

chemstyle — Writing chemistry with style ^{*}

Joseph Wright [†]

Released 2007/09/18

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

		7.1 Setup code	5
		7.2 Alkyl radicals	6
		7.3 Option handling	6
		7.4 Extra units and related commands	8
		7.5 Float formatting	8
		7.6 Cross-references	9
		7.7 Latin words	9
		7.8 Handling section naming	11
		7.9 Loading the style definition	13
1 Introduction	1	8 Configuration files	13
		8.1 RSC style	13
2 Style options for chemstyle	2	8.2 <i>Angew. Chem.</i> style	13
2.1 Package options	2	8.3 <i>J. Am. Chem. Soc.</i> style	13
3 Naming of the references section	3	8.4 <i>Inorg. Chem.</i> style	14
		8.5 <i>J. Phys. Chem.</i> style	14
4 Additional macros	3	8.6 <i>Org. Lett.</i> style	14
4.1 Additional units	3	8.7 The empty style — none	15
4.2 The standard state symbol	3		
4.3 Alkyl radicals	3		
4.4 Latin phrases	4		
5 A template for chemical articles	4		
6 Advice for users of rsc and achemso	5		
7 The package code	5	9 Change History	16
		10 Index	16

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.

^{*}This file describes version v1.1a, last revised 2007/09/18.

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

Option	Journals using this style
none	Not applicable
angew	<i>Angew. Chem., Chem. Eur. J.</i>
ic	<i>Inorg. Chem.</i>
jacs	<i>J. Am. Chem. Soc.</i>
jcp	<i>J. Phys. Chem. A, J. Phys. Chem. B</i>
orglett	<i>Org. Lett.</i>
rsc	<i>Chem. Commun., Org. Biomol. Chem.</i> <i>Dalton Trans.</i>

Table 1: Styles provided by `chemstyle`

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`

`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 implicitly using the `none` option, for example.

2.1 Package options

As well as the various journal styles provided with the package, a number of options are recognised to alter compatibly with other packages and to give the desired output. `chemstyle` will also pass options through to the `chemscheme`

package, which is used to provide support for schemes. Users should consult the `chemscheme` documentation for the options applicable.

Currently, `chemstyle` has options `nonotes`, `nophrases`, `siunits` and `xspace`. These are explained at the appropriate places within this document. All are hopefully relatively clearly named.

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 “Notes and References” section.

The “References” naming commands are language-aware, *via* the `babel` interface. Currently, `chemstyle` includes appropriate labels for `babel` languages `english`, `UKenglish`, `ngerman` and `french`.¹ Other languages can be added if appropriate wordings are provided to the author.

The `chemstyle` packages recognises the `nonotes` option, which affects how the References section is headed. With the `nonotes` option, the “Notes” part of the section name is omitted, and only the “References” phrase is output.

4 Additional macros

4.1 Additional units

`\Hz`
`\mmHg`
`\molar`
`\Molar`
`\cmc`

Both the `unitsdef` and `siunits` packages provide a great number of easy to use unit commands, and handles spacing between numbers and units very well. However, a few useful units seem to be missing, and are provided here. By default, `chemstyle` uses the `unitsdef` package to achieve this, but using the `siunits` option will use that package instead. The most obvious of these is `\Hz`, which simply gives Hz. The command `\mmHg` has a non-breaking thin space, leading to mmHg. Two related commands are given for concentration: `\molar` gives mol dm⁻³, whilst `\Molar` gives M. Finally, the command `\cmc` is provided for generating cm³. When using `unitsdef`, all of the new macros space correctly with numbers, so inputting `10\cmc` gives 10 cm³, with a non-breaking space. Users of `siunits` will be familiar with the approach used by that package.

4.2 The standard state symbol

`\standardstate`

Related to the above, but not exactly a unit is the `\standardstate` command. This generates the tricky \ominus symbol: `\Delta S\standardstate` generates ΔS^\ominus . Note that this is safe in either text or maths mode.

4.3 Alkyl radicals

`\iPr`
`\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`

¹The `french` option for `babel` causes a clash with `unitsdef` concerning the command `\fg`. `babel` uses this for *guillemets*, while `unitsdef` uses it as an abbreviation for `\femtogram`. The `chemstyle` package prevents the clash, and leaves the `babel` definition intact.

gives `iPr`, `\iBu` gives `iBu` and `\tBu` gives `tBu`, and so on. The style of the output depends on the journal style specified; most journals seem to favour one version of the abbreviation. When `chemstyle` is loaded using the `xspace` option, the `xspace` package is automatically used to add space after the command names, so that `\iPr` group will result in “`iPr` group” being typeset.²

4.4 Latin phrases

`\latin` 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. The `\etal` again, these macros use `xspace`, when given as a package option, to handle automatic addition of spaces after these phrases. 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 `\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`, `\etc~more text`, *etc.* If this is a big issue, the package author may look at it again.

5 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.

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.

```
1 \documentclass[fontsize=10pt,paper=a4]{scrreprt}
```

The `geometry` package allows the users to alter page layout with ease; much better than trying to hack the raw LaTeX system.

```
2 \usepackage{geometry}
```

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

```
3 \usepackage[english,UKenglish]{babel}
```

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 `bpchem` or (optionally) `chemcompounds` to track compound numbers.

```
4 \usepackage[rsc,xspace]{chemstyle}
```

²Reader of the source of this document will see that this document has been compiled *without* the `xspace` option!

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

```
5 \usepackage[version=3]{mhchem}
```

Greek letters should be italic if used for variables, but upright (Roman) if used otherwise. So the standard LaTeX `\delta` is fine for typesetting an NMR shift (e.g. $\delta = \dots$ ppm.), but not for a σ -bond or a π -complex. The `upgreek` package provides commands such as `\upsigma` and `\uppi` which are ideal for this.

```
6 \usepackage{upgreek}
```

The default LaTeX table formatting is not very good. The `booktabs` package does things properly, and has good advice in the manual. A highly-recommended package for those interested in formatting (*i.e.* all (La)TeX users!).

```
7 \usepackage{booktabs}
```

The `microtype` package improves formatting when used with the pdfTeX engine.

```
8 \usepackage[final]{microtype}
```

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

```
9 \usepackage[footnotes]{notes2bib}
```

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

```
10 \usepackage[colorlinks]{hyperref}
```

The bulk of the document can then start.

```
11 \begin{document}
12 \end{document}
```

6 Advice for users of `rsc` and `achemso`

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

7 The package code

7.1 Setup code

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.

```
13 \NeedsTeXFormat{LaTeX2e}
14 \ProvidesPackage{chemstyle}
15 [2007/09/18 v1.1a Writing chemistry with style]
```

Packages that are needed under all circumstances are loaded here.

```
16 \RequirePackage{graphicx,varioref}
```

7.2 Alkyl radicals

`\cst@emph` Alkyl radical abbreviations are produced using the macro `\cst@radical`. This
`\cst@hyph` uses a series of formatting commands, which have default values provided here.
`\cst@super` The names are hopefully pretty self-explanatory.

```
17\let\cst@emph\relax
18\def\cst@hyph{}
19\let\cst@super\relax
20\newif \ifcst@prefix \cst@prefixtrue

\cst@radical The \cst@radical macro does the hard work of declaring each abbreviation.
To ensure there is no unexpected clash of names, \newcommand is used first to
check for any problems. The commands are made robust so they can (hopefully)
be used anywhere.

21\def\cst@radical#1#2{%
22 \expandafter\newcommand\expandafter{\csname #1#2\endcsname}{}%
23 \expandafter\DeclareRobustCommand\expandafter{%
24 \csname #1#2\endcsname}{}%
25 \ifcst@prefix
26 \cst@super{\cst@emph{#1}}\cst@hyph#2%
27 \else
28 #2\cst@hyph\cst@super{\cst@emph{#1}}%
29 \fi%
30 \cst@xspace
31 }
32 }
```

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

```
33\cst@radical{n}{Pr}%
34\cst@radical{i}{Pr}%
35\cst@radical{n}{Bu}%
36\cst@radical{i}{Bu}%
37\cst@radical{s}{Bu}%
38\cst@radical{t}{Bu}%
```

7.3 Option handling

Firstly, the “pass through” options are handled, as these will be sent through to chemscheme. The package options for chemstyle are also handled here.

```
39\newif \ifcst@notes \cst@notesttrue
40\newif \ifcst@phrases \cst@phrasesttrue
41\newif \ifcst@si \cst@sifalse
42\newif \ifcst@xspace \cst@xspacefalse
43\DeclareOption{ch}{\ExecuteOptions{chapter}}
44\DeclareOption{chapter}%
45 {\PassOptionsToPackage{chapter}{chemscheme}}
46\DeclareOption{chemcompounds}%
47 {\PassOptionsToPackage{chemcompounds}{chemscheme}}
48\DeclareOption{nonotes}{\cst@notesfalse}
49\DeclareOption{nophrases}{\cst@phrasesfalse}
50\DeclareOption{siunits}{\cst@sitrue}
51\DeclareOption{xspace}{\cst@xspacetrue}
```

In order to handle the loading of style definitions correctly, a bit of care is needed. The package options need to be loaded *before* `chemscheme`, but the style definitions need to be loaded after `chemscheme`. This is achieved by saving the options and looking at them again once the package loading is done. Notice that only one style definition can be used, so this is taken care of with a warning if needed.

```
52 \let\cst@option\relax
53 \DeclareOption*{%
54   \ifx\cst@option\relax\else
55     \PackageWarning{chemstyle}
56       {Extra option \cst@option\space ignored}
57   \fi
58   \edef\cst@option{\CurrentOption}
59 }
```

If `siunits` is already loaded, then the option has to be set.

```
60 \@ifpackageloaded{siunits}
61   {\ExecuteOptions{siunits}}
62   {}
63 \ProcessOptions*
64 \ifcst@si
65   \@ifpackageloaded{unitsdef}
66     {\PackageWarning{chemstyle}{%
67       You have set the siunits option,\MessageBreak
68       but have already loaded unitsdef!\MessageBreak
69       unitsdef will be used for extra unit macros}
70     \cst@sifalse}
71     {\RequirePackage{siunits}}
72 \else
```

The `\fg` macro is defined in French as a *guillemet* sign, and so to avoid a clash a bit of patching is needed. A check to see if `babel` has already defined `\fg`. If it has, the definition is saved and deleted.

```
73   \ifx\fg\@undefined
74   \else
75     \ifx\fg\relax
76     \else
77       \let\cst@fg\fg
78       \let\fg\@undefined
79     \fi
80   \fi
81   \RequirePackage{unitsdef}
```

If `\cst@fg` exists, it is restored here, otherwise `\fg` is undefined. At the beginning of the document, `\fg` is defined as an abbreviation for a femtogram if it is not being used by `babel`. This way there is not a problem if `babel` is loaded after `chemstyle`.

```
82   \ifx\cst@fg\@undefined
83     \let\fg\@undefined
84   \else
85     \let\fg\cst@fg
86   \fi
87 \AtBeginDocument{%
88   \ifx\fg\@undefined
89     \newcommand{\fg}{\femtogram}
```

```

90     \fi
91   }
92 \fi
93 \RequirePackage{chemscheme}

```

7.4 Extra units and related commands

`\cubiccentimeter` A few additional unit types are provided, which `unitsdef` and `siunits` do not provide. Most of these require different set up for the two supported packages.

```

\cmc
\Hz
94 \ifcst@si
95   \newcommand{\cubiccentimeter}{\centi\metre\cubed}
\molar
96   \newcommand{\Molar}{\textsc{m}}
97   \newcommand{\molar}{\mole\usl\deci\metre\rpcubed}
\mmHg
98   \newcommand{\mmHg}{\milli\metre~Hg}
\mol
99   \newcommand{\mol}{\mole}
100 \else
101   \newunit{\cubiccentimeter}{\cm\unitsuperscript{3}}
102   \newunit{\Molar}{\textsc{m}}
103   \newunit{\molar}{\mole\unitsep\dm\unitsuperscript{--3}}
104   \newunit{\mmHg}{\mm\unitsep{}Hg}
105   \newunit{\mol}{\mole}
106 \fi
107 \newcommand{\cmc}{\cubiccentimeter}
108 \newcommand{\Hz}{\hertz}

```

`\cst@varnothing` In a very similar vein, the “standard state” symbol is handy. This is produced by rotating the “varnothing” symbol from the AMS set. Note that the rotation angle here has been carefully checked, but is set by eye. The symbol is loaded directly here, rather than using the `amssymb` package, to avoid any clashes.

```

\standardstate
109 \DeclareSymbolFont{CSTAMS}{U}{msb}{m}{n}
110 \DeclareMathSymbol{\cst@varnothing}{\mathord}{CSTAMS}{"3F}
111 \newcommand*{\standardstate}%
112   {\textsuperscript{\rotatebox[origin=c]{140.8}%
113     {\ensuremath{\cst@varnothing}}}}

```

7.5 Float formatting

The next step is to format the floats correctly. Unfortunately, `memoir` does not provide all of the commands needed to achieve this. Thus the `float` package is needed; in order to load it, the `\newfloat` command in `memoir` is killed off. The `chemscheme` package does not load `float` when `memoir` is being used, because creating a new float type is catered for directly by `memoir`. Unfortunately, the `memoir` system isn’t flexible enough for what is needed by `chemstyle`, so the hard work of `chemscheme` is undone here! For other document classes, `float` will already have been loaded by `chemscheme`.

```

114 \@ifclassloaded{memoir}
115   {\let\newfloat\@undefined
116     \RequirePackage{float}}
117   {}

```


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.

```
118 \floatstyle{plaintop}
119 \restylefloat{table}
120 \floatstyle{plain}
121 \restylefloat{scheme}
122 \restylefloat{figure}
123 \floatplacement{table}{htbp}
124 \floatplacement{scheme}{htbp}
125 \floatplacement{figure}{htbp}
```

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

```
126 \floatcontentscentre
```

7.6 Cross-references

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

```
127 \labelformat{scheme}{\schemename~#1}
128 \labelformat{figure}{\figurename~#1}
129 \labelformat{table}{\tablename~#1}
```

7.7 Latin words

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.

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

```
130 \let\latinemphon\relax
131 \let\latinemphoff\relax
132 \let\cst@latin\relax
133 \ifcst@phrases
134   \newcommand{\latinemphon}{\let\cst@latin\emph}
135   \newcommand{\latinemphoff}{\let\cst@latin\relax}
136   \AtBeginDocument{%
137     \providecommand\latin{\cst@latin}
138   }
139 \fi
140 \latinemphon
```

In ACS journals, the abbreviations “*e.g.*” and “*i.e.*” are always followed by a comma. In order to provide an automated way to do this, a series of macros are needed to test the punctuation trailing the macro names. Firstly, a switch is provided for the journal style files.

```
141 \newif \ifcst@comma \cst@commafalse
```

The implementation of the testing code is delayed until the beginning of the document to allow the switching to occur. First the case where a comma is needed is handled.

```

142 \AtBeginDocument{%
143   \ifcst@comma

```

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

```

144   \def\cst@punct{.,\cst@xspace}

```

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

```

145   \def\cst@addpunct{%
146     \cst@add@punct\relax%
147   }

```

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.

```

148   \def\cst@add@punct#1{%
149     \futurelet\@tempa\cst@add@punct@%
150   }

```

The checking occurs here. If a match is made, then the process is repeated to allow the punctuation to be gobbled.

```

151   \def\cst@add@punct@{%
152     \ifx\@tempa.%
153       \let\@tempb\cst@add@punct%
154     \else
155       \ifx\@tempa,%
156         \let\@tempb\cst@add@punct%

```

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

```

157       \else
158         \let\@tempb\cst@punct%
159       \fi
160     \fi
161     \@tempb%
162   }

```

The case where a comma is not needed is much more simple. No fancy work is done; instead a single check is made on the next character.

```

163   \else
164     \def\cst@addpunct{\@ifnextchar.{\spacefactor999.\cst@xspace}}
165   \fi
166 }

```

`\cst@xspace` The use of `xspace` is optional; this is handled using a package option and the internal macro `\cst@xspace`.

```

167 \ifcst@xspace
168   \RequirePackage{xspace}
169   \let\cst@xspace\xspace
170 \else
171   \let\cst@xspace\relax
172 \fi

```

<code>\etc</code> <code>\invacuo</code> <code>\etal</code> <code>\eg</code> <code>\ie</code>	<p>For the macros themselves, care is taken about trailing full stops. The <code>\xspace</code> 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 <code>\providecommand</code> to avoid standing on the user's own versions, if they exist.</p> <pre> 173 \ifcst@phrases 174 \AtBeginDocument{ 175 \providecommand{\etc}% 176 {\latin{etc}\@ifnextchar.{}{.\cst@xspace}} 177 \providecommand{\invacuo}% 178 {\latin{in vacuo}\cst@xspace} 179 \providecommand{\etal}% 180 {\latin{et~al}\@ifnextchar.{}{.\cst@xspace}} 181 \providecommand{\eg}% 182 {\latin{e.g}\cst@addpunct} 183 \providecommand{\ie}% 184 {\latin{i.e}\cst@addpunct} 185 } 186 \fi </pre>
--	--

7.8 Handling section naming

First, a new if is needed to differentiate between “Notes and References” and “References and Notes.”

```
187 \newif \ifcst@notesbefore \cst@notesbeforetrue
```

<code>\cst@name@refs</code>	<p>To keep life simple in the main macro, and to save on redundant code, the ordering of “Notes,” “and” and “References” is handled here.</p>
-----------------------------	---

```

188 \def\cst@name@refs#1#2#3{%
189   \ifcst@notesbefore
190     #3\space#2\space#1%
191   \else
192     #1\space#2\space#3%
193   \fi
194 }

```

<code>\cst@language</code>	<p>The default language of the document is probably English. However, this is alterable if needed, as it is not set in stone. Anyone needing to mess with this is probably happy using <code>\makeatletter</code>, so the command is kept out of user space.</p>
----------------------------	--

```
195 \def\cst@language{english}
```

<code>\cst@refsection</code>	<p>The existence of a suitable reference section name command is checked here. Some document classes (e.g. <code>minimal</code>) may not define a suitable command. Assuming that is not the case, <code>\cst@refsection</code> is used to store the name of the macro holding the references section.</p>
------------------------------	--

```

196 \ifx\refname\@undefined
197   \ifx\bibname\@undefined
198     \PackageWarning{chemstyle}%
199       {No bibliography name command defined in document class}
200     \def\cst@namerefs#1#2#3#4{%
201       \else
202       \def\cst@refsection{\bibname}

```

```

203     \fi
204 \else
205     \def\cst@refsection{\refname}
206 \fi

```

`\cst@namerefs` The main macro of the heading-altering section takes four arguments, `{\language}`, `{\references}`, `{\and}` and `{\notes}`. The first is the babel label for the language that the words are four, and the other three parameters and the words “References”, “and” and “Notes”, respectively.

```

207 \ifx\cst@namerefs\@undefined
208     \def\cst@namerefs#1#2#3#4{%
209         \ifcst@notes%

```

The non-babel version of the function only does anything if the language passed in #1 is the default document language (probably English). The check for this needs the macro `\cst@language@check` to get `\ifx` to behave as desired.

```

210         \def\cst@language@check#1%
211         \ifx\cst@language\cst@language@check
212             \expandafter\renewcommand\expandafter{\cst@refsection}%
213             {\cst@name@refs{#2}{#3}{#4}}%
214         \fi

```

To ensure that things work whether babel is loaded or not, a few hoops have to be jumped through. Altering the babel strings is delayed until the start of the document, in case babel is loaded after chemstyle. There is also a complication that MikTeX defines `\language` even if babel is not loaded. This may contain an undesired value, and so the presence of babel needs to be tested for to avoid strange errors.

```

215     \AtBeginDocument{%
216         \@ifpackageloaded{babel}{%
217             \expandafter\addto\expandafter{\csname captions#1\endcsname}%
218             {\expandafter\renewcommand\expandafter{\cst@refsection}%
219             {\cst@name@refs{#2}{#3}{#4}}}%
220             \expandafter\selectlanguage\expandafter{\language}%
221         }%
222     \else%
223         \ifx#1\cst@language
224             \expandafter\renewcommand\expandafter{\cst@refsection}{#2}%
225         \fi
226     \AtBeginDocument{%
227         \@ifpackageloaded{babel}{%
228             \expandafter\addto\expandafter{\csname captions#1\endcsname}%
229             {\expandafter\renewcommand\expandafter{\cst@refsection}{#2}}%
230             \expandafter\selectlanguage\expandafter{\language}%
231         }%
232     \fi%
233 }
234 \fi

```

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

```

235 \cst@namerefs{english}{References}{and}{Notes}
236 \cst@namerefs{UKenglish}{References}{and}{Notes}
237 \cst@namerefs{ngerman}{Literatur}{und}{Notizen}

```

```
238 \cst@namerefs{french}{R\ref\erences}{et}{Notes}
```

7.9 Loading the style definition

The style definition is loaded here, once else everything 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.

```
239 \ifx\cst@option\relax
240   \PackageWarning{chemstyle}
241     {You didn't say which style to use \MessageBreak
242       Defaulting to the ``no change'' style: none}
243   \def\cst@option{none}
244 \fi
245 \InputIfFileExists{\cst@option.jdf}
246   {\PackageInfo{chemstyle}
247     {Loaded \cst@option.jdf}}
248   {\PackageWarning{chemstyle}
249     {Required style \cst@option\space does not exist}}
```

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

```
250 \ProvidesFile{rsc.jdf}[2007/09/18 v1.1a]
251 \AtBeginDocument{%
252   \renewcommand{\figurename}{Fig.}}
253 \RequirePackage[labelsep=quad,labelfont=bf]{caption}
254 \let\cst@emph\emph
255 \def\cst@hyph{-}
256 \let\cst@super\relax
257 \cst@prefixtrue
258 \latinemphon
```

8.2 Angew. Chem. style

```
259 \ProvidesFile{cst-angew.def}[2007/09/18 v1.1a]
260 \RequirePackage[labelsep=period,labelfont={bf,it},font=sf]{caption}
261 \captionsetup[table]{labelsep=colon}
262 \captionsetup{singlelinecheck=off}
263 \let\cst@emph\emph
264 \def\cst@hyph{}
265 \let\cst@super\relax
266 \cst@prefixtrue
267 \latinemphoff
```

8.3 J. Am. Chem. Soc. style

```
268 \ProvidesFile{jacs.jdf}[2007/09/18 v1.1a]
```

```

269 \RequirePackage[labelfont={bf,it,sf},textfont=sf]{caption}
270 \DeclareCaptionLabelSeparator{perquad}{.\quad}
271 \captionsetup{labelsep=perquad}
272 \captionsetup[figure]{textfont=rm}
273 \captionsetup{singlelinecheck=off}
274 \let\cst@emph\emph
275 \def\cst@hyph{}
276 \let\cst@super\textsuperscript
277 \cst@prefixtrue
278 \floatstyle{plaintop}
279 \restylefloat{scheme}
280 \floatstyle{plain}
281 \latinemphoff
282 \cst@notesbeforefalse
283 \cst@commatrue

```

8.4 *Inorg. Chem.* style

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

```

284 \ProvidesFile{ic.jdf}[2007/09/18 v1.1a]
285 \input {jacs.jdf}
286 \captionsetup{textfont=rm}

```

8.5 *J. Phys. Chem.* style

```

287 \ProvidesFile{jpc.jdf}[2007/09/18 v1.1a]
288 \RequirePackage[font=bf,labelsep=colon]{caption}
289 \DeclareCaptionFormat{labelcaps}{\MakeUppercase{#1}#2#3}
290 \captionsetup{format=labelcaps}
291 \captionsetup[figure]{format=plain,textfont=md,labelsep=period}
292 \captionsetup{singlelinecheck=off}
293 \let\cst@emph\emph
294 \def\cst@hyph{}
295 \let\cst@super\textsuperscript
296 \cst@prefixtrue
297 \floatstyle{plaintop}
298 \restylefloat{scheme}
299 \floatstyle{plain}
300 \latinemphoff
301 \cst@notesbeforefalse
302 \cst@commatrue

```

8.6 *Org. Lett.* style

```

303 \ProvidesFile{orglett.jdf}[2007/09/18 v1.1a]
304 \RequirePackage{xcolor}

```

`\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.

```

305 \newcommand*{\OrgLettColour}{purple}

```

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.

```

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

```

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

```

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

```

Figures need to be slightly different, so an almost identical command is needed.

```

314 \newcommand\fs@orglettfig{\def\@fs@cfont{\bfseries}
315 \let\@fs@capt\floatc@plain
316 \def\@fs@pre{\begingroup\color{\OrgLettColour}
317 \hrule height12pt depth0pt \kern2pt\endgroup}%
318 \def\@fs@mid{\vspace\abovecaptionskip\relax}%
319 \def\@fs@post{\begingroup\color{\OrgLettColour}\kern2pt
320 \hrule height1.5pt depth0pt\endgroup}%
321 \let\@fs@iftopcapt\iffalse}

```

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

```

322 \floatstyle{orglettfig}
323 \restylefloat{figure}
324 \floatstyle{orglett}
325 \restylefloat{scheme}
326 \restylefloat{table}

```

The more usual style commands now occur.

```

327 \RequirePackage[labelfont=bf]{caption}
328 \DeclareCaptionLabelSeparator{perquad}{.\quad}
329 \captionsetup{labelsep=perquad}
330 \let\cst@emph\emph
331 \def\cst@hyph{}
332 \let\cst@super\textsuperscript
333 \cst@prefixtrue
334 \latinemphoff
335 \cst@notesbeforefalse
336 \cst@commatrue

```

8.7 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 nothing 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.

```

337 \ProvidesFile{none.jdf}[2007/09/18 v1.1a]
338 \RequirePackage{caption}
339 \floatstyle{plain}
340 \restylefloat{table}
341 \labelformat{scheme}{#1}
342 \labelformat{figure}{#1}
343 \labelformat{table}{#1}

```

```

344 \floatplacement{table}{tbp}
345 \floatplacement{scheme}{tbp}
346 \floatplacement{figure}{tbp}

```

9 Change History

v1.0		Fixed (another) error in alkyl formatting	13
General: Initial release of package . .	1	Fixed error with spacing after <i>e.g.</i> and <i>i.e.</i>	1
v1.0a		License changed from GPL to LPPL	1
General: Added <i>J. Phys. Chem.</i> style	14	Phrases modified to better avoid clash with user's own commands	1
Added <i>Org. Lett.</i> style	14	\cst@add@punct: New macro . .	10
Fixed incorrect Latin formatting for <i>angew</i> option	13	\cst@add@punct@: New macro . .	10
No longer load <i>fixltx2e</i> package	5	\cst@add@punct: New macro . .	10
v1.0b		\cst@latin: New macro	9
General: Fixed formatting of alkyls	13	\cst@varnothing: New macro . .	8
Sorted a problem with <i>babel</i> and figure name format	13	\eg: Adds comma for ACS journals	11
\mol: New macro	8	\ie: Adds comma for ACS journals	11
v1.1			
General: Added <i>phrases</i> option . .	6	v1.1a	
Added <i>siunits</i> option	6	General: Fixed error in documentation compilation under LaTeX . .	1
Added <i>xspace</i> option	6		

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.

Symbols	264, 275, 294, 331	243, 245, 247, 249
\'	238	\cst@language
	 <u>195</u> , 211, 223
C		\cst@language@check
\cmc	<u>3</u> , <u>94</u> 20, 257, 266, 277, 296, 333
\cst@add@punct . .	<u>145</u>	\cst@latin
\cst@add@punct@ .	<u>145</u> <u>130</u>
\cst@add@punct <u>145</u> , 182, 184	\cst@name@refs
	 <u>188</u> , 213, 219
\cst@commafalse .	141	\cst@namerefs
\cst@commatrue 283, 302, 336	.. 200, <u>207</u> , 235–238
\cst@emph		\cst@notesbeforefalse
	. <u>17</u> , 26, 28, 254, 263, 274, 293, 330 282, 301, 335
\cst@fg	77, 82, 85	\cst@notesbeforetrue
\cst@hyph <u>17</u> , 26, 28, 255, 187
		\cst@notesfalse
	 48
		\cst@notetrue
	 39
		\cst@option
		54, 56, 58, 239,
		\cst@phrasesfalse 49
		\cst@phrasetrue . 40
		\cst@prefixtrue
	 20, 257, 266, 277, 296, 333
		\cst@punct
	 <u>144</u> , 158
		\cst@radical <u>21</u> , 33–38
		\cst@refsection
	 <u>196</u> , 212, 218, 224, 229
		\cst@sifalse
	 41, 70
		\cst@sittrue
	 50
		\cst@super
	 <u>17</u> , 26, 28, 256, 265, 276, 295, 332
		\cst@varnothing
	 <u>109</u>
		\cst@xspace

... 30, 144, 164, <u>167</u> , 176, 178, 180	\ie <u>4</u> , <u>173</u>	\mol <u>94</u>
\cst@xspacefalse . 42	\ifcst@comma . 141, 143	\Molar <u>3</u> , <u>94</u>
\cst@xspacetrue .. 51	\ifcst@notes .. 39, 209	\molar <u>3</u> , <u>94</u>
\cubiccentimeter . <u>94</u>	\ifcst@notesbefore 187, 189	N
E	\ifcst@phrases 40, 133, 173	\nBu <u>3</u>
\eg <u>4</u> , <u>173</u>	\ifcst@prefix .. 20, 25	O
\etal <u>4</u> , <u>173</u>	\ifcst@si ... 41, 64, 94	\OrgLettColour <u>305</u> ,
\etc <u>4</u> , <u>173</u>	\ifcst@xspace . 42, 167	308, 311, 316, 319
F	\invacuo <u>4</u> , <u>173</u>	R
\fs@orglett 306	\iPr <u>3</u> , <u>3</u>	\rpcubed 97
\fs@orglettfig .. 314	L	S
\futurelet 149	\latin .. <u>4</u> , <u>130</u> , 176, 178, 180, 182, 184	\sBu <u>3</u>
H	\latinemphoff <u>130</u> ,	\schemename 127
\Hz <u>3</u> , <u>94</u>	267, 281, 300, 334	\standardstate . <u>3</u> , <u>109</u>
I	\latinemphon . <u>130</u> , 258	T
\iBu <u>3</u>	M	\tBu <u>3</u>
	\mmHg <u>3</u> , <u>94</u>	