chemstyle — Writing chemistry with style*

Joseph Wright†

Released 2008/07/23

Abstract

The chemstyle package provides a "one-stop shop" for setting up formatting of LaTeX documents following the editorial policies of various chemical journals. It provides a number of handy chemistry-related commands, and loads several support packages to aid the chemist.

Contents

*This file describes version v1.3b, last revised 2008/07/23.

†E-mail: joseph.wright@morningstar2.co.uk

Table 1: Styles provided by chemstyle

Option	Journals using this style
none	Not applicable
angew	*Angew. Chem., Chem. Eur. J.*
jomc	*J. Organomet. Chem., Coord. Chem. Rev.*
ic	*Inorg. Chem.*
jacs	*J. Am. Chem. Soc.*
jcp	*J. Phys. Chem. A, J. Phys. Chem. B*
orglett	*Org. Lett.*
rsc	*Chem. Commun., Org. Biomol. Chem. Dalton Trans.*
tetlett	*Tetrahedron, Tetrahedron Lett.*

1 Introduction

The aim of chemstyle is to provide a quick method to set up various document parameters (such as caption formatting), simply by specifying the model journal. The package has also been designed to allow rapid addition of new journal styles. Each style definition is a separate file, and new styles can be added very readily. chemstyle has grown out of the rsc package, which had a similar aim but was much more limited (and less robustly implemented). The chemstyle package is also designed with the use of biblatex in mind: the rsc package is closely bound to traditional BibTeX use.

As a successor to the rsc package, chemstyle provides a range of chemistry-related additional macros. The set provided here is an extended version of those provide by rsc. Everything that can be done using the rsc LaTeX package is therefore possible using the chemstyle package.

The formatting system provided by chemstyle are intended for writing a variety of documents. Thus the stylistic changes made by the package do not seek to reproduce the appearance of printed journal articles. The package aims to be suitable for use in preparing drafts of papers, but also for writing reports, theses and so on.

2 Style options for chemstyle

`journal` chemstyle should be loaded with a package option specifying which journal style to follow. Currently, chemstyle is aware of a the styles listed in Table 1. New styles can be developed by creating a new file modelled on the existing definitions; chemstyle will automatically search for correctly-named styles. The style files provided with chemstyle have been derived from current practice in the target journals. It is not always easy to pick the correct stylistic settings from (sometimes inconsistent) real-world examples. The package author welcomes feedback on the styles provided.

The `none` style is notable as it is not based on a journal. Instead this is a minimal style, which provides the additional commands without making formatting changes. It is also the default style if no option is given. This

document has been compiled implicity using the `none` option, for example.

2.1 Creating styles

The process of creating a new styles for chemstyle is intended to be relatively easy. New styles should be saved as files with the extension `.jdf` (standing for for Journal Definition File), and should be saved somewhere in the path searched by TEX.[1] The definition files included in the package should provide a guide to the basic options available for producing new styles. Arbitrary TEX commands can be included, if they are necessary for a particular style. For example, other packages can be loaded in `\usepackage`.

The maintainer of chemstyle is happy to add new styles to the package, either by contribution by users or on request (when he has sufficient time!). If you have a new style to add (or corrections to an existing one), please contact the package author.

2.2 Package options

`journal` As of version 1.2, chemstyle uses the keyval system for option management. The choice of journal style can therefore be given as a normal option or by using the key `journal=`. The other package options provided by the package are described as they arise in the documentation.

`\cstsetup` Some of the package options can only be set when loading the package. The choice of journal style is one of these. Others can be altered anywhere in the document, using the `\cstsetup` macro. This accepts a keyval list and processes it as needed.

`chemcompounds` `floatrow` chemstyle loads the chemscheme package, and therefore will pass through the `chemcompounds` option. It can also request floats are created either by float or floatrow; this is controlled by the `floatrow` option. Setting `floatrow=false` means that float is used to generate floats; the default value is **`true`**.

3 Naming of the references section

chemstyle alters the naming of the references section of a document. By default, chemstyle alters the value of `\bibname` or `\refname` (as appropriate) to the form of words chosen by the target journal for the "References" section.

The "Notes and References" naming commands are language-aware, *via* the babel interface. Currently, chemstyle includes appropriate labels for babel languages `english`, `UKenglish`, `ngerman` and `french`. Other languages can be added if appropriate wordings are provided to the author. The naming system is designed to work correctly with both natbib and biblatex.

`notes` `nonotes` The package recognises the `notes` option for controlling how the references section is named. The accepts the values `true`, `false` and `auto`. The Boolean values either enable or disable the addition of "Notes and" to the "References" of the section title. The `auto` option works in conjunction with the notes2bib package. If notes are added, "Notes and" is included in the section title, whereas

[1] Depending on your TEX distribution, you may need to rebuild your file database after creating a new style. For MikTEX users, this can be done graphically or by typing `mpm -update-db` at the command line; for TeXLive, run `texhash`.

if no notes are given the section title remains as “References”. The `nonotes` option is equivalent to `notes=false`.

`notesbefore` The second option for this area is `notesbefore`. This takes `true` and `false` only, and sets whether “Notes and References” or “References and Notes” is produced.

4 Additional macros

4.1 Additional units

`\Hz` `\mmHg` `\molar` `\Molar` `\cmc` Both the Slunits and unitsdef packages provide methods for handling a range of units, but do not include a small number of unit macros useful to chemists. In contrast, the siunitx package provides all of these using the `synchem` module. If Slunits or unitsdef are loaded, chemstyle provides the missing units. On the other hand, if no unit package is loaded, siunitx is loaded and provides the units itself.

10 Hz
20 mmHg
30 mol dm^{-3}
40 M
50 cm^3

```
\SI{10}{\Hz}\\
\SI{20}{\mmHg}\\
\SI{30}{\molar}\\
\SI{40}{\Molar}\\
\SI{50}{\cmc}
```

4.2 The standard state symbol

`\standardstate` Related to the above, but not exactly a unit is the `\standardstate` command.[2] This generates the tricky `\standardstate` symbol. The symbol will adapt to local sizing.

the standard conditions are indicated: $\ominus$
Common but not correct: $\Delta G_\mathrm{f}^\ominus$ or $\Delta G_\mathrm{f}{}^\ominus$
Better: $\Delta_\mathrm{f} G^\ominus$
Sizing: $\int_{T^\ominus}^{T_\mathrm{out}}$

```
the standard conditions are indicated: \standardstate\\
Common but not correct:
  $\Delta G_\mathrm{f}^\standardstate$ or
  $\Delta G_\mathrm{f}{}^\standardstate$ \\
Better:
  $\Delta_\mathrm{f}G^\standardstate$\\
Sizing:
  $\int^{T_{\mathrm{out}}}_{T^\standardstate}$
```

4.3 Alkyl radicals

`\nPr` `\iPr` `\nBu` `\iBu` `\sBu` `\tBu` There are a few alkyl radicals that come up all of the time. No one seems to have put these into a package, so they are provided here. As you would expect, `\iPr` gives *i*-Pr, `\iBu` gives *i*-Bu and `\tBu` gives *t*-Bu, and so on. The style of the output depends on the journal style specified; most journals seem to favour one version of the abbreviation.

The alkyl group could be *n*-Pr, *i*-Pr
or *n*-Bu without affecting the selectivity.

```
The alkyl group could be \nPr, \iPr\\
or \nBu without affecting the selectivity.
```

`xspace` When chemstyle is loaded using the `xspace` option, the xspace package is

[2] The `\standardstate` macro is only defined if the user does not have their own version.

automatically used to add space after the command names, so that `\iPr group` will result in "*i*-Pr group" being typeset.

radhyphen
rademph
radsuper
radprefix

The appearance of these radical abbreviations is controlled by the package options `radhyphen`, `rademph`, `radsuper` and `radprefix`. The journal styles set these automatically, but they can be redefined at any point. The options work as might be expected, and are Boolean switches.

```
\cstsetup{radhyphen=false,radsuper=true}\\
\iPr \tBu
\cstsetup{radhyphen=true,radsuper=false,%
  radprefix=false,rademph=false}\\
\iBu \sBu
```

^{i}Pr ^{t}Bu
Bu-i Bu-s

4.4 Latin phrases

\latin
\etc
\eg
\ie
\etal
\invacuo
abbremph

The various Latin phrases commonly used in chemistry are made available as the obvious commands. By altering the definition of `\latin`, this allows ready switching from italic to Roman typesetting. Notice that `\etc`, `\ie` and `\eg` are aware of trailing periods, and so doubling-up should not occur. Once again, these macros use xspace when given as a package option to handle automatic addition of spaces after these phrases.

The use of italic for these abbreviations is set by altering the package option `abbremph`, which takes values `true` and `false`.

```
\etal or \invacuo plus \latin{some text}\\
\cstsetup{abbremph=false}
\ie equals \eg perhaps \etc
```

et al. or *in vacuo* plus *some text*
i.e. equals e.g. perhaps etc.

abbrcomma

For American journals, where it is obligatory to follow "*e.g.*" and "*i.e.*" with a comma, the package provides a mechanism for handling this automatically. Thus, when using an appropriate journal style, `\eg`, `\eg.` and `\eg,` will all result in typesetting "*e.g.*,". The Boolean package option `abbrcomma` controls this.

```
\eg this\\
\cstsetup{abbrcomma=true}
\ie that
```

e.g. this
i.e., that

The `\etc` and `\etal` commands are set up on the assumption that they come at the end of a sentence. Hence the spacing after these will default to an inter-sentence space. If you desire an inter-word space, use the normal methods

```
\etc\ more text \\
\etal~have shown
```

etc. more text
et al. have shown

nophrases

The definitions of all of the phrases are designed not to overwrite any given by the user *in the preamble*. So, if you have your own `\latin` macro, it will be used even if you load chemstyle. If you encounter any problems, try loading the package with the `nophrases` option; this option prevents the package even trying to define any of the phrase macros.

\latinemphon
\latinemphoff

For backward-compatibiltiy with previous versions of chemstyle, the macros `\latinemphon` and `\latinemphoff` are provided. These work as would be expected, to alter the formatting produced by `\latin`. The more general keyval method is now preferred for making this change.

5 Additional information

5.1 Advice for users of the rsc package

The chemstyle package is intended as a replacement for the rsc package. As such, it covers almost everything the the rsc LaTeX package does, and more. Users of rsc are strongly encouraged to update to using chemstyle. The bibliography styles provided by rsc will continue to be required, of course. Migration of these styles to biblatex is an on-going project.

5.2 Interactions with other packages

The chemstyle package has been designed to avoid, as far as possible, clashes with other packages. The package requires the presence of the standard graphicx and varioref packages. If these packages need to be loaded with explicit options, this should be done *before* loading chemstyle. The chemscheme package is also needed, as it provides the floating `scheme` environment essential in synthetic chemistry documents.

5.3 Captions above floats

The `scheme` float type is generated using either the float or floatrow package. This has the side-effect that the placement of captions for floats does not depended on where the `\caption` command comes inside the floating environment.[3] If you wish to alter the placement of captions, the mechanism of the underlying package will be needed. There are some subtle differences between the two: altough floatrow provides the float macros, they are not all 100% compatible. This document is compiled using floatrow, and so to fix the position of captions the following code is appropriate.

```
\begin{table}[ht]
  The float contents
  \caption{A caption below the float contents in the source}
\end{table}
\floatsetup[table]{style=plain} % When using floatrow
% \floatstyle{plain}
% \restylefloat{table} When using float
\begin{table}[ht]
  The second float contents, which should be above the caption
  \caption{A second caption below the float contents
    in the source}
\end{table}
```

Table 2: A caption below the float contents in the source

The float contents

[3] Normally this is a good thing.

The second float contents, which should be above the caption

Table 3: A second caption below the float contents in the source

6 A template for chemical articles

This is a very simple template for chemistry-related documents. Hopefully it contains a few extra hints for getting well-formatted documents quickly. For simplicity, the template assumes that the user is writing a thesis for a U.K. university. Hence it uses U.K. defaults and RSC-based styling. Most of the packages used have good documentation, but a brief summary of why they are recommend follows.

```
\documentclass[fontsize=10pt,paper=a4,english,UKenglish]
  {scrreprt}
\usepackage{geometry,upgreek,booktabs,babel}
\usepackage[journal=rsc,xspace=true]{chemstyle}
\usepackage[version=3]{mhchem}
\usepackage[footnotes]{notes2bib}
\usepackage[final]{microtype}
\usepackage[final,inactive]{pst-pdf}
\usepackage[colorlinks]{hyperref}
\begin{document}
  Document contents go here
\end{document}
```

The versatile KOMA-script bundle provides more advanced versions of the standard document classes. If you want paragraphs separated out, with no indents (a common style for theses), add the `parskip` option to the font and paper size ones given here.

The babel system is loaded to sort out hyphenation and so on, and could be useful if there are any foreign-language quotes.

geometry allows the users to alter page layout with ease: much better than trying to hack the raw LaTeX system. booktabs gives much nicer looking tables than the LaTeX 2ε default. upgreek provides non-italic lowercase Greek letters, which should be used for things such as bond descriptions.

π-bond σ*-orbital `$\uppi$-bond $\upsigma^*$-orbital`

Load chemstyle (of course) to give not only some easy formatting, but also to automatically provide a float type for schemes, thanks to the chemscheme package. This also loads either chemcompounds or (optionally) bpchem to track compound numbers.

The mhchem package provides the `\ce` command for rapidly typesetting formulas, so that you can type `\ce{H2SO4}` and get H_2SO_4.[4]

The microtype package improves formatting when used with the pdfTeX engine. By giving the `final` option, it is active even when using `draft` as a global option.

Using notes2bib allows the user to automatically add notes to the bibliography from within the document body. So you can put `\bibnote{A note}` in the

[4]There is a slight cheat here, as this document uses lower-case numerals in the text. The example is written as `$\ce{H2SO4}$`.

source, and this will move into the References section without any further effort. The `footnotes` option means that footnotes do the same.

Finally, the hyperref package makes headings, citations and so on into hyperlinks.

7 The package code

7.1 Setup code

\cst@id
\cst@ver

First of all, the package identifies itself and loads other packages needed to function. Loading chemscheme is deferred until later, to allow processing of options at the correct point.

```
\NeedsTeXFormat{LaTeX2e}
\def\cst@id$#1: #2.#3 #4 #5-#6-#7 #8 #9${%
    \def\cst@ver{#5/#6/#7\space v1.3b\space}}
\cst@id $Id: chemstyle.dtx 46 2008-07-23 11:29:06Z joseph $
\ProvidesPackage{chemstyle}
  [\cst@ver Writing chemistry with style]
```

Packages that are needed under all circumstances are loaded here. There are some limitations on the age of xkeyval.

```
\RequirePackage{xkeyval}[2005/05/07]
\RequirePackage{graphicx,varioref,caption,xspace}
```

\cst@tempa
\cst@tempb

A couple of temporary private macros.

```
\newcommand*{\cst@tempa}{}
\newcommand*{\cst@tempb}{}
```

7.2 Option handling

\cst@boolkey

Option handling is processed using xkeyval. As a number of Boolean keys are needed, a shortcut is handy.

```
\newcommand*{\cst@boolkey}[1]{
  \define@boolkey[cst]{opt}[cst@]{#1}[true]{}}
```

\ifcst@chemcompounds
\ifcst@floatrow
\ifcst@xspace
\ifcst@phrases
\ifcst@radprefix
\ifcst@rademph
\ifcst@radsuper
\ifcst@radhyphen
\ifcst@abbremph
\ifcst@abbrcomma
\ifcst@notesbefore

The Boolean keys are now defined.

```
\cst@boolkey{chemcompounds}
\cst@boolkey{floatrow}
\cst@boolkey{xspace}
\cst@boolkey{phrases}
\cst@boolkey{radprefix}
\cst@boolkey{rademph}
\cst@boolkey{radsuper}
\cst@boolkey{radhyphen}
\cst@boolkey{abbremph}
\cst@boolkey{abbrcomma}
\cst@boolkey{notesbefore}
```

\ifcst@notes
\ifcst@autonotes
\cst@tempa
\cst@tempb

For notes, some choices are available. These need a bit of testing.

```
\newif\ifcst@notes
\newif\ifcst@autonotes
```

```
\define@choicekey*[cst]{opt}{notes}[\cst@tempa]
  {true,false,auto}[true]
  {\cst@notesfalse
   \cst@autonotesfalse
   \edef\cst@tempb{auto}
   \ifx\cst@tempa\cst@tempb
     \cst@notestrue
     \cst@autonotestrue
   \fi
   \edef\cst@tempb{true}
   \ifx\cst@tempa\cst@tempb
     \cst@notestrue
   \fi}
```

Some old options are no longer needed at all.

```
\DeclareOptionX[cst]<opt>{siunits}
  {\PackageInfo{chemstyle}{Option siunits no longer required}}
\DeclareOptionX[cst]<opt>{SIunits}
  {\PackageInfo{chemstyle}{Option siunits no longer required}}
\DeclareOptionX[cst]<opt>{nonotes}
  {\ExecuteOptionsX[cst]<opt>{notes=false}}
\DeclareOptionX[cst]<opt>{nophrases}
  {\ExecuteOptionsX[cst]<opt>{phrases=false}}
```

`\cst@journal` `\cst@language`

The `journal` option can be given with or without `journal=`.

```
\define@cmdkeys[cst]{opt}[cst@]{journal,language}
\DeclareOptionX*{\edef\cst@journal{\CurrentOption}}
```

The default options are executed, and the user options are processed.

```
\setkeys[cst]{opt}{
  notes=true,
  phrases=true,
  chemcompounds=true,
  xspace=true,
  floatrow=true,
  journal=none,
  radprefix=true,
  rademph=true,
  radsuper=false,
  radhyphen=true,
  abbremph=true,
  abbrcomma=false,
  language=english,
  notesbefore=true}
\ProcessOptionsX[cst]<opt>
```

With the options processed, those to be passed to chemscheme are sorted out.

```
\ifcst@chemcompounds
  \PassOptionsToPackage{chemcompounds}{chemscheme}
\fi
\ifcst@floatrow
  \RequirePackage[floats=floatrow]{chemscheme}
\else
  \RequirePackage[floats=float]{chemscheme}
\fi
```

\cst@disablekey Some of the options are no longer relevant once the package is loaded. These are disabled here with a suitable message.

```
\newcommand*{\cst@disablekey}[1]{
  \define@key[cst]{opt}{#1}{\PackageWarning{chemstyle}
    {Option `#1' only valid when loading package}}}
\AtBeginDocument{
  \cst@disablekey{floatrow}
  \cst@disablekey{journal}
  \cst@disablekey{phrases}
  \cst@disablekey{nophrases}
  \cst@disablekey{notes}
  \cst@disablekey{nonotes}
  \cst@disablekey{chemcompounds}}
```

\cstsetup A macro for changing the settings at run-time without needing to know the internals of the package.

```
\newcommand*{\cstsetup}[1]{%
  \iffalse{\fi\ifnum0=`}\fi
  \setkeys[cst]{opt}{#1}%
  \ifnum0=`{\fi\iffalse}\fi}
```

7.3 Extra units

\cmc
\Hz
\cubiccentimeter
\Molar
\molar
\mmHg
\mol

The siunitx package provides all of the units desired here. So if it is loaded, the module `synchem` is all that needs to be loaded. Otherwise, a series of tests are needed for SIunits and unitsdef. If either of these are loaded, they are used; if not, then siunitx is loaded in any case.

```
\@ifpackageloaded{siunitx}
  {\requiresiconfigs{synchem}}
  {\@ifpackageloaded{SIunits}
     {\newcommand*{\cubiccentimeter}{\centi\metre\cubed}
      \newcommand*{\Molar}{\textsc{m}}
      \newcommand*{\molar}{\mole\usk\deci\metre\rpcubed}
      \newcommand*{\mmHg}{\milli\metre Hg}
      \newcommand*{\mol}{\mole}
      \newcommand*{\cmc}{\cubiccentimeter}
      \newcommand*{\Hz}{\hertz}}
     {\@ifpackageloaded{unitsdef}
      {\newunit{\cubiccentimeter}{%
         \cm\unitsuperscript{3}}
       \newunit{\Molar}{\textsc{m}}
       \newunit{\molar}{%
         \mole\unitsep\dm\unitsuperscript{--3}}
       \newunit{\mmHg}{mmHg}
       \newunit{\mol}{\mole}
       \newcommand*{\cmc}{\cubiccentimeter}
       \newcommand*{\Hz}{\hertz}}
      {\RequirePackage[alsoload=synchem]{siunitx}}}}
```

7.4 Standard state symbol

\standardstate
\cst@sstate
\cst@s@state

The "standard state" symbol is handy. This definition is adapted from one posted several times to `comp.text.tex` by Donald Arseneau.

```
\providecommand*{\standardstate}{%
  {\ensuremath{\protect\cst@sstate}}}
\newcommand*{\cst@sstate}{\mathpalette\cst@s@state\circ}
\newcommand*{\cst@s@state}[2]{%
  \ooalign{\hfil$#1-$\hfil\cr\hfil$#1#2$\hfil\cr}}
```

7.5 Alkyl radicals

\cst@radical

The `\cst@radical` macro does the hard work of declaring each abbreviation. The commands are made robust so they can (hopefully) be used anywhere.

```
\newcommand*{\cst@radical}[2]{%
  \expandafter\DeclareRobustCommand\expandafter*\expandafter{%
    \csname #1#2\endcsname}{%
```

\cst@emph
\cst@hyphen
\cst@super

To allow redefinition while running, the various component parts of the system are defined each time an abbreviation is used.

```
    \ifcst@radhyphen
      \def\cst@hyphen{-}%
    \else
      \let\cst@hyphen\relax
    \fi
    \ifcst@radsuper
      \let\cst@super\textsuperscript
    \else
      \let\cst@super\relax
    \fi
    \ifcst@rademph
      \let\cst@emph\emph
    \else
      \let\cst@emph\relax
    \fi
```

The final check is for a prefix, which is easiest to do here.

```
    \ifcst@radprefix
      \cst@super{\cst@emph{#1}}\cst@hyphen#2%
    \else
      #2\cst@hyphen\cst@super{\cst@emph{#1}}%
    \fi
    \cst@xspace}}
```

\cst@xspace

A re-useable `\xspace`-based macro.

```
\newcommand*{\cst@xspace}{%
  \ifcst@xspace
    \expandafter\xspace%
  \fi}
```

\nPr
\iPr
\nBu
\iBu
\sBu
\tBu

With a mechanism in place, the abbreviations are declared. The format of the resulting output will depend upon the configuration file used.

```
\cst@radical{n}{Pr}%
\cst@radical{i}{Pr}%
\cst@radical{n}{Bu}%
\cst@radical{i}{Bu}%
\cst@radical{s}{Bu}%
\cst@radical{t}{Bu}%
```

7.6 Float formatting

The next step is to format the floats correctly. The standard float types are now restyled to place the captions correctly (for most journals). Normally in chemical documents the author expects the float to be "here" if possible; this is therefore set as the default.

```
\floatstyle{plaintop}
\restylefloat{table}
\floatstyle{plain}
\restylefloat{figure}
\floatplacement{table}{htbp}
\floatplacement{figure}{htbp}
```

Work with `scheme`s needs to be done after hyperref might be loaded.

```
\AtBeginDocument{
  \restylefloat{scheme}
  \floatplacement{scheme}{htbp}}
```

The contents of floats are centred by default, using the hook from the chemscheme package.

```
\floatcontentscentre
```

7.7 Cross-references

The naming for cross-references is sorted out properly using the varioref package.

```
\labelformat{figure}{\figurename~#1}
\labelformat{table}{\tablename~#1}
\AtBeginDocument{\labelformat{scheme}{\schemename~#1}}
```

7.8 Latin phrases

`\latin`
`\cst@latin`
`\latinemphon`
`\latinemphoff`

A series of Latin phrases are provided, with a quick switch to print them in Roman letters if needed. A mechanism is needed to alter the effect of the `\latin` command *only* if the user does not have their own version. This is achieved here, with precautions taken to ensure the user can define their own `\latin` command *after* loading chemstyle and still have everything work properly.

```
\newcommand*{\cst@latin}{%
  \ifcst@abbremph
    \expandafter\emph%
  \fi}
\newcommand*{\latinemphon}{\cstsetup{abbremph=true}}
\newcommand*{\latinemphoff}{\cstsetup{abbremph=false}}
\AtBeginDocument{\providecommand*\latin{\cst@latin}}
```

\etc
\invacuo
\etal
\eg
\ie

For the macros themselves, care is taken about trailing full stops. The \xspace command deals with any problems of spacing. Things could go wrong with complex punctuation, as no other checks are performed. All of these functions use \providecommand to avoid standing on the user's own versions, if they exist.

```
\ifcst@phrases
  \AtBeginDocument{
    \providecommand*{\etc}
      {\@ifnextchar.{\cst@etc}{\cst@etc.\cst@xspace}}
    \providecommand*{\invacuo}
      {\latin{in vacuo}\cst@xspace}
    \providecommand*{\etal}
      {\@ifnextchar.{\cst@etal}{\cst@etal.\cst@xspace}}
    \providecommand*{\eg}
      {\ifcst@abbrcomma
         \expandafter\cst@commaabbr%
       \else
         \expandafter\cst@nocommaabbr%
       \fi
       {eg}}
    \providecommand*{\ie}
      {\ifcst@abbrcomma
         \expandafter\cst@commaabbr%
       \else
         \expandafter\cst@nocommaabbr%
       \fi
       {ie}}}
```

\cst@etal
\cst@etc
\cst@eg
\cst@ie

Internal macros are used for items ending in a full stop, to allow clean handling of spacing. Notice that \ie and \eg cannot come at the end of a sentence, they are designed to give only an inter-word space.

```
\newcommand*{\cst@etal}{\latin{et~al}}
\newcommand*{\cst@etc}{\latin{etc}}
\newcommand*{\cst@ie}{\latin{i.e\spacefactor999\relax}}
\newcommand*{\cst@eg}{\latin{e.g\spacefactor999\relax}}
```

\cst@commaabbr
\cst@nocommaabbr

To handle the comma issue, two general macros are used.

```
\newcommand*{\cst@commaabbr}[1]{%
  \csname cst@#1\endcsname%
  \cst@addpunct}
\newcommand*{\cst@nocommaabbr}[1]{%
  \@ifnextchar.%
    {\csname cst@#1\endcsname}
    {\csname cst@#1\endcsname.\cst@xspace}}
```

\cst@addpunct

The following macros are very closely based on those in the cite package used for moving citations after punctuation. The first macro is used as an initial hook. Notice that \relax is essential here, as it provides an argument for \cst@add@punct in the first round of checking.

```
\newcommand*{\cst@addpunct}{%
  \cst@add@punct\relax}
```

\cst@add@punct Here, a plain TEX \futurelet is used to test the next character. Notice that this macro takes a single argument, which is used to recursively gobble up punctuation.

```
\newcommand*{\cst@add@punct}[1]{%
  \futurelet\cst@tempa\cst@add@punct@}
```

\cst@add@punct@ \cst@tempa \cst@tempb The checking occurs here. If a match is made, then the process is repeated to allow the punctuation to be gobbled.

```
\newcommand*{\cst@add@punct@}{%
  \ifx.\cst@tempa
    \let\cst@tempb\cst@add@punct
  \else
    \ifx,\cst@tempa
      \let\cst@tempb\cst@add@punct
```

No match, and so new punctuation is to be added and the loop ended.

```
    \else
      \let\cst@tempb\cst@commapunct
    \fi
  \fi
  \cst@tempb}
```

\cst@commapunct The \cst@punct macro holds the comma-containing punctuation to be added.

```
\newcommand*{\cst@commapunct}{.,\cst@xspace}
```

7.9 Loading the style definition

The style definition is loaded here, once everything necessary is in place. A style must be loaded, so a default is provided to be on the safe side. The journal style file must have extension .jdf.

```
\InputIfFileExists{\cst@journal.jdf}
  {\PackageInfo{chemstyle}
    {Loaded \cst@journal.jdf}}
  {\PackageWarning{chemstyle}
    {Requested style `\cst@journal' does not exist\MessageBreak
     Loading default style}
    \InputIfFileExists{none.jdf}{}
      {\PackageError{chemstyle}{Failed to load none.jdf}
         {The default journal definition file was not
          found\MessageBreak This is a core part of
          chemstyle\MessageBreak Something is wrong with
          the installation}}}
```

7.10 Handling reference section naming

\cst@refsec To enable the renaming of \bibname or \refname to the desired value, some work is needed. Both natbib and biblatex alter the behaviour here subtly. The babel package also requires careful handling. The initial step is to fin whether the document class defines chapters, as this affects whether \refname or \bibname should be changed. This is done inside a group so that a false definition is not left around.[5] A second check is needed, in case the document class does not

[5] If \@undefined is ever fixed, this will no longer be needed. If ε-TEX was required, the \ifdefined would be a choice here; however, ε-TEX is not going to be required just for this.

define any reference section at all (for example, minimal does not).

```
\begingroup
  \@ifundefined{chapter}
    {\@ifundefined{refname}
       {\PackageInfo{chemstyle}
          {No bibliography name command found}
        \cstsetup{notes=false}}
       {\gdef\cst@refsec{\refname}}}
    {\gdef\cst@refsec{\bibname}}
\endgroup
```

\c@bibnote The business-end of the work is only carried out if the `notes` option is set to `true` or `auto`. The check above means that a reference section is also defined. First, some checking for the `auto` option occurs; the counter used by notes2bib needs to be defined even if the package is not loaded.

```
\ifcst@autonotes
  \AtBeginDocument{
    \@ifpackageloaded{notes2bib}{}
      {\newcounter{bibnote}}}
\fi
```

\cst@refname
\cst@ref@name
\cst@ref@name@ The internal macros to actually display the reference section name deals with the various options. The `\expandafter`s are for caution here: nothing *should* go wrong with the nested `\if` statements.

```
\newcommand*{\cst@refname}{%
  \ifcst@notes
    \ifcst@autonotes
      \ifnum\the\value{bibnote}>\z@
        \expandafter\expandafter\expandafter\expandafter
          \expandafter\expandafter\expandafter\cst@ref@name
      \else
        \expandafter\expandafter\expandafter\expandafter
          \expandafter\expandafter\expandafter\cst@ref@name@
      \fi
    \else
      \expandafter\expandafter\expandafter\cst@ref@name
    \fi
  \else
    \expandafter\cst@ref@name@%
  \fi}
\newcommand*{\cst@ref@name}[3]{%
  \ifcst@notesbefore
    #3\space#2\space#1%
  \else
    #1\space#2\space#3%
  \fi}
\newcommand*{\cst@ref@name@}[3]{#1}
```

\cst@setrefname Setting up the section name is a tricky affair. If no section is defined, then the whole process is skipped. Otherwise, the first check made s if the default language matches the definition given. If so, the heading is changed now.

```
\@ifundefined{cst@refsec}
  {\let\cst@setrefname\@gobblefour}
```

```
{\newcommand*{\cst@setrefname}[4]{
   \edef\cst@tempa{#1}%
   \ifx\cst@tempa\cst@language
     \expandafter\renewcommand\expandafter*\expandafter{%
       \cst@refsec}
       {\cst@refname{#2}{#3}{#4}}
   \fi
```

cst@tempa At the beginning of the document, a check needs to be made for babel and biblatex. For the later, a low-level hack is made to attach to the mechanism. For babel the appropriate commands are added to the `\captions...` macro.

```
   \AtBeginDocument{
     \@ifpackageloaded{biblatex}
       {\def\cst@tempa{\csgappto{bib@strings@#1}}
        \expandafter\cst@tempa\expandafter{\expandafter
          \renewcommand\expandafter*\expandafter{%
            \cst@refsec}{%
              \cst@refname{#2}{#3}{#4}}}}
       {\@ifpackageloaded{babel}
         {\def\cst@tempa{\expandafter\addto\expandafter{%
            \csname captions#1\endcsname}}
          \expandafter\cst@tempa\expandafter
            {\expandafter\renewcommand\expandafter*%
              \expandafter{\cst@refsec}{%
                \cst@refname{#2}{#3}{#4}}}
```

To make sure the changes are applied *now*, the current language is re-selected.

```
           \expandafter\selectlanguage\expandafter{%
             \languagename}}
         {}}}}}
```

Default names are loaded for a range of languages. Most journals stick to the same words, with only the order changing.

```
\cst@setrefname{english}{References}{and}{Notes}
\cst@setrefname{UKenglish}{References}{and}{Notes}
\cst@setrefname{ngerman}{Literatur}{und}{Notizen}
\cst@setrefname{german}{Literatur}{und}{Notizen}
\cst@setrefname{french}{R\'ef\'erences}{et}{Notes}
\cst@setrefname{frenchb}{R\'ef\'erences}{et}{Notes}
```

8 Configuration files

Each journal style needs slightly differing commands to get the formatting just right. This is handled here, with each style in a separate file. There is not a lot happening in most of these files, as the information is by its nature quite repetitive.

8.1 RSC style

```
\ProvidesFile{rsc.jdf}
  [\cst@ver RSC journal style]
\AtBeginDocument{%
```

```
  \renewcommand{\figurename}{Fig.}}
\captionsetup{labelsep=quad,labelfont=bf}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=false,
  radhyphen=true,
  abbremph=true,
  abbrcomma=false}
```

8.2 *Angew. Chem.* style

```
\ProvidesFile{angew.def}
  [\cst@ver Angew. Chem. journal style]
\captionsetup{labelsep=period,labelfont={bf,it},font=sf,
  singlelinecheck=off}
\captionsetup[table]{labelsep=colon}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=false,
  radhyphen=false,
  abbremph=false,
  abbrcomma=false}
```

8.3 *J. Organomet. Chem.* style

```
\ProvidesFile{jomc.def}
  [\cst@ver J. Organomet. Chem. journal style]
\captionsetup{labelsep=period}
\captionsetup[table]{labelsep=newline,singlelinecheck=off}
\AtBeginDocument{%
  \renewcommand{\figurename}{Fig.}}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=true,
  radhyphen=false,
  abbremph=false,
  abbrcomma=false,
  notesbefore=true}
```

8.4 *Tetrahedron Lett.* style

```
\ProvidesFile{tetlett.def}
  [\cst@ver Tetrahedron Lett. journal style]
\captionsetup{labelsep=period,singlelinecheck=off,labelfont=bf}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=false,
  radhyphen=true,
  abbremph=false,
  abbrcomma=true,
  notesbefore=true}
```

8.5 *J. Am. Chem. Soc.* style

```
\ProvidesFile{jacs.jdf}
  [\cst@ver J. Am. Chem. Soc. journal style]
\DeclareCaptionLabelSeparator{perquad}{.\quad}
\captionsetup{labelfont={bf,it,sf},textfont=sf,
  labelsep=perquad}
\captionsetup[figure]{textfont=rm}
\captionsetup{singlelinecheck=off}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=true,
  radhyphen=false,
  abbremph=false,
  abbrcomma=true,
  notesbefore=true}
\floatstyle{plaintop}
\restylefloat{scheme}
\floatstyle{plain}
\cst@notesbeforefalse
```

8.6 *Inorg. Chem.* style

Almost exactly the same as for *J. Am. Chem. Soc.*, so most of the work is left to `jacs.jdf`.

```
\ProvidesFile{ic.jdf}
  [\cst@ver Inorg. Chem. journal style]
\input {jacs.jdf}
\captionsetup{textfont=rm}
```

8.7 *J. Phys. Chem.* style

```
\ProvidesFile{jpc.jdf}
  [\cst@ver J. Phys. Chem. journal style]
\DeclareCaptionFormat{labelcaps}{\MakeUppercase{#1}#2#3}
\captionsetup{font=bf,labelsep=colon,format=labelcaps}
\captionsetup[figure]{format=plain,textfont=md,labelsep=period}
\captionsetup{singlelinecheck=off}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
  radsuper=false,
  radhyphen=false,
  abbremph=false,
  abbrcomma=true,
  notesbefore=true}
\floatstyle{plaintop}
\restylefloat{scheme}
\floatstyle{plain}
\cst@notesbeforefalse
```

8.8 *Org. Lett.* style

```
\ProvidesFile{orglett.jdf}
```

```
  [\cst@ver Org. Lett. journal style]
\RequirePackage{xcolor}
```

Currently, this style only works with float.

```
\@ifpackageloaded{floatrow}
  {\PackageError{chemstyle}
    {Org. Lett. style only works with float}
    {Load chemstyle with the floatrow=false option
      to\MessageBreak use the orglett style}}
  {}
```

`\OrgLettColour` To allow the user to control the colour of the "bars" in this style, an additional macro is provided. The purple colour looks about right for matching the printed journal, but if anyone has a better suggestion please let the package author know.

```
\newcommand*{\OrgLettColour}{purple}
```

`\fs@orglett` In order to get the distinctive coloured bars used by *Org. Lett.*, a new style for floats is needed. This is based on the `ruled` style from the float package.

```
\newcommand*{\fs@orglett}{%
  \def\@fs@cfont{\bfseries}%
  \let\@fs@capt\floatc@ruled
  \def\@fs@pre{\begingroup\color{\OrgLettColour}
    \hrule height12pt depth0pt \kern2pt\endgroup}%
```

This is the same `\@fs@mid` as is used in the float package for `plaintop` floats.

```
  \def\@fs@mid{\vspace\belowcaptionskip\relax}%
  \def\@fs@post{\begingroup\color{\OrgLettColour}\kern2pt
    \hrule height1.5pt depth0pt\endgroup}%
  \let\@fs@iftopcapt\iftrue}
```

`\fs@orglettfig` Figures need to be slightly different, so an almost identical command is needed.

```
\newcommand*{\fs@orglettfig}{%
  \def\@fs@cfont{\bfseries}%
  \let\@fs@capt\floatc@plain
  \def\@fs@pre{\begingroup\color{\OrgLettColour}%
    \hrule height12pt depth0pt \kern2pt\endgroup}%
  \def\@fs@mid{\vspace\abovecaptionskip\relax}%
  \def\@fs@post{\begingroup\color{\OrgLettColour}\kern2pt
    \hrule height1.5pt depth0pt\endgroup}%
  \let\@fs@iftopcapt\iffalse}
```

The new style is now applied. Users can change back to normal floats by changing back to the `plain` style.

```
\floatstyle{orglettfig}
\restylefloat{figure}
\floatstyle{orglett}
\restylefloat{scheme}
\restylefloat{table}
```

The more usual style commands now occur.

```
\DeclareCaptionLabelSeparator{perquad}{.\quad}
\captionsetup{labelfont=bf,labelsep=perquad}
\setkeys[cst]{opt}{
  radprefix=true,
  rademph=true,
```

```
438   radsuper=true,
439   radhyphen=false,
440   abbremph=false,
441   abbrcomma=true,
442   notesbefore=true}
443 \cst@notesbeforefalse
```

8.9 The empty style — none

To allow the user to load the extra macros provided here without any style changes, a "do nothing" style is provided. It simply makes sure that very little changes compared to the LaTeX kernel. This requires undoing the defaults provided above. For commands where a default is needed (*e.g.* the `\latin` command) the style of the RSC is followed. As the float package has been loaded, notice that captions will be placed below floats even if the `\caption` command appears above the contents of the floating environment.

```
444 \ProvidesFile{none.jdf}
445   [\cst@ver Empty journal style]
446 \floatstyle{plain}
447 \restylefloat{table}
448 \labelformat{figure}{#1}
449 \labelformat{table}{#1}
450 \floatplacement{table}{tbp}
451 \floatplacement{figure}{tbp}
452 \AtBeginDocument{
453   \floatplacement{scheme}{tbp}
454   \labelformat{scheme}{#1}}
```

9 Change History

10 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.