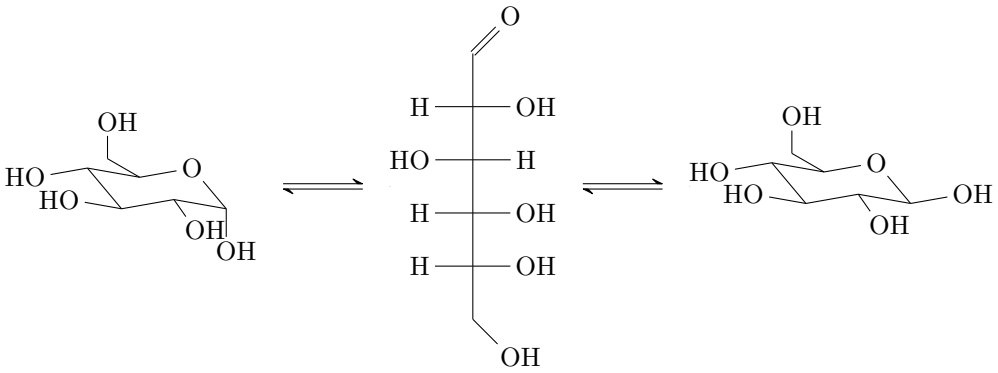
chemnum v0.3

2011/08/03

Clemens Niederberger

http://www.mychemistry.eu/
contact@mychemistry.eu

chemnum is a new and hopefully comprehensive approach for the numbering of chemical compounds.



(**1α**) (**1c**) (**1β**)

Scheme 1: In a solution the α- and the β-anomer of D-glucose (**1α** and **1β**) are in equilibrium via the open-chain form (**1c**).

Contents

1 Licence, Requirements

chemnum v0.3 underlies the LaTeX Project Public License Version 1.3 or later. (http://www.latex-project.org/lppl.txt)

chemnum internally loads the packages expl3 and xparse[1]. xparse is part of the l3packages[2] bundle, expl3 is part of the l3kernel[3] bundle. This means, chemnum needs LaTeX3 Support.

chemnum also needs the packages etoolbox[4], psfrag[5] and (sometimes) textgreek[6], see section 3.

2 New in chemnum v0.3

Since v0.3 the following is new:

- the key `sub-cmpd-sep` now is `sub-marker`, see page 21
- the key `cmpd-sub-counter` now is `sub-counter`, see page 16

The renaming follows the wish, to distinguish different classes of keys indicating the scope they're used for, see page 15.

- several keys to create labels like **1a-c,e**, see section 9.10 on page 21.
- the command `\cmpdref`, which replaces strings in `eps` files, see section 8.3.
- the key `ref-tag`, see page 16.

3 Package Options

chemnum has one package option.

```
\usepackage[textgreek = <value>]{chemnum}
```

With this you can both choose between the three different styles of the textgreek package[7] or load chemnum without the textgreek package. Available values are `artemisia`, `cbgreek`, `euler` and `false`. If you use chemnum without option, then textgreek is loaded with the

[1] http://www.ctan.org/pkg/xparse
[2] http://www.ctan.org/pkg/l3packages
[3] http://www.ctan.org/pkg/l3kernel
[4] http://www.ctan.org/pkg/etoolbox
[5] http://www.ctan.org/pkg/psfrag
[6] http://www.ctan.org/pkg/textgreek
[7] See the textgreek documentation for details.

cbgreek style. Using \usepackage[textgreek]{chemnum} is the same as \usepackage[textgreek = cbgreek]{chemnum} and \usepackage{chemnum}.

If you don't have the textgreek package installed and don't need to use greek labels, you can use \usepackage[textgreek = false]{chemnum}. Then choosing the counter options greek or Greek will the result in using alph or Alph, respectively (see section 9.5).

4 Motivation

As far as I know there are three packages that are meant to help with numbering chemical compounds. All of them have their weaknesses.

The first one – chemcono[8] – redefined bibliography commands for that purpose. Compounds have to be declared in what is called \theffbibliography. Then one can reference them with \fcite. However, it produces a list of compounds in the text. So the package author suggests:

> *After compilation and printout, discard the last page.*
>
> *Stefan Schulz*

Obviously that's not a perfect solution.

The second one – chemcompounds[9] – was written, because the author didn't want to work with the weaknesses of chemcono any more. When he wrote the package he basically used the same mechanism to create the labels as chemcono did.

> *When taking a closer look at the chemcono package, I realised that the only thing one has to do is to get rid of everything which produces text. Thus, as a basis I used the mechanism of \bibitem and \cite in pretty much the same way as chemcono does by extracting the corresponding code from article.cls and latex.ltx but deleting any unnecessary commands producing output. I also introduced several lines of code to make the printing of the compound names more customisable.*
>
> *Stephan Schenk*

Some points still left me unsatisfied, though:

1. Compounds usually need to be declared with \declarecompound. They need to be declared in any case if you need a label like **1a**. Then, one even needs to choose the label by hand, what somehow undermines the automatic numbering principle.
2. The layout can't be changed for a single label but only for all.
3. The numbers can't be reset. *Although in most cases this is neither necessary nor can it be recommended*, there can be individual cases where this would be useful.
4. A list of several compounds \compound{a,b,c} can only be customized with more effort than what would be convenient.

Then there is bpchem[10], which provides commands similar to \label \ref: \CNlabel{}, \CNlabelnoref{} and \CNref{}. It provides commands for sublabels, too: \CNlabelsub{}{}, \CNlabelsubnoref{}{} and \CNrefsub{}{}. This makes it more flexible than the

[8] http://www.ctan.org/pkg/chemcono
[9] http://www.ctan.org/pkg/chemcompounds
[10] http://www.ctan.org/pkg/bpchem

others regarding sublabels. However, it barely provides possibilities to customize the labels, lists are not possible and the fact that there are different commands for labels and sublabels isn't the best solution, either.

chemnum is intended to fill these gaps. For this all commands have been written from scratch. Some of the ideas of chemcompounds e. g. regarding delimiters and layout have been picked up, though.

If you notice any feature missing, please let me know by sending me an email.

5 Numbering compounds

5.1 Basic command

The main command of this package is

```
\cmpd{<label name>}
```

When `<label name>` is used the first time, the label is created, saved (= declared) and printed. Each further use just prints the label.

```
1  Compounds \cmpd{a} and \cmpd{b} are declared and can be used any time: \
       cmpd{a}. No pre-declaring is necessary. Compounds like \cmpd{c} are
       numbered in the order they appear in the text.\par
2  Once again: \cmpd{b}, \cmpd{a}, \cmpd{c}.
```

Compounds **1** and **2** are declared and can be used any time: **1**. No pre-declaring is necessary. Compounds like **3** are numbered in the order they appear in the text.
Once again: **2**, **1**, **3**.

If it is necessary to declare a compound without printing the label, this is possible with

```
\cmpd*{<label name>}
```

This declares the label but doesn't print anything. **This command is not needed when `\cmpdinit{}` is used, see section 6 (pages 7ff).**

```
1  The hidden version\cmpd*{d} declares the label but doesn't print anything.
       The next \cmpd{e} continues to count with the next number. With \cmpd{d}
        the label can be used, of course.
```

The hidden version declares the label but doesn't print anything. The next **5** continues to count with the next number. With **4** the label can be used, of course.

You can pretty much use what you like for a label name[11]. You shouldn't leave blanks in the name, though. This may not cause an error at all, but *could* declare different compounds with the same name.

[11] There are restrictions, see section 5.2 on page 6 and section 9.9 on page 21.

```
\cmpd{aa  }, \cmpd{a a }, \cmpd{a  a}, \cmpd{ a a}, \cmpd{  aa} and \cmpd{
    aa } all have the same label. As well as \cmpd{aa }, \cmpd{a a}, \cmpd{
    aa} and \cmpd{aa}.
```

6, **6**, **6**, **6**, **6** and **6** all have the same label. As well as **6**, **6**, **6** and **6**.

5.2 Sublabels

If you want a label like **1a**, you need to use the following syntax:

```
\cmpd{<label name>.<subname>}
```

`<label name>` is the main name which stays the same, `<subname>` varies. This syntax means that the point `.` *cannot* be a part of `<label name>` or `<subname>`.

```
\cmpd{f.one} and \cmpd{f.two} are related, as are \cmpd{g.one} and \cmpd{g.
    two}. Of course, these labels can be used again: \cmpd{g.two} and \cmpd{
    f.one}.
```

7a and **7b** are related, as are **8a** and **8b**. Of course, these labels can be used again: **8b** and **7a**.

This also works, if the main name has already been used.

```
\cmpd{a} and its variants \cmpd{a.one} and \cmpd{a.two}
```

1 and its variants **1a** and **1b**

The same way the main name of combined labels can be used solely.

```
\cmpd{f} and \cmpd{g}
```

7 and **8**

How you can create a label like **7a,b** is explained in section 9.10.

5.3 Lists

There is actually more to the `\cmpd` command. It also prints lists of labels.
The right description would be something like:

```
\cmpd{<(possibly comma separated list of)label name(s)>}
```

```
1  More than one label can be put inside \verb=\cmpd=, separated by commas.
       Then a list like \cmpd{a, b, c, e, g.two} is printed.
```

More than one label can be put inside \cmpd, separated by commas. Then a list like **1**, **2**, **3**, **5**, and **8b** is printed.

The – in the eyes of non-US Americans odd looking – **, and** between **5** and **8b** suggests that there are options to customize the list, see section 9.8 (page 19).

6 Initialize labels at once

The commands described in this section are not essential for the usage of chemnum, but provide additional functionality that some users might find useful.

6.1 Usage

6.1.1 Main labels

Perhaps you're missing chemcompounds' command \declarecompound already. For one thing one has a summary of all used label names. And one may be warned if one uses an uninitialized label name[12]. This behaviour can be realized with these commands:

```
\cmpdinit{<comma separated list of label names>}
\cmpdinit*{<comma separated list of label names>}
```

All compounds set inside \cmpdinit{} are initialized and declared *in the given order*. If one uses a label name that's not in the list, chemnum produces a warning message.

```
1  \cmpdinit{A, B, C} \cmpd{B} \cmpd{A.a} \cmpd{C} \cmpd{D}
```

2 1a 3 4

```
**********************************************
* chemnum warning:  "cmpd-init"
*
* You used \cmpdinit but didn't initiate compound "D" on line 1.
**********************************************
```

Please note that for labels with sublabels only the *main name* can and should be initialized. This initializes all compounds with the same main name.

\cmpdinit*{} also produces a warning but doesn't declare the labels. They are declared when you first use them in the order of appearance in the text.

[12] for instance due to a typing error

```
1  \cmpdinit*{X, Y, Z} \cmpd{Y} \cmpd{X.x} \cmpd{Z} \cmpd{W}
```

1 2a 3 4

```
************************************************
* chemnum warning:  "cmpd-init"
*
* You used \cmpdinit but didn't initiate compound "W" on line 1.
************************************************
```

If you need to use own labels (see page 16) but still want to initialize all labels at once, you should use `\cmpdinit*{}`, because then no labels are declared when they're initialized.

This command can be used several times, with or without `*`. A useful way would be a single usage, though, proposedly in the preamble of your document after the chemnum setup (see section 9).

6.1.2 Sublabels

If you'd like a warning for label-sublabel combinations as well, or want to preset their order, you can use

```
\cmpdinit[sub-init = true]{<comma separated list of label names>}
```

This will give you a warning in these cases, too.

```
1  \cmpdinit[sub-init = true]{E.e} \cmpd{E.e, E.f}
```

1a and **1b**

```
************************************************
* chemnum warning:  "cmpd-sub-init"
*
* You used \cmpdinit and "sub-init = true" but didn't initiate sub-compound
* "E.f" on line 1.
************************************************
```

6.2 Error instead of warning

If you want chemnum to produce an error instead of a warning when you use an unititialized label name, you can use `\cmpdinit` like this:

```
\cmpdinit[strict=true]{<comma separated list of label names>}
```

A usage of `strict=false` would be the usage without optional argument. Of course the *-version has the same option.

With `strict=true` the LaTeX run will be aborted with a corresponding error message, if a label name is used that hasn't been initialized before.

Of course you can use `\cmpdinit` with both keys:

```
\cmpdinit[strict = true, sub-init = true]{<comma separated list of label
                                    names>}
```

If you have chosen a different sublabel marker (see page 21) you need to use it with `\cmpdinit[sub-init = true]` as well.

6.3 Why initiate labels?

As mentioned in the beginning of this section the initiation of labels is not required. There are two possible reasons, why one would like to use them, anyway.

1. The use of `\cmpdinit*` is a way to keep track of which labels you have used and gives you a warning/error message if you use a new one or just have mispelled an existing one. Depending on the number of labels you (have to) use, this can be lots of additional work, though.

2. The use of `\cmpdinit` does the same thing as `\cmpdinit*` *and* declares all the labels in the given order. So if you use it, you won't have to use `\cmpd*` (section 5.1) and `\cmpd+` (section 8.1). This gives you direct control, which number is given to which compound.

7 Resetting the counter

With the command

```
\cmpdreset[<number>]
```

it is possible to reset the counter. Without argument it is reset to `1`. So `\cmpdreset` and `\cmpdreset[1]` are the same. Unlike most other chemnum commands its impact is not only inside the local TeX group but resets the counter globally.

```
1  \cmpdreset The numbering starts with 1 again: \cmpd{h, i, j}
```

The numbering starts with 1 again: **1**, **2**, and **3**

You should be careful with how you use this command! Usually it cannot be recommended to reset the counter. Different compounds can get the same label!

```
Same label: \cmpd{a}, \cmpd{h}
```

Same label: **1**, **1**

8 Special usage

8.1 Moving arguments – section titles

Using `\cmpd` inside a section title in the first instant doesn't seem to be a problem. This doesn't give any errors and the output looks like expected:

```
\section{Compound \cmpd[cmpd-delim]{b}}
\ldots
```

However, if you use `\tableofcontents` – which is rather probable, I guess – something unwanted happens: because the table of contents is at the beginning of the document, the label is declared in the table of contents instead of at the first appearance in the text. There are two ways to circumvent this:

1. You declare your labels in the order you want with `\cmpdinit{}` (see page 7) in the preamble or the beginning of the document.
2. You use `\cmpd+` to set the label. It will read the label from the file `<jobname>.cmpd` (supposing your main file is called `<jobname>.tex`). This means *at least two*, maybe more LaTeX runs are necessary until all labels are set.

The same is true for any other list of moving arguments that is created *before* the actual appearance of the label in the text.

The command

`\cmpd+{<label name>}`

reads the label name that is to be printed from `<jobname>.cmpd`. In a way this is the opposite of `\cmpd*` since it doesn't declare a label but only references and prints it. So in order to use `\cmpd+{<label>}`, `<label>` must have been declared, either with `\cmpd{<label>}` or with `\cmpd*{<label>}`. It doesn't matter, if this declaration is done before or after the usage of `\cmpd+`. This command is useful, if the label is used inside a moving argument like in `\section{}`. **It is not needed when `\cmpdinit{}` is used, see section 6 (pages 7ff).**

```
\subsubsection{Test section title with compound label \cmpd+[cmpd-delim]{f.one}}
% with hyperref:
\subsubsection{Test section title with compound label \texorpdfstring{\cmpd+[cmpd-delim]{f.two}}{(7b)}}
```

8.1.1 Test section title with compound label (7a)

8.1.2 Test section title with compound label (7b)

8.2 Floating environments

The usage of \cmpd inside a float shouldn't be a problem: labels are declared in the order the floating environment appears in the code. However, to be on the safe side you can of course use \cmpd+ here as well.

```
% preamble:
% \usepackage{chemscheme}
% document:
\cmpdreset\cmpd{float1, float2}. Now inside of a scheme:
\begin{scheme}[h]
 \centering
 \caption{This label should be a 2: \cmpd{float2}}
 If 2 = \cmpd{float2} is true, everything's fine.
\end{scheme}
```

1 and **2**. Now inside of a scheme:

If 2 = **2** is true, everything's fine.

Scheme 2: This label should be a 2: **2**

This also works, of the floating environment is placed before the code, for instance by using the option [t].

8.3 Schemes created with external programs

If you create your schemes with external programs like for e. g. ChemDraw and want to use chemnum with them as well, you can use this command:

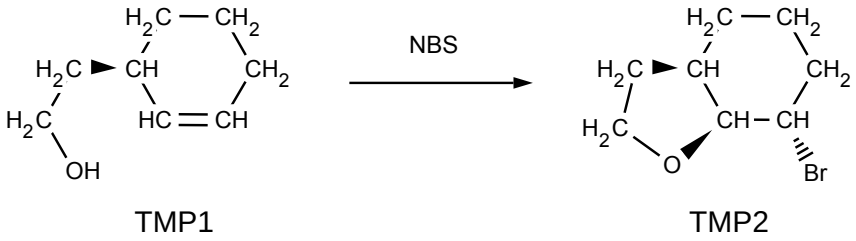
```
\cmpdref[<keyval>][<tag>]{<label name>}
```

This command is inspired by the \schemeref command from the package chemscheme[13]. It works in a very similar way. You create the scheme and save it with temporary labels as an eps file.

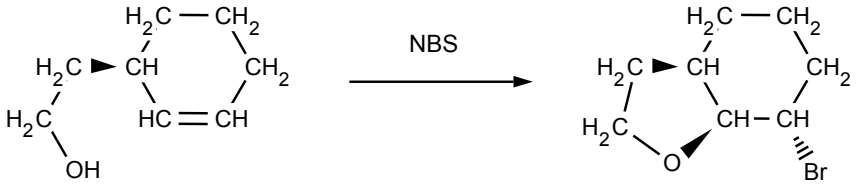
[13] http://www.ctan.org/pkg/chemscheme

```
\begin{scheme}[ht]
 \centering
 \includegraphics{scheme-tmp}
 \caption{Scheme with temporary markers.}
\end{scheme}
\begin{scheme}[ht]
 \centering
 \cmpdref{Alc} % replaces TMP1
 \cmpdref{EtherBr} % replaces TMP2
 % \cmpdref{third} would replace TMP3
 \includegraphics{scheme-tmp}
 \caption{Scheme with automatted labels.}
\end{scheme}
```

So you nummerate the compounds with TMP1, TMP2 etc.. These markers will then



Scheme 3: Scheme with temporary markers.



Scheme 4: Scheme with automatted labels.

be replaced with the corresponding labels. Like chemscheme chemnum uses the \psfrag command to do that. This means you have to either compile via LaTeX, DVIPS, PS2PDF, or for instance use the package auto-pst-pdf[14] to be able to use PDFLaTeX.

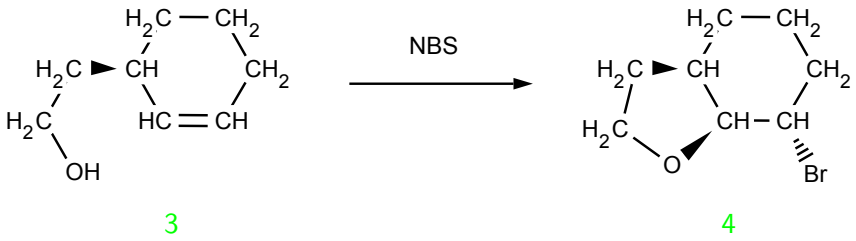
In each case it is important that the marker is saved as *text* in the eps file.

You can customize the labels with the keys presented in section 9.

```
\begin{scheme}[ht]
 \centering
 \cmpdref[cmpd-style=\bf\sf\textcolor{green}]{Alc} % replaces TMP1
```

[14] http://www.ctan.org/pkg/auto-pst-pdf

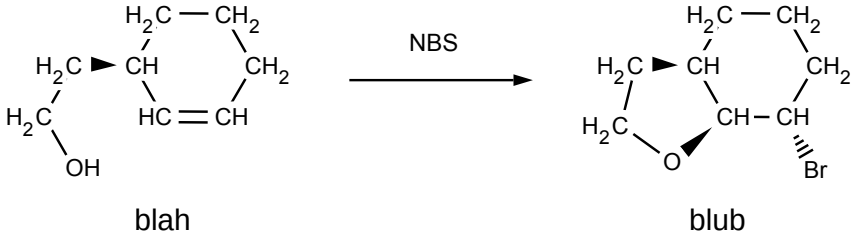
```
    \cmpdref[cmpd-style=\bf\sf\textcolor{green}]{EtherBr} % replaces TMP2
    \includegraphics{scheme-tmp}
    \caption{Scheme with automatted labels.}
  \end{scheme}
```



Scheme 5: Scheme with automatted labels.

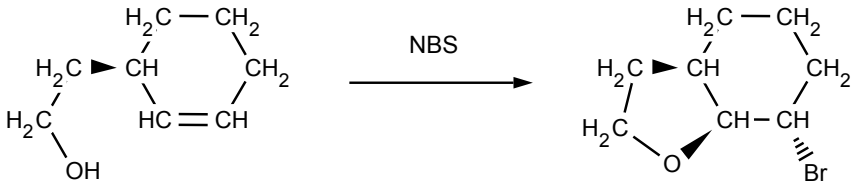
You also can replace arbitrary text by using the second optional argument.

```
  \begin{scheme}[ht]
    \centering
    \includegraphics{scheme-bla}
    \caption{Scheme with arbitrary markers.}
  \end{scheme}
  \begin{scheme}[ht]
    \centering
    \cmpdref[][blah]{Alc}
    \cmpdref[][blub]{EtherBr}
    % \cmpdref{third} would replace TMP1
    \includegraphics{scheme-bla}
    \caption{Scheme with explicitly placed labels.}
  \end{scheme}
```



Scheme 6: Scheme with arbitrary markers.

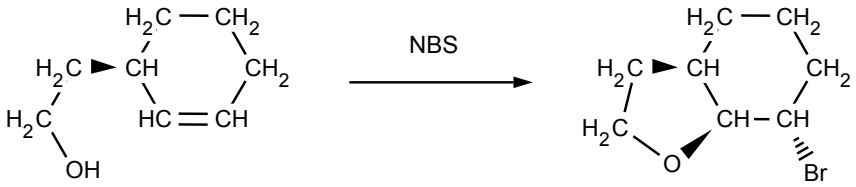
Of course sublabels work in the known way:



3 **4**

Scheme 7: Scheme with explicitly placed labels.

```
\begin{scheme}[ht]
 \centering
 \cmpdref{cpd.red}
 \cmpdref{cpd.ox}
 \includegraphics{scheme-tmp}
 \caption{Scheme with automatted labels and sublabels.}
\end{scheme}
```



5a **5b**

Scheme 8: Scheme with automatted labels and sublabels.

9 Customization

9.1 The principle

chemnum is customized with a key-value system. They can either be used for all following commands with

```
\cmpdsetup{key1 = value1, key2 = value2}
```

or only for a single command directly with

```
\cmpd[key1 = value1, key2 = value2]{<label name>}
```

There are different kinds of keys. Depending on it they can have different kinds of values.

`macro` This kind of keys expects one or more macros as value. The last of them may have a mandatory argument, e. g. `\bfseries` or `\textbf`. chemnum has only one such key, see section 9.2.

`literal` These keys use the given value "as it is", i. e. literally. Most of chemnum's keys are `literal` keys.

`choice` For these keys there is a list of options from which one must be chosen as a value.

`boolean` These keys can either have the value `true` or `false`.

If you use keys without values like `\cmpdsetup{key1,key2}`, default values are used. These are *not* necessarily empty are *not* necessarily the default setting of chemnum. *All* of chemnum's keys have default values. They are listed in section 9.11 on page 23.

The keys are divided in to different classes, which indicate the scope they're used for. They all follow the form `<class>-key = <value>`:

`cmpd` These keys change the whole label or the main label.

`sub` These keys have an impact on sublabels.

`list` Keys which change the output of lists.

`ref` Keys which have an impact on `\cmpdref`.

9.2 Font style

With the key

`cmpd-style = <style>`	type: `macro`

you can change the style with which the labels are printed.

```
% preamble:
% \usepackage[normalem]{ulem}
% document:
\cmpd{a, b, f.two, k}\\                          % default
\cmpd[cmpd-style = \uline]{a, b, f.two, k}\\ % underlined
\cmpd{a, b, f.two, k}\\                          % default again
\cmpdsetup{cmpd-style = \itshape\uline}
\cmpd{a, b, f.two, k}                            % underlined and in italics
```

1, **2**, **7b**, and **9**
1, 2, 7b, and 9
1, **2**, **7b**, and **9**
1, *2*, *7b*, and *9*

9.3 Own labels

With the key

`cmpd-label = <label>`	type: `literal`

one can choose the label associated with the label name.

```
Own label: \cmpd[cmpd-label = XYZ]{1}; in a list \emph{all new compounds}
    get this label! \cmpd[cmpd-label = XYZ]{1, a, m}
```

Own label: **XYZ**; in a list *all new compounds* get this label! **XYZ**, **1**, and **XYZ**

9.4 Own markers in images

If you don't want to use `TMP` as temporary markers for `\cmpdref` (see page 11), you can change this. For example into `tmp`:

```
\cmpdsetup{ref-marker = tmp}
```

9.5 Counter

You can change the format of the counters, too, if you like.

`cmpd-counter = <counter>`	type: `choice`
`sub-counter = <counter>`	type: `choice`

You can choose between `arabic`, `alph`, `Alph`, `greek`, `Greek`, `roman`, `Roman` and `Symbol`. Please note: a change of format only affects labels *which are not yet declared*!

```
\cmpdsetup{cmpd-counter = Alph, sub-counter = arabic}
\cmpd{a, b, f.two, k}\\ % no affect with existing labels
\cmpd{n.one,n.two,o.one}\\
\cmpd{f.three,f.four}\\ % be careful: the main label already exists and isn
    't changed!
\cmpd[cmpd-counter,sub-counter = greek]{p.one,p.two}
```

1, **2**, **7b**, and **9**
J1, **J2**, and **K1**
73 and **74**
12α and **12β**

9.6 Delimiters

Both single labels and whole lists can get delimiters. This is done with the keys

`cmpd-delim = <odelim><cdelim>`	type: `literal`
`list-delim = <odelim><cdelim>`	type: `literal`
values need *two* tokens!	

```
\cmpd{a, b, f.two, k}\\                    % default
\cmpd[cmpd-delim = ()]{a, b, f.two, k}\\ % in braces
\cmpd{a, b, f.two, k}\\                    % default again
\cmpdsetup{cmpd-delim = ()}
\cmpd{a, b, f.two, k}                      % in braces
```

1, **2**, **7b**, and **9**
(**1**), (**2**), (**7b**), and (**9**)
1, **2**, **7b**, and **9**
(**1**), (**2**), (**7b**), and (**9**)

Please be aware that the default values (see pages 15 and 23) of `cmpd-delim` and `list-delim` and the default setting of chemnum are *not* the same. By default chemnum doesn't use delimiters. Also please note that the list delimiters only are used if `\cmpd` contains at least two label names.

```
\cmpd{a, b, f.two, k}\\                    % default
\cmpd[list-delim = {[}{]}]{a, b, f.two, k}\\ % in braces
\cmpd{a, b, f.two, k}\\                    % default again
\cmpdsetup{list-delim = []}
\cmpd{a, b, f.two, k}\\                    % in braces
\cmpd{a}                                   % NOT a list!
```

1, **2**, **7b**, and **9**
[**1**, **2**, **7b**, and **9**]
1, **2**, **7b**, and **9**
[**1**, **2**, **7b**, and **9**]
1

If you generally use labels in braces but want to use a single one without braces, you have several possibilities:

```
\cmpdsetup{cmpd-delim = ()}
normally: \cmpd{b}, \cmpd{c}, \cmpd{d.one}, but sometimes so \cmpd[cmpd-
    delim = ]{e} or so \cmpd-{e}.
```

normally: (**2**), (**3**), (**4a**), but sometimes so **5** or so **5**.

With

```
\cmpd-{<label name>}
\cmpdref-{<label name>}
```

the braces of single labels can easily be removed (but not the ones of a list).

```
\cmpd{a, b, f.two, k}\\  % default
\cmpdsetup{cmpd-delim = (), list-delim = []}
\cmpd{a, b, f.two, k}\\  % braced twice
\cmpd-{a, b, f.two, k}   % braced once
```

1, **2**, **7b**, and **9**
[(**1**), (**2**), (**7b**), and (**9**)]
[**1**, **2**, **7b**, and **9**]

Using `\cmpdsetup{cmpd-delim = }` and `\cmpdsetup{list-delim = }` will restore the default behaviour.

9.7 Suffix and prefix

If you want to, you can also add a prefix and/or a suffix to a compound and/or a list.

`cmpd-prefix = <prefix>`	type: `literal`
`cmpd-suffix = <prefix>`	type: `literal`
`list-prefix = <prefix>`	type: `literal`
`list-suffix = <prefix>`	type: `literal`

The same rule as for the delimiters applies: list attributes are only used with lists, i. e. at least with two items.

An example for the label attributes:

```
\cmpd{a, b, f.two, k}\\ % default
\cmpd[cmpd-prefix = Nr.]{a, b, f.two, k}\\
\cmpdsetup{cmpd-prefix = \textnumero}
\cmpd{a, b, f.two, k}\\
\cmpd{a}              % NOT a list!
```

1, **2**, **7b**, and **9**
Nr. **1**, Nr. **2**, Nr. **7b**, and Nr. **9**
№ **1**, № **2**, № **7b**, and № **9**
№ **1**

An example for the list attributes:

```
\cmpd{a, b, f.two, k}\\ % default
\cmpd[list-prefix = list:]{a, b, f.two, k}\\
\cmpdsetup{list-prefix = collection:}
\cmpd{a, b, f.two, k}\\
\cmpd{a}              % NOT a list!
```

1, **2**, **7b**, and **9**
list: **1**, **2**, **7b**, and **9**
collection: **1**, **2**, **7b**, and **9**
1

9.8 Separators

9.8.1 In case of labels

If you have labels with sublabels, you can use a separator symbol:

`cmpd-sep = <separator>`	type: `literal`

```
\cmpd{a, b, f.two, f.three, k}\\ % default
\cmpd[cmpd-sep = -]{a, b, f.two, f.three, k}\\
\cmpdsetup{cmpd-sep = $\cdot$}
\cmpd{a, b, f.two, f.three, k}\\
\cmpd[cmpd-sep = :]{a, b, f.two, f.three, k}
```

1, **2**, **7b**, **73**, and **9**
1, **2**, **7-b**, **7-3**, and **9**
1, **2**, **7·b**, **7·3**, and **9**
1, **2**, **7:b**, **7:3**, and **9**

9.8.2 In case of lists – generally

In case of lists you also can set a separator symbol, which determines how the labels are separated from each other. This is the comma as default.

`list-sep = <separator>`	type: `literal`

```
\cmpd{a, b, f.two, k}\\ % default
\cmpd[list-sep = ]{a, b, f.two, k}\\
\cmpd{a, b, f.two, k}\\
\cmpdsetup{list-sep = ;}
\cmpd{a, b, f.two, k}
```

1, **2**, **7b**, and **9**
1 2 7b and **9**
1, **2**, **7b**, and **9**
1; **2**; **7b**; and **9**

9.8.3 In case of lists – last separator

The last separator in a list is a special case. chemnum provides two keys with which it can be customized.

`list-last-sep = <separator>`	type: `literal`
`list-lang = <lang>`	type: `choice`

There is the `choice` key `list-lang`, which changes language dependent settings. You can choose between `US` (default), `GB`, `DE`, `FR`, `ES` and `IT`[15].

```
\cmpd[list-lang = US]{a, b, f.two, k} \cmpd[list-lang = US]{a, b}\\
\cmpd[list-lang = GB]{a, b, f.two, k} \cmpd[list-lang = GB]{a, b}\\
\cmpd[list-lang = DE]{a, b, f.two, k} \cmpd[list-lang = DE]{a, b}\\
\cmpd[list-lang = FR]{a, b, f.two, k} \cmpd[list-lang = FR]{a, b}\\
\cmpd[list-lang = ES]{a, b, f.two, k} \cmpd[list-lang = ES]{a, b}\\
\cmpd[list-lang = IT]{a, b, f.two, k} \cmpd[list-lang = IT]{a, b}
```

1, **2**, **7b**, and **9** **1** and **2**
1, **2**, **7b** and **9** **1** and **2**
1, **2**, **7b** und **9** **1** und **2**
1, **2**, **7b** et **9** **1** et **2**
1, **2**, **7b** y **9** **1** y **2**
1, **2**, **7b** e **9** **1** e **2**

With the key `list-last-sep` you also can set the last separator individually.

```
\cmpdsetup{list-lang = GB}%
\cmpd[list-last-sep = {and also}]{a, b, f.two, k}\\
\cmpd[list-last-sep = ]{a, b, f.two, k}\\
\cmpd[list-last-sep = {as well as}]{a, b, f.two, k}\\
\cmpd[list-last-sep = empty]{a, b, f.two, k}
```

1, **2**, **7b** and also **9**
1, **2**, **7b** **9**
1, **2**, **7b** as well as **9**
1, **2**, **7b**, **9**

`empty` is a special value. If it is used, the separator set with `list-sep` is used for the whole list.

[15] If you're missing a language or have caught me with an error please send me an e-mail.

```
\cmpdsetup{list-last-sep = empty}%
\cmpd{a, b, f.two, k}\\
\cmpd[list-sep = ;]{a, b, f.two, k}\\
\cmpd[list-sep = {\ and}]{a, b, f.two, k}
```

1, **2**, **7b**, **9**
1; **2**; **7b**; **9**
1 and **2** and **7b** and **9**

9.9 Sublabel marker

As a default setting chemnum uses the point `.` as a token to divide main label names from sub label names. You can change that as you like. Tokens you *can't use* are `,` `%` `#` and *you shouldn't* use `@`.

`sub-marker = <separator>`	type: `literal`

```
\cmpdsetup{sub-marker = !}%
\cmpd{f!one, g!two}\\
\cmpd[sub-marker= +]{f+one, g+two}\\
\cmpd[sub-marker= ~]{f~one, g~two}\\
\cmpd[sub-marker= &]{f&one, g&two}\\
\cmpd[sub-marker= *]{f*one, g*two}
```

7a and **8b**
7a and **8b**
7a and **8b**
7a and **8b**
7a and **8b**

You should – not only for the sake of consistency – decide only *once* in the beginning of your document *before* the `\tableofcontents` or in the preamble, which marker you use. Otherwise, the `\tableofcontents` won't know the marker anymore and your labels might be at least displayed wrong in the table of contents. Also you need to setup the marker *before* you use it with `\cmpd` or `\cmpdinit` etc.

9.10 Lists and ranges of sublabels

Sometimes it can be useful to display a label with a list or a range of sublabels. Suppose you have compounds **13a**, **13b**, **13c**, **13d**, and **13e** which for example differ in their substituents. It can be useful to refer to them all at once: **13a-e**.

chemnum provides two keys to create such an output and two keys to customize the range.

`cmpd-all = <bool>`	type: `boolean`
`sub-list = {<list of sublabel names>}`	type: `other`
`sub-range-sep = <separator>`	type: `literal`
`sub-range-marker = <marker>`	type: `literal`

The type of the key `sub-list` differs from other keys. It is similar to `literal`. Best you look at the example below.

```
List of labels: \cmpd{q.one, q.two, q.three, q.four, q.five} \\
All at once: \cmpd[cmpd-all]{q}
```

List of labels: **13a**, **13b**, **13c**, **13d**, and **13e**
All at once: **13a-e**

If you don't want to address all sublabels but some of them, you can use the key `sub-list`. As input you write a comma separated list of sublabel names associated with the main label you use.

```
\cmpd[sub-list={one,three,four}]{q}
```

13a,c,d

You also can use this key to print a range of sublabels. As a marker between the names of the sublabels two points `..` are used.

```
\cmpd[sub-list={two..four}]{q}\\
\cmpd[sub-list={one,three..five}]{q}\\
\cmpd[sub-list={one..three,five}]{q}
```

13b-d
13a,c-e
13a-c,e

With the keys `sub-range-sep` and `sub-range-marker` you can both change the symbol that's used to indicate the range and the marker you use to input a range.

```
\cmpdsetup{sub-range-sep = {--}, sub-range-marker = : }
\cmpd[sub-list={two:four}]{q}
```

13b–d

In order to get the right sublabels the command needs to know, which sublabels have been declared. This means that they're read from the `<jobname>.cmpd` file (supposing your source file is called `<jobname>.tex`). So you may need to compile twice (or more) until the labels are printed right.

For the time being the `sub-list` key requires a little caution. Unfortunately automatic sorting isn't quite as trivial as one would wish. So you have to use the sublabel names in the right order if you want to avoid labels like these:

```
\cmpd[sub-list={five..three}]{q} or \cmpd[sub-list={three,one,four}]{q}.
```

13e-c or **13c,a,d**.

9.11 Overview

In this section all keys are listed. Please be aware that the default values of `cmpd-delim` and `list-delim` and the default setting of chemnum are *not* the same. By default chemnum doesn't use delimiters.

key	default	type
cmpd-style	\textbf	macro
cmpd-label		literal
cmpd-delim[a]	()	literal
cmpd-odelim		literal
cmpd-cdelim		literal
cmpd-prefix		literal
cmpd-suffix		literal
cmpd-sep		literal
cmpd-counter	arabic	choice
cmpd-all	true	boolean
sub-marker	.	literal
sub-counter	alph	choice
sub-list		other[b]
sub-range-sep	-	literal
sub-range-marker	..	literal
list-delim[a]	()	literal
list-odelim		literal
list-cdelim		literal
list-prefix		literal
list-suffix		literal
list-sep	,	literal
list-lang	US	choice
list-last-sep	and[c]	literal
ref-tag	TMP	literal

a value needs *two* tokens.

b input is a comma separated list, see p. 22.

c depends on the value of `list-lang`.

10 List of commands

In this section all commands provided by chemnum as well as their variants are listed.

command	description
\cmpd[]{}	basic command, declares and prints labels or lists of labels, see pages 5ff.
\cmpd*{}	invisible, declares label, see page 5
\cmpd-[]{}	without delimiters, see page 18
\cmpd+-[]{}	label is read from auxiliary file, see page 10
\cmpdinit{}	initialise and declare labels, see pages 7f.
\cmpdinit*{}	initialise labels, see pages 7f.
\cmpdref[][]{}	replace temporary markers in eps files by labels, see page 11
\cmpdref-[][]{}	replace temporary markers in eps files by labels without delimiters, see page 18
\cmpdreset[]	reset counter, siehe page 9
\cmpdsetup{}	setup chemnum, siehe pages 14ff.

11 Credits

I would like to thank Joseph Wright and Russell Hewitt who provided me with valuable feedback and suggestions to improve chemnum.