.entry
}
```

3.12.7 Booklet

`booklet` A work that is printed and bound, but without a named publisher or sponsoring institution. Required: ‘’ title. Optional: ‘’ author, ‘’ howpublished, ‘’ address, ‘’ month, ‘’ year, ‘’ note. Title is quoted. ‘’ key used if ‘’ author is empty.

```
FUNCTION {booklet}
{ begin.entry
  make.authors key.q comma output
  new.block
  make.year.month space output
  new.block
  make.title.doublequote "title" check comma output
  new.block
  howpublished space output
  address comma output
  new.block
  note space output
  finish.entry
}
```

3.12.8 Inproceedings

inproceedings
conference

An article in a conference proceedings. Required: ‘’ author, ‘’ title, ‘’ booktitle, ‘’ year. Optional: ‘’ editor, ‘’ volume or ‘’ number, ‘’ series, ‘’ pages, ‘’ address, ‘’ month, ‘’ organization, ‘’ publisher, ‘’ note. ‘’ key used where ‘’ author empty. Title quoted.

```
FUNCTION {inproceedings}
{ begin.entry
  make.authors "author" check key.q space output
  new.block
  make.year.month "year" check space output
  new.block
  make.title.doublequote "title" check space output
  new.block
  crossref empty$
    { make.editors.secondary space output
      make.booktitle.ital "booktitle" check comma output
      volume empty$
        { output.series.number
        }
        { output.volume.series
        }
      if$
      make.edition comma output
      new.block
      organization space output
      address space output
      publisher colon output
      make.pages comma output
    }
    { "In " make.crossref * space output
      make.chapter comma output
      make.pages comma output
      chapter empty$ pages empty$ and
        { "empty chapter and pages in " cite$ * warning$
        }
        'skip$
      if$
    }
  if$
  new.block
  note space output
  finish.entry
}
FUNCTION {conference}
{ inproceedings
}
```

3.12.9 Proceedings

proceedings

The proceedings of a conference. Required: ‘’ title, ‘’ year. Optional: ‘’ editor, ‘’ volume or ‘’ number, ‘’ series, ‘’ address, ‘’ month, ‘’ organization, ‘’ publisher, ‘’ note. ‘’ key used for empty ‘’ editor. Title italicized.

```
FUNCTION {proceedings}
{ begin.entry
  make.editors.primary key.q space output
  new.block
  make.year.month "year" check space output
  new.block
  make.title.ital "title" check comma output
  new.block
  make.edition comma output
  new.block
  volume empty$
    { output.series.number
    }
    { output.volume.series
    }
  if$
  new.block
  organization space output
  address comma output
  publisher colon output
  new.block
  note comma output
  finish.entry
}
```

3.12.10 Misc

Required: *none*. The rest are optional. First ‘’ author (‘’ key if empty). Then ‘’ year with ‘’ month. Then ‘’ title, quoted. Then ‘’ howpublished. Then ‘’ note.

```
FUNCTION {misc}
{ begin.entry
  make.authors key.q space output
  new.block
  make.year.month space output
  new.block
  title space output
  new.block
  howpublished comma output
  new.block
  note space output
  finish.entry
}
```

3.12.11 Techreport

`techreport` A report published by a school or other institution, usually numbered within a series. Required: ‘’ author, ‘’ title, ‘’ institution, ‘’ year. Optional: ‘’ type, ‘’ number, ‘’ address, ‘’ month, ‘’ note. ‘’ key used when ‘’ author empty.

```
FUNCTION {techreport}
{ begin.entry
  make.authors "author" check key.q comma output
  new.block
  make.year.month "year" check comma output
```

```
  new.block
  make.title.doublequote "title" check comma output
  new.block
  make.number.tr comma output
  institution "institution" check comma output
  address comma output
  new.block
  note comma output
  finish.entry
}
```

3.12.12 Unpublished

unpublished A document having an author and title, but not formally published. Required: ‘’ author, ‘’ title, ‘’ note. Optional: ‘’ month, ‘’ year. Uses ‘’ key for no ‘’ author.

```
FUNCTION {unpublished}
{ begin.entry
  make.authors "author" check key.q space output
  new.block
  make.year.month space output
  new.block
  make.title.doublequote "title" check space output
  new.block
  note "note" check space output
  finish.entry
}
```

3.12.13 Default

default.type

```
FUNCTION {default.type}
{ misc }
```

3.13 Sorting

sortify Prepare the string on the stack for sort comparison \stack⟨*string*⟩⟨*string*⟩

```
FUNCTION {sortify}
{ purify$
  "l" change.case$
}
```

chop.word \stack⟨*string*⟩ ⟨*num*⟩ ⟨*string*⟩⟨*string*⟩

```
% stack: S n T
%
% if S = substr(1,n,T)
%    then substr(n+1, max, T)
%    else S
%
% if T begins with S, then chop it off the beginning of T.
%
```

```
FUNCTION {chop.word}
{ 't :=
  'len :=
  t #1 len substring$ =
    { t len #1 + global.max$ substring$
    }
    't
  if$
}
```

`sort.format.names` `\stack`⟨*list of names*⟩⟨*list of names suitable for sorting*⟩

```
FUNCTION {sort.format.names}
{ 's :=
  #1 'nameptr :=
  ""
  s num.names$ 'numnames :=
  numnames 'namesleft :=
    { namesleft #0 >
    }
    { nameptr #1 >
        { "   " *
        }
        'skip$
      if$
      s nameptr "{vv{ } }{ll{ }}{  f{ }}{  jj{ }}" format.name$ 't :=
      nameptr numnames = t "others" = and
        { etal *
        }
        { t sortify *
        }
      if$
      nameptr #1 + 'nameptr :=
      namesleft #1 - 'namesleft :=
    }
  while$
}
```

`make.title.sort` FIX: explain why we do all this.

```
FUNCTION {make.title.sort}
{ title field.or.null
  't :=
  "A " #2
    "An " #3
      "The " #4 t chop.word
    chop.word
  chop.word
  sortify
  #1 global.max$ substring$              %$
}
```

`make.author.sort` Put an author-sort string on the stack. If there is no ‘’ author field, use the ‘’ key field.

```
FUNCTION {make.author.sort}
{ author empty$
```

```
    { key empty$
        { "to sort, need author or key in " cite$ * warning$
          ""
        }
        { key sortify
        }
      if$
    }
    { author sort.format.names
    }
  if$                       %$
}
```

make.author.editor.sort Put an author-or-editor sort string on the stack. If there is no ‘’ author field, use the ‘’ editor field. If there is no ‘’ editor field either, use the ‘’ key field.

```
FUNCTION {make.author.editor.sort}
{ author empty$
    { editor empty$
        { key empty$
            { "to sort, need author, editor, or key in " cite$ * warning$
              ""
            }
            { key sortify
            }
          if$
        }
        { editor sort.format.names
        }
      if$
    }
    { author sort.format.names
    }
  if$                   %$
}
```

make.author.organization.sort Put an author-or-organization-sort string on the stack. If there is no ‘’ author field, use the ‘’ organization field. If there is no ‘’ organization field either, use the ‘’ key field.

```
FUNCTION {make.author.organization.sort}
{ author empty$
    { organization empty$
        { key empty$
            { "to sort, need author, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify
            }
          if$
        }
        { organization sortify
        }
      if$
    }
    { author sort.format.names
```

```
    }
  if$
}
```

make.editor.organization.sort Put an editor-or-organization-sort string on the stack. If there is no ‘’ editor field, use the ‘’ organization field. If there is no ‘’ organization field either, use the ‘’ key field.

```
FUNCTION {make.editor.organization.sort}
{ editor empty$
    { organization empty$
        { key empty$
            { "to sort, need editor, organization, or key in " cite$ * warning$
              ""
            }
            { key sortify
            }
          if$
        }
        { organization sortify
        }
      if$
    }
    { editor sort.format.names
    }
  if$                %$
}
```

presort ***To do:*** *confirm this with the manual, 15.65.*

Set labels ‘long.label’, ‘short.label’, and ‘year.label’. Set `sort.key$` to ‘long.label’ + ‘year.label’ + X + title, where X is a string involving the author, editor, organization, or key, depending on the entry type.

Note that ‘year.tag.label’ is not set yet.

FIX: Why do we insert four spaces between those elements?

```
FUNCTION {presort}
{ calc.labels
  long.label sortify
  "    " *
  year.label *
  "    " *
  type$ "book" = type$ "inbook" = or
    'make.author.editor.sort
    { type$ "proceedings" =
        'make.editor.organization.sort
        { type$ "manual" =
            'make.author.organization.sort
            % default case:
            'make.author.sort
          if$
        }
      if$
    }
  if$
  *
```

```
  "    "  *
  make.title.sort *
  entry.clip
  'sort.key$ :=
}
```

initialize.extra.label.stuff

```
FUNCTION {initialize.label.vars}
{ unidentified 'last.long.label :=
  unidentified 'last.long.year.label :=
  "" 'following.year.tag.label :=
  #0 'last.year.tag.num :=
}
```

forward.pass
make.printed.name

Pass through the new list, comparing current entry's 'long.label' plus 'year.label' to the last entry's, stored in the global variable 'last.long.year.label'. If they are the same, increment 'last.year.tag.num' and set 'year.tag.label' to the nth lowercase letter, where n is the global integer variable 'last.year.tag.num'. If they are not the same, set 'last.long.year.label' to the current 'long.label' plus 'year.label', set 'year.tag.label' to nil, and set 'last.year.tag.num' to 0.

After this pass, 'year.tag.label' will be blank for all first entries, and "b," "c," etc. for second entries, third entries, and so on. The "a"'s will be added in the right places by the reverse pass.

We also use the forward pass to set the proper value of 'printed.name'. Let `S` be the value of the field indicated by 'printed.name.type'. Then 'printed name will be `\SCduplicate{S}` whenever the current 'long.label' is the same as the global variable 'last.long.label', or simply a copy of `S` otherwise.

```
FUNCTION {make.printed.name}
{ printed.name.type #0 =
    { author field.or.null format.names.lastfirst
    }
    { printed.name.type #1 =
        { editor field.or.null format.names.lastfirst
        }
        { printed.name.type #2 =
            { organization field.or.null
            }
            { key field.or.null
            }
          if$
        }
      if$
    }
  if$
}

FUNCTION {forward.pass}
{ long.label year.label * entry.clip
  duplicate$ last.long.year.label =
     % same as last entry
     { pop$
       last.year.tag.num #1 + 'last.year.tag.num :=
       last.year.tag.num int.to.chr$ 'year.tag.label :=
```

```
    }
    % different from last entry
    { "a" chr.to.int$ 'last.year.tag.num :=
      "" 'year.tag.label :=
      'last.long.year.label :=
    }
  if$
```

FIX: Handle repeated `''` organization and `''` key.

Above, we compared the current entry's 'long.label' plus 'year.label' to the last entry's. Here we just compare 'long.label'. FIX: Hmm, I don't think 'long.label' is ever going to be 'unidentified' here. I'm commenting out this logic.

```
  long.label entry.clip
%  duplicate$ unidentified =
%    { pop$
%      unidentified 'last.long.label :=
%      make.printed.name 'printed.name :=
%    }
%    {
       duplicate$ last.long.label =
        { pop$
          "\SCduplicate{" make.printed.name * "}" *
        }
        { 'last.long.label :=
           make.printed.name
        }
      if$
      'printed.name :=
%    }
%  if$
}
```

`reverse.pass` Pass through the entries in reverse order. Set current 'year.tag.label' to "a" on entries that precede ones with "b"s. Set 'following.year.tag.label' to the current 'year.tag.label'.

```
FUNCTION {reverse.pass}
{ following.year.tag.label "b" =
    { "a" 'year.tag.label :=
    }
    'skip$
  if$
  year.tag.label 'following.year.tag.label :=
}
```

`begin.bib` `end.bib` These two functions are called to begin and end the entire bibliography.

```
FUNCTION {begin.bib}
{ preamble$ empty$
    'skip$
    { preamble$ write$ newline$
    }
  if$
  "\begin{thebibliography}{}" write$ newline$
}
FUNCTION {end.bib}
```

```
{ newline$
  "\end{thebibliography}" write$ newline$
}
```

3.14 Actions

This is the main subroutine which calls all the others.

```
READ

EXECUTE {init.state.consts}

ITERATE {presort}
SORT

EXECUTE {initialize.label.vars}

ITERATE {forward.pass}
REVERSE {reverse.pass}

EXECUTE {begin.bib}
ITERATE {call.type$}
EXECUTE {end.bib}
```

References

Abrams, M. H. 1973. *Natural Supernaturalism: Tradition and Revolution in Romantic Literature*. New York: W. W. Norton and Co.

Abrams, M. H. et al., eds. 1979. *The Norton Anthology of English Literature*. 4th ed. 2 vols. New York: W. W. Norton and Co.

Acheson, James. 1979. "Murphy's Metaphysics." *Journal of Beckett Studies*, no. 5:9–23.

Ackerley, Chris. 1997. "'Do Not Despair': Samuel Beckett and Robert Greene." *Journal of Beckett Studies,* n.s., 6 (1): 119–24.

Alighieri, Dante. 1961. *The Purgatorio*. Translated from the Italian by John Ciardi. New York: New American Library.

———. 1970. *The Paradiso*. Translated from the Italian by John Ciardi. New York: New American Library.

———. 1984. *Purgatorio: A Verse Translation*. Bantam Classic ed. Volume 2 of *The Divine Comedy*. Translated from the Italian by Allen Mandelbaum. Illustrated by Barry Moser. New York: Bantam Books. Notes by Laury Magnus, Allen Mandelbaum, and Anthony Oldcorn, with Daniel Feldman.

Amiran, Eyal. 1993. *Wandering and Home: Beckett's Metaphysical Narrative*. University Park: Pennsylvania State University Press.

Anspaugh, Kelly. 1996. "'Faith, Hope, and—What Was It?': Beckett Reading Joyce Reading Dante." *Journal of Beckett Studies,* n.s., 5 (1–2): 19–38.

Bair, Deirdre. 1990. *Samuel Beckett: A Biography*. New York: Summit Books. First published in 1978 by Harcourt Brace Jovanovich (New York) and Jonathan Cape, but this edition does not appear, technically, to be a reprint.

Beckett, Samuel. 1954. *Waiting for Godot*. First Evergreen ed. Translated from the French by the author. New York: Grove Press. Reprint, New York: Grove Press, 1956. Composed in 1948. First published as *En attendant Godot* in 1952. Premiered in French in 1953, in English in 1955.

———. 1956a, 7 April. Letter to Barney Rosset of Grove Press in the Special Collections of Syracuse University Library, Syracuse, New York.

———. 1956b. Pages 108–19 in *Yellow*, Volume 10 of *New American Library Mentor Books*. New York: New American Library. First published (with minor differences) in Beckett 1972.

———. 1972. *More Pricks Than Kicks*. New York: Grove Press. First published by Chatto and Windus, 1934.

———. 1986. *Happy Days: The Production Notebook of Samuel Beckett*. Edited by James Knowlson. New York: Grove Press. First published in London: Faber and Faber, 1985.

———. 1992a. *Endgame: With a Revised Text*. Volume 2 of *The Theatrical Notebooks of Samuel Beckett*. Edited by S. E. Gontarski. London: Faber and Faber.

———. 1992b. *Krapp's Last Tape: With a Revised Text.* Volume 3 of *The Theatrical Notebooks of Samuel Beckett.* Edited by James Knowlson. New York: Grove Press.

———. 1994. *Waiting for Godot: With a Revised Text.* Volume 1 of *The Theatrical Notebooks of Samuel Beckett.* Edited by Dougald McMillan and James Knowlson. New York: Grove Press. First published in Great Britain: Faber and Faber, 1993.

———. 1995. *Samuel Beckett: The Complete Short Prose, 1929–1989.* Edited by S. E. Gontarski. New York: Grove Press.

Blake, William. 1966. *Blake's Job: William Blake's* Illustrations of the Book of Job. Edited by S. Foster Damon. University Press of New England.

Bloom, Harold. 1973. *The Anxiety of Influence: A Theory of Poetry.* New York: Oxford University Press.

———. 1988. "Introduction." In *Samuel Beckett's* Molloy*,* Malone Dies*,* The Unnamable, edited by Harold Bloom, Modern Critical Interpretations, 1–12. New York: Chelsea House Publishers.

Borges, Jorge Luis. 1963. *Ficciones.* Edited by Anthony Kerrigan. Translated from the Spanish by the editor et al. New York: Grove Weidenfeld. First published in Buenos Aires: Emecé Editores, 1956.

Brater, Enoch. 1989. *Why Beckett.* New York: Thames and Hudson.

Bryden, Mary. 1995. "Beckett and the Three Dantean Smiles." *Journal of Beckett Studies,* n.s.*,* 4 (2): 29–33.

———. 1998. *Samuel Beckett and the Idea of God.* New York: St. Martin's Press.

Buning, Marius. 1990. "Samuel Beckett's Negative Way: Intimations of the *Via Negativa* in his Late Plays." Chapter 9 of *European Literature and Theology in the Twentieth Century: Ends of Time*, edited by David Jasper and Colin Crowder, 129–42. New York: St. Martin's Press.

Burrows, Rachel. 1989. "Interview with Rachel Burrows, Dublin, Bloomsday, 1982." *Journal of Beckett Studies*, no. 11–12:6–15. Interviewed by S. E. Gontarski, Martha Feshenfeld, and Dougald McMillan.

Calder, John. 1979. "Review of *Samuel Beckett: A Biography* by Deirdre Bair (Jonathan Cape, 1978)." *Journal of Beckett Studies*, no. 4:74–79.

Carey, Phyllis. 1992. "Stephen Dedalus, Belacqua Shuah, and Dante's *Pietà*." In Carey and Jewinski 1992b, Chapter 8, 104–16.

Carey, Phyllis and Ed Jewinski. 1992a. "Introduction." In Carey and Jewinski 1992b, xi–xviii.

———, eds. 1992b. *Re: Joyce'n Beckett.* New York: Fordham University Press.

Carroll, John S. 1906. *Prisoners of Hope: an Exposition of Dante's* Purgatorio. London: Hodder and Stoughton.

Cassedy, Steven. 1988. "Mathematics, Relationism, and the Rise of Modern Literary Aesthetics." *Journal of the History of Ideas* 49 (1): 109–32.

Coetzee, J. M. 1972. "The Manuscript Revisions of Beckett's *Watt*." *Journal of Modern Literature* 2:472–80.

Cohn, Ruby, ed. 1975. *Samuel Beckett: A Collection of Criticism*. New York: McGraw-Hill.

———. 1980. *Just Play: Beckett's Theater*. Princeton: Princeton University Press.

Common Prayer. 1789. *The Book of Common Prayer and Administration of the Sacraments and Other Rites and Ceremonies of the Church*. New York. "This edition . . . was . . . reproduced from a certified edition . . . as adopted by the General Convention of 1928 and emended by subsequent Conventions."

Connor, Steven. 1982. "Beckett's Animals." *Journal of Beckett Studies*, no. 8:29–44.

Copleston, Frederick. 1950. "The Pseudo-Dionysius." Chapter 9 of *Augustine to Scotus*, Volume 2 of *A History of Philosophy*. 9 vols. in 3 books. Newman Press. Reprint, New York: Image Books, 1985.

———. 1953. "Nicholas of Cusa." Chapter 15 of *Ockham to Suarez*, Volume 3 of *A History of Philosophy*. 9 vols. in 3 books. Newman Press. Reprint, New York: Image Books, 1985.

Cronin, Anthony. 1996. *Samuel Beckett: The Last Modernist*. London: HarperCollins.

Culik, Hugh. 1983. "The Place of *Watt* in Beckett's Development." *Modern Fiction Studies* 29 (1): 57–71 (spring).

———. 1993. "Mathematics as Metaphor: Samuel Beckett and and the Esthetics of Incompleteness." *Papers on Language & Literature* 29 (2): 131–51 (spring).

Davies, Paul. 1994. *The Ideal Real: Beckett's Fiction and Imagination*. Associated University Presses.

Dearlove, J. E. 1977. "'Last Images': Samuel Beckett's Residual Fiction." *Journal of Modern Literature* 6 (1): 104–126 (February).

Donne, John. 1980. *Paradoxes and Problems*. Edited by Helen Peters. Oxford: Clarendon Press.

Driver, Tom F. 1961. "Beckett by the Madeleine." *The Columbia University Forum* 4 (3): 21–25 (summer).

Eliot, George. 1984. *Middlemarch*. New York: Random House.

Ellis, Reuben J. 1989. "'Matrix of Surds': Heisenberg's Algebra in Beckett's *Murphy*." *Papers on Language & Literature* 25 (1): 120–23 (winter).

Ellmann, Richard. 1982. *James Joyce*. New and Rev. ed. New York: Oxford University Press. First published in 1959.

Ellmann, Richard, Robert O'Clair, et al., eds. 1988. *Norton Anthology of Modern Poetry*. 2d ed. New York: W. W. Norton and Co.

Empson, William. 1966. *Seven Types of Ambiguity*. 3d ed. New York: New Directions.

Euclid. 1952. *The Thirteen Books of Euclid's Elements*. Volume 11 of *Great Books of the Western World*. Translated by Sir Thomas L. Heath. Chicago: Encylopaedia Britannica.

Farrow, Anthony. 1991. *Early Beckett: Art and Allusion in* More Pricks Than Kicks *and* Murphy. Troy, New York: Whitston Publishing Company.

Federman, Raymond and John Fletcher. 1970. *Samuel Beckett: His Works and His Critics; An Essay in Bibliography.* Berkeley: University of California Press.

Fergusson, Francis. 1953. *Dante's Drama of the Mind: A Modern Reading of the* Purgatorio. Princeton University Press.

Fletcher, John. 1967. *Samuel Beckett's Art.* London: Chatto and Windus.

Flexner, Stuart Berg et al., eds. 1987. *The Random House Dictionary of the English Language.* 2d ed. New York: Random House.

Freud, Sigmund. 1961. *Beyond the Pleasure Principle.* Edited by James Strachey. Translated from the German by James Strachey. New York: Liveright. Reprint, New York: W. W. Norton and Co., 1990. First published in 1920. On p. xxxvii: "The present translation is a corrected reprint of the *Standard Edition* version" (i.e., English translation of 1955).

Frye, Northrop. 1957. "Anatomy of Criticism." In *Criticism: The Major Texts*, edited by Walter Jackson Bate, Enl. ed., 609–25. San Diego: Harcourt Brace Jovanovich.

Gardner, Martin. 1979. Chapter 10 of *Cyclic Numbers*, 111–22. New York: Alfred A. Knopf.

Gascoigne, George. 1907. *The Posies.* Volume 1 of *The Complete Works of George Gascoigne.* 2 vols. Edited by John W. Cunliffe. Grosse Pointe, Michigan: Scholarly Press. Reprint, Cambridge: Cambridge University Press, 1969.

Gates, David. 1996. "He Couldn't Go On, He Went On." *The New York Times Book Review*, 26 May, 4.

Gleitman, Henry. 1986. *Psychology.* 2d ed. New York: W. W. Norton and Co.

Gontarski, S. E., ed. 1985. *The Intent of Undoing in Samuel Beckett's Dramatic Texts.* Bloomington: Indiana University Press.

Gove, Philip Babcock et al., eds. 1993. *Webster's Third New International Dictionary of the English Language, Unabridged: A Merriam-Webster.* Springfield, Massachusetts: Merriam-Webster.

Green, David. 1994. "A Note on Augustine's Thieves." *Journal of Beckett Studies,* n.s., 3 (2): 77–78.

——— 1996. "Beckett's *Dream*: More Niente than Bel." *Journal of Beckett Studies,* n.s., 5 (1–2): 67–80.

Greene, Robert. n.d. [1881–86]. "The Repentance of Robert Greene." In *The Life and Complete Works in Prose and Verse of Robert Greene*, edited by Alexander B. Grosart, Volume 12 of *Huth Library*, 15 vols., 151–88. London and Aylesbury: printed for private circulation. Reprint, New York: Russell and Russell, 1964.

Gregory, Richard L., ed. 1987. *The Oxford Companion to the Mind.* Oxford: Oxford University Press.

Harper, George Mills. 1992. "Review of *The Irish Beckett* by John P. Harrington (Syracuse: Syracuse University Press, 1991)." *Journal of Beckett Studies,* n.s., 2 (1): 131–34.

Harrington, John P. 1992. "A Note on *Malone Dies* and Local Phenomena." *Journal of Beckett Studies,* n.s., 1 (1–2): 141–43.

Harvey, Lawrence E. 1970. *Samuel Beckett: Poet and Critic*. Princeton: Princeton University Press.

Heraclitus. 1970. *The Cosmic Fragments*. Edited by G. S. Kirk. Cambridge: Cambridge University Press.

Hill, Geoffrey. 1989. "A Pharisee to Pharisees: Reflections on Vaughan's 'The Night.'" *English* 38 (summer): 97–113.

Hobson, Harold. 1956. "Samuel Beckett, Dramatist of the Year." *International Theater Annual*, no. 1:153–55.

Hopkins, Gerard Manley. 1953. *Poems and Prose*. Edited by W. H. Gardner. PenGuin Books.

Iser, Wolfgang. 1985. "The Pattern of Negativity in Beckett's Prose." In *Samuel Beckett*, edited by Harold Bloom, Modern Critical Views, 125–36. New York: Chelsea House Publishers.

Israel, Calvin. 1979. "Review of *Samuel Beckett: A Biography* by Deirdre Bair (New York: Harcourt Brace Jovanovich, 1978)." *Journal of Beckett Studies*, no. 4:80–85.

James, William. 1890. *The Principles of Psychology*. 2 vols. New York: Henry Holt and Co. Reprint, New York: Dover Publications, 1950.

Jewinski, Ed. 1992. "James Joyce and Samuel Beckett: From Epiphany to Anti-Epiphany." In Carey and Jewinski 1992b, Chapter 11, 160–74.

Johnson, Mark. 1987. *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason*. Chicago: University of Chicago Press.

Jones, Anthony. 1981. "The French Murphy: From 'rare bird' to 'cancre.'" *Journal of Beckett Studies*, no. 6:37–50.

Joyce, James. 1939. *Finnegans Wake*. New York: PenGuin Books.

———. 1968. *Dubliners*. Rev. ed. Viking. Reprint, New York: Penguin Books, 1976. First published in 1916 (USA).

———. 1986. *Ulysses: The Corrected Text*. First Vintage Books ed. Edited by Hans Walter Gabler et al. New York: Random House. First published in 1922.

———. 1993. *A Portrait of the Artist as a Young Man*. Case Studies in Contemporary Criticism. Edited by R. B. Kershner. Boston: Bedord Books of St. Martin's Press.

Juliet, Charles. 1995. *Conversations with Samuel Beckett and Bram van Velde*. Translated from the French by Janey Tucker. Academic Press Leiden. First published as *Rencontres avec Bram van Velde and Rencontre avec Samuel Beckett* in Montpellier: Editions Fata Morgana, 1973, 1986.

Jung, C. G. 1966. "Psychology and Literature." In *The Collected Works of C. G. Jung*, edited by Sir Herbert Read, Michael Fordham, and Gerhard Adler, Volume 15, 84–105. Translated by R. F. C. Hull. Princeton University Press.

——— 1968a. *Analytical Psychology: Its Theory and Practice*. The Tavistock Lectures. New York: Vintage Books.

——— 1968b. "Lecture 3." In Jung 1968a, 78–113.

Kennedy, Sighle. 1971. *Murphy's Bed: A Study of Real Sources and Sur-real Associations in Samuel Beckett's First Novel*. Lewisburg, Pennsylvania: Bucknell University Press.

Kenner, Hugh. 1968. *Samuel Beckett: A Critical Study*. New ed. Berkeley: University of California Press.

———. 1975. "Shades of Syntax." In Cohn 1975, 21–31.

Knowlson, James. 1996. *Damned to Fame: The Life of Samuel Beckett*. New York: Simon and Schuster.

Knowlson, James and John Pilling. 1979. *Frescoes of the Skull: The Later Prose and Drama of Samuel Beckett*. London: John Calder.

Koffka, K. 1935. *Principles of Gestalt Psychology*. New York: Harcourt, Brace and World.

Krance, Charles. 1983. "Odd Fizzles: Beckett and the Heavenly Sciences." *Bucknell Review* 27 (2): 96–107.

Kroll, Jeri L. 1978. "Belacqua as Artist and Lover: 'What a Misfortune.'" *Journal of Beckett Studies*, no. 3:10–39.

Lakoff, George. 1987. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. Chicago: University of Chicago Press.

Lakoff, George and Mark Johnson. 1980. *Metaphors We Live By*. Chicago: University of Chicago Press.

Lees, Heath. 1984. "*Watt*, Music, Tuning and Tonality." *Journal of Beckett Studies*, no. 9:5–24.

Levy, Jay A. 1992. "Conversations with Samuel Beckett." *The American Scholar* 61 (1): 124–31 (winter).

Linden, Stanton J. 1978. "Herbert and the Unveiling of Diana: Stanza Three of 'Vanitie.'" *The George Herbert Journal* 1 (spring): 30–37.

Lippman, Carlee. 1993. "Proprioceptive Deficit in Beckett's 'Not I.'" *Journal of Beckett Studies,* n.s., 2 (2): 81–83.

Lodge, David. 1968. "Some Ping Understood." *Encounter* 30 (2): 85–89 (February). Also in McCarthy 1986, 119–27.

Marculescu, Ileana. 1989. "Beckett and the Temptation of Solipsism: '*Esse est aut percipere aut percipi*.'" *Journal of Beckett Studies*, no. 11–12:53–64.

McCarthy, Patrick A., ed. 1986. *Critical Essays on Samuel Beckett*. Critical Essays on Modern British Literature. Boston: G. K. Hall and Co.

McHugh, Roland. 1991. *Annotations to* Finnegans Wake. Rev. ed. Baltimore: Johns Hopkins University Press.

McMillan, Dougald. 1975. "Samuel Beckett and the Visual Arts: The Embarrassment of Allegory." In Cohn 1975, 121–35.

Melville, Herman. 1988. *Moby-Dick, or The Whale*. Volume 6 of *The Writings of Herman Melville*. 15 vols. Edited by Harrison Hayford, Hershel Parker, and G. Thomas Tanselle. Evanston, Illinois: Northwestern University Press and The Newberry Library.

Merwin, W. S. 1970. *The Carrier of Ladders*. New York: Atheneum.

Milton, John. 1998. "Paradise Lost." In *The Riverside Milton*, edited by Roy Flannagan, 296–710. Boston: Houghton Mifflin.

Miskinis, Steven. 1996. "Enduring Recurrence: Samuel Beckett's Nihilistic Poetics." *ELH* 63 (4): 1047–1067 (winter).

Montgomery, Angela. 1991. "Beckett and Science: *Watt* and the Quantum Universe." *Comparative Criticism* 13:171–81.

Mood, John J. 1971. "'The Personal System'—Samuel Beckett's *Watt*." *Publications of the Modern Language Association of America* 86 (2): 255–65.

Morris, William, ed. 1981. *The American Heritage Dictionary of the English Language*. Boston: Houghton Mifflin.

Murphy, P. J., Werner Huber, Rolf Breuer, and Konrad Schoell. 1994. *Critique of Beckett Criticism: A Guide to Research in English, French, and German*. Literary Criticism in Perspective. Columbia: Camden House.

Niven, Ivan Morton, Herbert S. Zuckerman, and Hugh L. Montgomery. 1991. *Theory of Numbers*. 5th ed. New York: Wiley.

O'Brien, Eoin. 1986. *The Beckett Country: Samuel Beckett's Ireland*. New York: Black Cat Press in association with Riverrun Press.

———. 1992. "Foreward." In *Dream of Fair to middling Women*, by Samuel Beckett, edited by Eoin O'Brien and Edith Fournier, xi–xx. New York: Arcade Publishing in association with Riverrun Press.

O'Hara, J. D. 1981. "Jung and the Narratives of *Molloy*." *Journal of Beckett Studies*, no. 7:19–47.

——— 1992. "Freud and the Narrative of *Moran*." *Journal of Beckett Studies,* n.s., 2 (1): 47–63.

——— 1994. "Beckett Backs Down: From Home to Murphy via Valéry." *Journal of Beckett Studies,* n.s., 3 (2): 37–55.

Opie, Iona and Peter Opie, eds. 1952. *The Oxford Dictionary of Nursery Rhymes*. Corrected ed. Oxford: Clarendon Press. First published in 1951.

Ovid [Publius Ovidius Naso]. 1977. *Ovid in Six Volumes*. 3d ed. The Loeb Classical Library. Translated by Frank Justus Miller. Cambridge, Massachusetts: Harvard University Press. Composed in 0007.

Perkins, David, ed. 1967. *English Romantic Writers*. San Diego: Harcourt Brace Jovanovich.

Pilling, John, ed. 1994. *The Cambridge Companion to Beckett*. Cambridge: Cambridge University Press.

———. 1998. *Beckett Before Godot*. Cambridge: Cambridge University Press.

Posnock, Ross. 1981. "Beckett, Valéry and *Watt*." *Journal of Beckett Studies*, no. 6:51–62.

Preminger, Alex and T. V. F. Brogan, eds. 1993. *The New Princeton Encyclopedia of Poetry and Poetics*. Princeton: Princeton University Press.

Putnam, Samuel, Maida Castelhun Darton, George Reavey, and J. Bronowski, eds. 1931. *France, Spain, England and Ireland*. Volume 1 of *The European Caravan: An Anthology of the New Spirit in*

European Literature. Brewer, Warren and Putnam. Subsequent volumes were not published.

Rabinovitz, Rubin. 1984. *The Development of Samuel Beckett's Fiction*. Urbana: University of Illinois Press.

———. 1989. "Beckett and Psychology." *Journal of Beckett Studies*, no. 11–12:65–77.

———. 1992. *Innovation in Samuel Beckett's Fiction*. Urbana: University of Illinois Press.

———. 1995. "Samuel Beckett's Revised Aphorisms." *Contemporary Literature* 36 (2): 203–25.

Renaud, Madeleine. 1967. "Beckett the Magnificant." In *Beckett at 60: A Festschrift*, edited by John Calder, 81–83. London: Calder and Boyars.

Ricks, Christopher B. 1993. *Beckett's Dying Words*. The Clarendon Lectures, 1990. New York: Oxford University Press.

Rudin, Walter. 1976. *Principles of Mathematical Analysis*. 3d ed. International Series in Pure and Applied Mathematics. New York: McGraw-Hill. First published in 1953.

Sacks, Oliver W. 1985. *The Man Who Mistook His Wife for a Hat and Other Clinical Tales*. New York: Summit Books.

Scruton, Roger. 1983. "Beckett and the Cartesian Soul." Chapter 16 of *The Aesthetic Understanding: Essays in the Philosophy of Art and Culture*, 222–41. Manchester, England: Carcanet Press.

Segrè, Elisabeth Bregman. 1977. "Style and Structure in Beckett's *Ping*: *That Something Itself*." *Journal of Modern Literature* 6 (1): 127–47 (February).

Sharp, Cecil J. 1932. *English Folk Songs from the Southern Appalachians: Comprising 274 Songs and Ballads with 968 Tunes, Including 39 Tunes Contributed by Olive Dame Campbell*. 2d and enl. ed. Edited by Maud Karpeles. London: Oxford University Press. First published in 1917.

Shenker, Israel. 1956. "Moody Man of Letters: A Portrait of Samuel Beckett, Author of the Puzzling 'Waiting for Godot.'" *New York Times*, 6 May, sec. 2, pp. 1, 3.

Shimony, Abner. 1989. "The Conceptual Foundations of Quantum Mechanics." In *The New Physics*, 373–95. Cambridge: Cambridge University Press.

Simpson, J. A. and E. S. C. Weiner, eds. 1989. *The Oxford English Dictionary*. 2d ed. New York: Oxford University Press.

Smith, Anna. 1993. "Proceeding by Aporia: Perception and Poetic Language in Samuel Beckett's *Worstward Ho*." *Journal of Beckett Studies,* n.s., 3 (1): 21–37.

Smith, Sir William and Sir John Lockwood. 1976. *Chambers Murray Latin-English Dictionary*. Edinburgh and London (respectively): W and R Chambers and John Murray. Also known as *A Smaller Latin-English Dictionary* in a 1933 edition.

Stern, Richard. 1991. "Samuel Beckett." *Salmagundi*, no. 90–91 (spring–summer): 179–90.

Stevenson, Key Gilliland. 1986. "Belacqua in the Moon: Beckett's Revisions of 'Dante and the Lobster.'" In McCarthy 1986, 36–46.

Swift, Jonathan. 1969. *A Tale of a Tub*. Edited by Edward Hodnett. New York: AMS Press.

Topsfield, Valerie. 1988. *The Humour of Samuel Beckett*. New York: St. Martin's Press.

Unger, Peter. 1979. "I Do Not Exist." Chapter 10 of *Perception and Identity: Essays Presented to A. J. Ayer, With His Replies*, edited by Graham Macdonald, 235–51. Cornell University Press.

University of Chicago Press. 1993. *The Chicago Manual of Style*. 14th ed. Chicago: University of Chicago Press.

Vernon, William Warren. 1907. *Readings on the* Purgatorio *of Dante: Chiefly Based on the Commentary of Benvenuto da Imola*. 3d ed. rev., 2 vols. London: Methuen. First published in 1889.

Whitman, Walt. 1965. *Leaves of Grass*. Edited by Sculley Bradley and Harold W. Blodgett. New York University Press. Reprint, New York: W. W. Norton and Co., 1973. First published in 1891.

Winston, Mathew. 1977. "*Watt*'s First Footnote." *Journal of Modern Literature* 6 (1): 69–82 (February).

Wolfson, Harry Austryn. 1973. *Studies in the History of Philosophy and Religion*. 2 vols. Edited by Isadore Twersky and George H. Williams. Harvard University Press.

Wolosky, Shira. 1995. *Language Mysticism: The Negative Way of Language in Eliot, Beckett, and Celan*. Stanford: Stanford University Press.

Work, John Wesley. 1915. *Folk Song of the American Negro*. Nashville, Tennessee. Reprint, New York: Negro Universities Press, 1969.

Zeifman, Hersh. 1975. "Religious Imagery in the Plays of Samuel Beckett." In Cohn 1975, 85–94.

Zurbrugg, Nicholas. 1989. "Review of Samuel Beckett Special Issue of *Irish University Review: A Journal of Irish Studies* 14 (1), spring 1984." *Journal of Beckett Studies*, pp. 163–66.