Biblatex Compatibility (`hibib.dtx`)

Charles Duan

July 20, 2023

Because metadata for legal citations is more structured and complex than for other citation systems, *Hereinafter* does not use Biblatex files as its primary input format. However, the `hibib` package provides a compatibility layer that uses Biblatex to process `.bib` files into *Hereinafter* data structures. The package also enables rudimentary production of `.bib` files from *Hereinafter* reference definitions.

```
\ProvidesPackage{hibib}[2021/02/26 Hereinafter to BibLaTeX conversion]
\RequirePackage{strings}
\RequirePackage{etoolbox}
\RequirePackage[datamodel=hibl]{biblatex}
```

1 Initial Setup

There are two initial setup tasks that must be completed. First, the reference type names are not consistent between *Hereinafter* and Biblatex, so a mapping first needs to be established.

```
\def\hi@bib@map#1#2{%
    \@namedef{hi@bib@kind@#1}{#2}%
    \@namedef{hi@bib@hikind@#2}{#1}%
    \csletcs{def#2}{def#1}%
}
\hi@bib@map{jrnart}{article}
\hi@bib@map{website}{misc}
\hi@bib@map{website}{online}
\hi@bib@map{case}{jurisdiction}
\hi@bib@map{procart}{inproceedings}
```

Converts a *Hereinafter* reference type to a biblatex reference type.

```
\def\hi@bib@bibkind#1{%
    \@ifundefined{hi@bib@kind@#1}{#1}{\csname hi@bib@kind@#1\endcsname}%
}
```

Converts a biblatex reference type to a *Hereinafter* reference type.

```
\def\hi@bib@hiakind#1{%
    \@ifundefined{hi@bib@hikind@#1}{#1}{\csname hi@bib@hikind@#1\endcsname}%
}
```

Second, Biblatex only retains references that are actually cited in a document. Accordingly, we need to signal that every cited reference should be retained, by issuing a `\nocite` command.

Internally, we patch `\hi@pse@sigvolref`, which is called every time a reference is used in a citation. The hook below informs biblatex of the references used by adding a `\nocite` command, indicating to Biblatex that the reference should be included in the `.bcf` file.

```
\apptocmd\hi@pse@sigvolref{%
    \expandafter\ifx\csname fc@\@this@case\endcsname\fc@this \else
        \expandafter\nocite\expandafter{\@this@case}%
    \fi
}{}{}
```

2 Producing Biblatex Files

The `\hiBibFile{`⟨*file*⟩`}` macro initiates production of a Biblatex file. The command should be called before any *Hereinafter* references are defined.

Internally, we patch `\hi@param@read` (described in `refs.dtx`) to collect the reference's parameters and then output the reference definition to the bibliography file.

```
\def\hiBibFile#1{%
    \newwrite\tf@bib
    \immediate\openout \tf@bib #1.bib\relax
    \pretocmd\hi@param@read{\let\hi@bib@paramlist\@empty}{}{}
    \apptocmd\hi@param@read{\hi@bib@output}{}{}
}
\let\tf@bib\relax
```

If `\hiBibFile` has been executed, then the `\hi@bib@output` macro below is executed after the parameters for a reference have been read. The macro generates a Biblatex-style reference definition and writes it to the bibliography file defined by `\hiBibFile`.

```
\def\hi@bib@output{%
    \ifx\tf@bib\relax\else
        \let\reserved@b\@empty
        \@expand{\@unbrace{\@tfor\reserved@a:=}}\hi@bib@paramlist i\do{%
            \@ifundefined{hi@bib@p@\reserved@a}{}{%
                \protected@edef\reserved@b{%
                    \reserved@b
                    \space\space\reserved@a={%
                        \csname hi@bib@p@\reserved@a \endcsname
                    },^^J%
                }%
                \expandafter\let\csname hi@bib@p@\reserved@a\endcsname \relax
            }%
        }%
        \immediate\write\tf@bib{%
            @\expandafter\hi@bib@bibkind\expandafter{\hi@kv@kind}{%
                \@this@case,^^J%
                \reserved@b
            }^^J%
        }%
    \fi
}
```

3 Reference Parameters

`\hi@bib@paramlist` is a list of parameters to be included when printing a bibliography reference. This is set to `\@empty` inside `\hi@param@read` (see `\hiBibFile` above) and then augmented.

```
\let\hi@bib@paramlist\relax
```

We patch each `\KV@hi@`⟨*param*⟩ macro, to execute its normal operations and then call `\hi@bib@doparam`. The `\hi@params` macro lists every parameter with a `\do` command.

```
\def\do#1#2{%
    \def\reserved@a##1{%
        \@namedef{KV@hi@##1}####1{%
            % In here, |#1| is the param name, |##1| is the prior macro
            % definition of the param, and |####1| is the variable name for the
            % argument the param's new macro will take.
            ##1\hi@bib@doparam{#1}{####1}%
        }%
    }%
    \@expand\reserved@a{\csname KV@hi@#1\endcsname{##1}}{ii}%
}
\hi@params
```

Upon setting a parameter in a reference definition, this macro performs the work necessary to have the parameter saved to the Bibtex output. If a macro `\hi@bib@param@`⟨*param*⟩ is defined, then that macro is executed with the parameter value as the argument. Otherwise, the default `\hi@bib@saveparam` macro is run. #1 is the parameter name; #2 the given value.

```
\def\hi@bib@doparam#1#2{%
    %
    % Only run if we are saving parameters
```

```
    %
    \ifx\hi@bib@paramlist\relax \else
        \@ifundefined{hi@bib@param@#1}{%
            \ifblank{#2}{%
                %
                % For blank parameters, if there is a default value then use
                % that. Otherwise do not output to the .bib file (because biber
                % will ignore it anyway).
                %
                \@ifundefined{KV@hi@#1@default}{}{%
                    \@expand{\hi@bib@saveparam{#1}}{%
                        \csname kv@hi@#1@default\endcsname
                    }{ii}%
                }%
            }{%
                \hi@bib@saveparam{#1}{#2}%
            }%
        }{%
            \csname hi@bib@param@#1\endcsname{#2}%
        }%
    \fi
}
```

Save a given value for the parameter by appending it to `\hi@bib@paramlist`, and defining a new macro `\hi@bib@p@#1` with the parameter's value. #1 is the parameter; #2 the value.

```
\def\hi@bib@saveparam#1#2{%
    \def\reserved@a{#2}%
    \expandafter\edef\csname hi@bib@p@#1\endcsname{%
        \expandafter\strip@prefix\meaning\reserved@a
    }%
    \addto@macro\hi@bib@paramlist{{#1}}%
}
```

These are utility macros for changing how parameters are output to the Bibtex file.

Ignore a particular parameter.

```
\def\hi@bib@ignore#1{\@namedef{hi@bib@param@#1}##1{}}%
```

Rename a parameter.

```
\def\hi@bib@rename#1#2{%
    \@namedef{hi@bib@param@#1}##1{%
        \hi@bib@saveparam{#2}{##1}%
    }%
}
```

By default, the Bibtex parameter's value is the user's given value, regardless of *Hereinafter*'s processing. This macro directs the output to be the post-processed value.

```
\def\hi@bib@postval#1{%
    \@namedef{hi@bib@param@#1}##1{%
        \@expand{\hi@bib@saveparam{#1}}{\csname hi@kv@#1\endcsname}{ii}%
    }%
}
```

Like `\hi@bib@postval` but allows changing the output parameter name (#2).

```
\def\hi@bib@postval@rename#1#2{%
    \@namedef{hi@bib@param@#1}##1{%
        \@expand{\hi@bib@saveparam{#2}}{\csname hi@kv@#1\endcsname}{ii}%
    }%
}
```

For a parameter list #1, adds #2 to the list.

```
\def\hi@bib@list@add#1#2{%
    \@ifundefined{hi@bib@p@#1}{%
        \hi@bib@saveparam{#1}{#2}%
    }{%
        \expandafter\addto@macro\csname hi@bib@p@#1\endcsname{ and #2}%
    }%
}
```

For parameter list #1, adds name #2 to the list.

```
\def\hi@bib@name@add#1#2{%
    \hi@namesplit{#2}{\hi@bib@name@addsplit{#1}}%
}
```

3.1 Name Parameters

We have to mangle the name parts in order to be approximately compatible with Biblatex expectations. The translation is as follows:

- Given name ⟹ Biblatex given name
- Family name + suffix ⟹ Biblatex family name
- Institution ⟹ Biblatex suffix

```
\def\hi@bib@name@addsplit#1#2#3#4#5#6{%
    \def\reserved@a{given={#2}}%
    \ifblank{#3}{%
        % I don't have any way of doing a name suffix with no family name, so we
        % just pretend that the suffix is the family name
        \notblank{#4}{\appto\reserved@a{, family={#4}}}{}%
    }{%
        \ifblank{#4}{\appto\reserved@a{, family={#3}}}{%
            \appto\reserved@a{, family={#3 {#4}}}%
        }%
    }%
    \notblank{#5}{\appto\reserved@a{, "suffix={#5}"}}{}%
    \@expand{\hi@bib@list@add{#1}}\reserved@a i%
    \ifblank{#6}{}{\hi@bib@list@add{#1}{others}}%
}
```

3.2 Date Parameters

Deal with a date parameter. This sets up just the .bib output. #1 is the prefix to the date parameter name.

```
\def\hi@bib@date#1{%
    \hi@bib@ignore{#1date}% The date parameter is an alias
    \hi@bib@rename{#1year}{#1datetext}%
}
\def\hi@bib@month#1{%
    \ifcase#1\or
        Jan.\or Feb.\or Mar.\or Apr.\or May\or June\or
        July\or Aug.\or Sept.\or Oct.\or Nov.\or Dec.\or
        13\or14\or15\or16\or17\or18\or19\or20\or % Next is 21
        Spring\or Summer\or Fall\or Winter\or
        Spring\or Summer\or Fall\or Winter\or % Northern hemisphere
        Spring\or Summer\or Fall\or Winter\or % Southern hemisphere
        Spring\or Summer\or Fall\or Winter\or % Southern hemisphere
        Q1\or Q2\or Q3\or Q4\else ???%
    \fi
}
```

3.3 Specific Parameter Mappings

```
\def\hi@bib@param@agency#1{%
    \hi@bib@saveparam{agency}{#1}%
    \expandafter\let\csname hi@bib@p@court\endcsname\relax
}
\hi@bib@ignore{broadcaster}
\hi@bib@ignore{bill}
\hi@bib@ignore{cite}
\hi@bib@ignore{comment}
\hi@bib@ignore{docname}
\hi@bib@ignore{jcite}
\hi@bib@ignore{kind}
\hi@bib@ignore{issuer}
\hi@bib@ignore{journal}
\hi@bib@ignore{parse}
\hi@bib@ignore{parties}
\hi@bib@ignore{producer}
\hi@bib@ignore{publiclaw}
\hi@bib@ignore{publno}
\hi@bib@ignore{sameparties}
\hi@bib@ignore{serial}
\hi@bib@ignore{series}
\hi@bib@ignore{slipop}
\hi@bib@ignore{sponsor}
\hi@bib@ignore{src}
\hi@bib@ignore{state}
\hi@bib@ignore{volume}
\hi@bib@ignore{revparties}
```

```
\hi@bib@rename{hyphenation}{hyphenate}
\hi@bib@rename{name}{title}
\hi@bib@rename{page}{pages}
\hi@bib@rename{rep}{journaltitle}
\hi@bib@rename{inline}{shorttitle}
\hi@bib@date{}
\hi@bib@date{orig}
\hi@bib@date{issue}
\def\hi@bib@param@author#1{\hi@bib@name@add{author}{#1}}
\def\hi@bib@param@instauth#1{\hi@bib@list@add{author}{{#1}}}
\def\hi@bib@param@editor#1{\hi@bib@name@add{editor}{#1}}
\def\hi@bib@param@insted#1{\hi@bib@list@add{editor}{{#1}}}
\def\hi@bib@param@to#1{\hi@bib@name@add{to}{#1}}
\def\hi@bib@param@instto#1{\hi@bib@list@add{to}{{#1}}}
\def\hi@bib@param@paren#1{\hi@bib@list@add{paren}{{#1}}}
\def\hi@bib@param@country#1{\hi@bib@list@add{country}{{#1}}}
\hi@bib@postval{citation}
\hi@bib@postval{in}
\hi@bib@postval{reprinted}
\def\hi@bib@param@vol#1{%
    \find@in{:}{#1}{%
        \@tworun{\hi@bib@saveparam{volume}}{\hi@bib@saveparam{issue}}%
    }{\hi@bib@saveparam{volume}{#1}}%
}
```

4 Reading a Biblatex File

To use a Biblatex file for input, include it using the usual `\addbibresource` command. The `hibib` package patches Biblatex's input commands so that each time a reference is read, a corresponding *Hereinafter* reference is defined.

Note that for Biblatex to operate, the `.tex` file must be compiled first to produce a `.bcf` file, which is then processed through the `biber` program to produce a `.bbl` file that actually provides reference definitions. This package requires the same process. The document with citation commands must be compiled first, producing warnings about unknown references for every citation. Then `biber` may be called and the document recompiled.

Biblatex does not offer a way to collect all the parameters it reads from the `.bbl` file. So first we patch the necessary methods. These will construct a list `\hi@bib@bblparams` to contain every parameter in the bibliography entry.

```
\apptocmd\blx@bbl@entry{\let\hi@bib@bblparams\@empty}{}{}
\apptocmd\blx@bbl@fielddef{\listadd\hi@bib@bblparams{#1}}{}{}
\apptocmd\blx@bbl@namedef{\listadd\hi@bib@bblparams{#1}}{}{}
\apptocmd\blx@bbl@listdef{\listadd\hi@bib@bblparams{#1}}{}{}
```

Now install a hook performed at the time the .bbl file is read. `\AtDataInput` executes after a .bbl file entry is read, so the parameters are stored in Biblatex's internal memory.

At that point, `\hi@param@read` is redefined to iterate over all the parameters found in `\hi@bib@bblparams`, and process each one using `\hi@bib@setparams@do`. Next, the reference definer macro is constructed and executed. This will call `\hi@param@read`, thereby defining a *Hereinafter* reference using Biblatex's parameters.

```
\AtDataInput{%
    \begingroup
    \def\hi@param@read#1{%
        \forlistloop\hi@bib@setparams@do\hi@bib@bblparams
    }%
    \hi@bib@writeconv{}{%
        \expandafter\string\csname
            def\usefield\hi@bib@hiakind{entrytype}%
        \endcsname{%
            \thefield{entrykey}%
        }\@charlb
    }%
    \edef\reserved@a{%
        \expandafter\noexpand \csname def\csfield{entrytype}\endcsname
        {\thefield{entrykey}}{}%
    }\reserved@a
    \hi@bib@writeconv{}{\@charrb}%
```

```
    \endgroup
}
```

`\hi@bib@setparams@do` is called for each Biblatex parameter that was recorded, during the execution of `\hi@param@read` as a reference is being defined. It does one of the following depending on the parameter it finds:

- If a special processing macro `\hi@bib@bbl@`⟨*param*⟩ is defined, then that macro is executed with no arguments.
- If no default macro exists of the form `\KV@hi@`⟨*param*⟩, then nothing happens.
- If the field is undefined in the .bbl file, a warning is given.
- Otherwise, calls `\usefield` to get the Biblatex field value, and executes `\hi@bib@sethiparam` to transfer the value to *Hereinafter*.

```
\def\hi@bib@setparams@do#1{%
    \@ifundefined{hi@bib@bbl@#1}{%
        \@ifundefined{KV@hi@#1}{}{%
            \iffieldundef{#1}{%
                \PackageWarning\hi@pkgname{%
                    Field #1 in .bbl file not found%
                }%
            }{%
                \usefield\hi@bib@sethiparam{#1}{#1}%
            }%
        }%
    }{%
        \csname hi@bib@bbl@#1\endcsname
    }%
}
```

Sets a *Hereinafter* parameter to a given value. Note that this is backwards in terms of parameters: `#1` is the *value*, and `#2` is the parameter name. This macro also write the parameter to a file if requested.

```
\def\hi@bib@sethiparam#1#2{%
    \csname KV@hi@#2\endcsname{#1}%
    \hi@bib@writeconv{}{\space\space#2={#1},}%
}
```

Macros of the form `\hi@bib@bbl@`⟨*param*⟩ are special handlers for parameters from the `.bbl` file.

```
\def\hi@bib@bbl@paren{%
    \gdef\hi@bib@tmp{}%
    \indexlist[bbparenprint]{paren}%
    \hi@bib@tmp
}
\DeclareIndexListFormat{bbparenprint}{%
    \gappto\hi@bib@tmp{%
        \hi@bib@sethiparam{#1}{paren}%
    }%
}
```

4.1 Name Lists

These set up conversion of Biblatex's author lists to *Hereinafter* format.

```
\def\hi@bib@bbl@author{\hi@bib@bblnamelist{author}{author}{instauth}}
\def\hi@bib@bbl@editor{\hi@bib@bblnamelist{editor}{editor}{insted}}
\def\hi@bib@bbl@to{\hi@bib@bblnamelist{to}{to}{instto}}
\def\hi@bib@bblnamelist#1#2#3{%
    \def\hi@bib@bblhummac{#2}%
    \def\hi@bib@bblinstmac{#3}%
    \ifandothers{#1}{%
        \gdef\hi@bib@maybeetal{et al.}%
    }{%
        \gdef\hi@bib@maybeetal{}%
    }%
    \gdef\hi@bib@tmp{}%
    \indexnames[bbauthprint]{#1}%
    \hi@bib@tmp
}
\DeclareIndexNameFormat{bbauthprint}{%
    \ifdefvoid\namepartgiven{%
        % Institutional author.
        \edef\reserved@a{%
            \noexpand\hi@bib@sethiparam{%
```

```
                \expandonce\namepartfamily
            }{\hi@bib@bblinstmac}%
        }%
    }{%
        % Human author.
        \edef\reserved@a{%
            \noexpand\hi@bib@sethiparam{%
                parts:%
                {\expandonce\namepartgiven}%
                {\expandonce\namepartfamily}%
                {\expandonce\namepartsuffix}%
                {\expandonce\hi@bib@maybeetal}%
            }{\hi@bib@bblhummac}%
        }%
        \global\let\maybeetal\@empty
    }%
    \expandafter\gappto\expandafter\hi@bib@tmp\expandafter{%
        \reserved@a
    }%
}
```

Convert Biblatex's delimiters to simple spaces.

```
\def\bibnamedelima{ }
\def\bibnamedelimb{ }
\def\bibnamedelimc{ }
\def\bibnamedelimd{ }
\def\bibnamedelimi{ }
```

4.2 Dates

This sets up date parameters for conversion from Biblatex to *Hereinafter*.

```
\def\hi@bib@bbldate#1{%
    \@namedef{hi@bib@bbl@#1datetext}{%
        \usefield\hi@bib@sethiparam{#1datetext}{#1year}%
    }%
    \@namedef{hi@bib@bbl@#1year}{%
        \iffieldundef{#1datetext}{%
            \edef\reserved@a{%
                \noexpand\hi@bib@sethiparam\endcsname{%
                    \csname if#1datecirca\endcsname{c. }{}%
                    \iffieldundef{#1month}{}{%
                        \usefield\hi@bib@month{#1month}%
                        \iffieldundef{#1endmonth}{}{%
                            \iffieldsequal{#1month}{#1endmonth}{}{%
                                -\usefield\hi@bib@month{#1endmonth}%
                            }%
                        }%
                        \iffieldundef{#1day}{}{%
                            \space
                            \csfield{#1day}%
                            \iffieldundef{#1endday}{}{-\csfield{#1endday}}%
                            ,%
                        }%
                        \space
                    }%
                    \csfield{#1year}%
                    \iffieldundef{#1endyear}{}{%
                        \iffieldsequal{#1year}{#1endyear}{}{%
                            -\csfield{#1endyear}%
                        }%
                    }%
                }{#1year}%
            }\reserved@a
        }{}%
    }%
}%
\hi@bib@bbldate{}
\hi@bib@bbldate{orig}
\hi@bib@bbldate{issue}
```

4.2(a) Saving Biblatex Entries in *Hereinafter* Format The `\hiBibConvert{`⟨*file*⟩`}` macro constructs a *Hereinafter* format reference file based on any Biblatex entries read. This command should be called before any entries are read via `\addbibresource`.

```
\def\hiBibConvert#1{%
    \newwrite\tf@bibconv
    \immediate\openout\tf@bibconv #1.tex\relax
    \def\hi@bib@writeconv##1##2{%
        \begingroup
            ##1%
            \immediate\write\tf@bibconv{##2}%
        \endgroup
    }%
}
\let\hi@bib@writeconv\@gobbletwo
```

5 Citation Command Compatibility

Biblatex uses different commands for inserting citations. Fundamentally there cannot be straightforward compatibility between those commands and *Hereinafter*, because the underlying data model of citations is different. Nevertheless, some partial compatibility is provided for the `\autocite` and `\autocites` commands, which could help with automatic conversion of Pandoc markdown documents.

```
\DeclareRobustCommand\autocites{%
    \hi@pse@acc@reset
    \hi@pse@bib@read
}
\let\autocite\autocites
\def\hi@pse@bib@read{%
    \futurelet\@let@token\hi@pse@bib@read@
}
\def\hi@pse@bib@read@{%
    \let\hi@pse@bib@suffix\@empty
    \@testcase
    \ifx\@let@token[\fi{\hi@pse@bib@read@oneopt}%
    \ifx\@let@token\bgroup\fi{\hi@pse@bib@read@ref}%
    \default{\hi@pse@bib@run}%
}
\def\hi@pse@bib@read@oneopt[#1]{%
    \def\hi@pse@bib@suffix{#1}%
    \futurelet\@let@token\hi@pse@bib@read@nextopt
}
\def\hi@pse@bib@read@nextopt{%
    \@test\ifx\@let@token[\fi{\hi@pse@bib@read@twoopts}{\hi@pse@bib@read@ref}%
}
\def\hi@pse@bib@read@twoopts[#1]{%
    % Suffix is actually prefix
    \expandafter\hi@pse@bib@parseprefix\hi@pse@bib@suffix~\@stop
    \def\hi@pse@bib@suffix{#1}%
    \hi@pse@bib@read@ref
}
\def\hi@pse@bib@parseprefix#1~#2\@stop{%
    \hi@pse@acc@add\@gobble{#1}{}%
}
\def\hi@pse@bib@read@ref#1{%
    \hi@pse@acc@addnonblank\@gobble{ }{}%
    \hi@pse@acc@add\@gobble{#1}{}%
    \hi@pse@acc@savework\hi@pse@svr
    \ifx\hi@pse@bib@suffix\@empty
        \hi@pse@acc@savecite
    \else
        \expandafter\hi@pse@state@page\expandafter{\hi@pse@bib@suffix}%
    \fi
    \hi@pse@bib@read
}
\def\hi@pse@bib@run{%
    \unskip\footnote{%
        \@expand{%
            \hi@draw@citation{\@hi@captrue\@hi@senttrue}{\if@hi@dot\else.\fi}%
        }\hi@pse@acc@all i%
    }%
}
```