Adding label functionality to `scrlttr2` and `scrletter` using `makelabels.lco`

Markus Kohm

v1.0 2021/08/14*

Abstract

In opposite to the standard letter class, the KOMA-Script letter class `scrlttr2` and the KOMA-Script letter package `scrletter` do not provide generation of a label page using `\makelabels`. But KOMA-Script provides the option to add new letter features using LCO files. `makelabels.lco` is such a LCO file. It provides `\makelabels` similar to the standard letter class. The new `\makelabels` has a yet very rudimentary configurability but much more than the standard letter class provides. However, it is also as much compatible as needed that packages like `envlab` can be used. From version 1.0 `makelabels.lco` is implemented using `expl3`.

Contents

1 Installation 1

2 Basic Usage 2

3 Advanced Usage 2

4 Simple Example 3

5 Example using additional label packages 4

6 Implementation 6

Change History 13

Index 13

1 Installation

It is recommended to use the package installer of the TEX distribution. If you don't use a TEX distribution with package installer, that provides `makelabels.lco`, please see the advanced installation information in the `README.md`.

---

*For support and/or bug reports see the project pages.

2 Basic Usage

First of all please note, that `makelabels.lco` needs LaTeX from version 2020/10/01. It is recommended to use at least LaTeX 2021/06/01, because this is the LaTeX version used for development of `makelabels.lco`. Also KOMA-Script version 3.34 or newer is recommended. If you would use an older KOMA-Script version you would do this on your own risk. Please do not expect any support in this case.

To use the `makelabels.lco` you have to use either KOMA-Script letter class `scrlttr2` or `scrletter` or the KOMA-Script letter package `scrletter`. After loading the class resp. package you have to load `makelabels.lco` using:

```
\LoadLetterOption{makelabels}
```

in your document preamble. Note: This LCO cannot be loaded after `\begin{document}`.

Loading the LCO file already activates generation of label information inside the `aux`-file of your document. Nothing else.

`\makelabels` To activate the label generation you have to add

```
\makelabels
```

to your document preamble after loading `makelabels.lco` (see above). Now at the end of the document, after printing all letters `makelabels` generates one or more additional sheets with address labels. The default label sheet is of type Avery 5162. This is a label sheet with seven rows and two columns of labels. It is compatible with several other Avery label types.

`\selectlabeltype` If you need more than one label per letter or another label type you can use:

```
\selectlabeltype[⟨integer⟩]{⟨string⟩}
```

This selects ⟨*integer*⟩ labels of type ⟨*string*⟩ for each following letter (inclusive the current letter, if used between `\begin{letter}` and `\end{letter}`. See Table 1 for the allowed ⟨*string*⟩ arguments and the corresponding label types.

3 Advanced Usage

`\makelabels_add_label_type:nn` Advanced users can add their own label sheet definitions. But currently there is only an `expl3` interface for this. If you do not know `expl3`, please stop reading. There is no support for this feature, currently.

⟨*string*⟩	**Specification**	
	Measure	Value
`avery_5162`[a]	sheet height	11 in
	sheet width	8.5 in
	sheet top margin	0.845 in
	sheet bottom margin	0.845 in
	sheet left margin	0.167 in
	sheet right margin	0.167 in
	label height	1.330 in
	label width	4 in
	horizontal distance	0.166 in
	vertical distance	0 pt
	label left margin	5 pt
	label right margin	5 pt
	label top margin	0 pt
	label bottom margin	0 pt
	rows	7
	columns	2

Table 1: Known Label Types

---

[a] https://www.avery.com/products/labels/5162

```
\makelabels_add_label_type:nn ⟨string⟩
{
sheet height         = ⟨dim_1⟩,
sheet width          = ⟨dim_2⟩,
sheet top margin     = ⟨dim_3⟩,
sheet bottom margin  = ⟨dim_4⟩,
sheet left margin    = ⟨dim_5⟩,
sheet right margin   = ⟨dim_6⟩,
vertical distance    = ⟨dim_7⟩,
horizontal distance  = ⟨dim_8⟩,
label height         = ⟨dim_9⟩,
label width          = ⟨dim_10⟩,
label top margin     = ⟨dim_11⟩,
label bottom margin  = ⟨dim_12⟩,
rows                 = ⟨int_1⟩,
columns              = ⟨int_2⟩
}
```

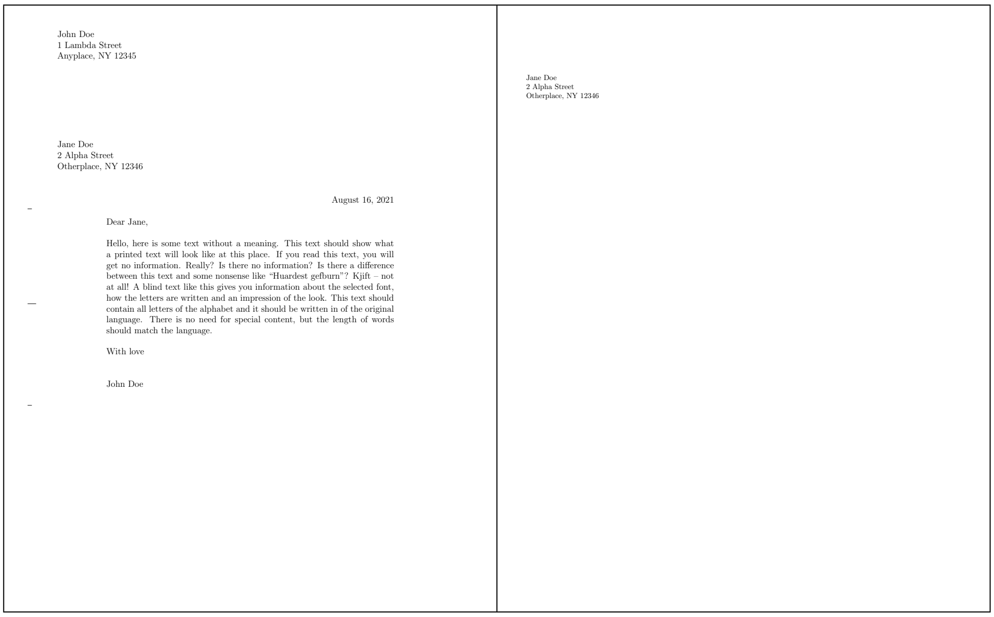
Currently all ⟨dim_x⟩, which are not specified, will be 0 pt and all ⟨int_x⟩, which are not specified, will be 1.

The names of the properties should be self-explaining. If not, don't use it!

Note: You can participate in the development of `makelabels.lco` by posting and explaining your own label specifications.

4 Simple Example

A very simple example for using `makelabels.lco` would be:



John Doe
1 Lambda Street
Anyplace, NY 12345

Jane Doe
2 Alpha Street
Otherplace, NY 12346

August 16, 2021

Dear Jane,

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

With love

John Doe

Jane Doe
2 Alpha Street
Otherplace, NY 12346

Figure 1: The letter and the label sheet of the simple example of section 4.

```
\documentclass[paper=letter]{scrletter}
\LoadLetterOptions{UScommercial9,makelabels}
\usepackage[english]{babel}
\usepackage{blindtext}
\setkomavar{fromname}{John Doe}
\setkomavar{fromaddress}{1 Lambda Street\\Anyplace, NY 12345}
\makelabels
\begin{document}
\begin{letter}{Jane Doe\\2 Alpha Street\\Otherplace, NY 12346}
\opening{Dear Jane,}
\blindtext
\closing{With love}
\end{letter}
\end{document}
```

It would produce the two pages shown in Figure 1.

If you like, you could add an additional

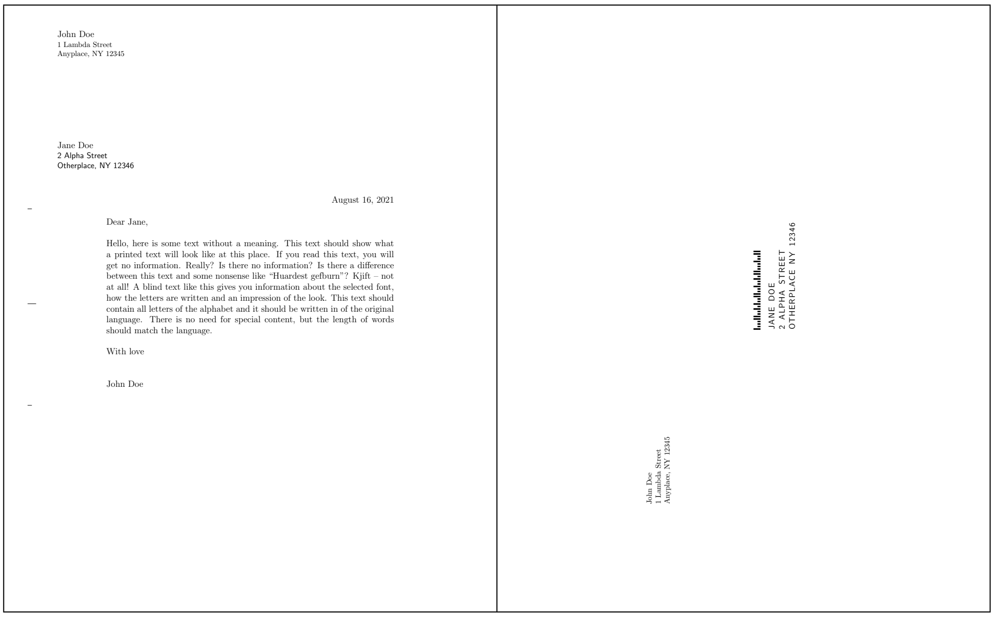
```
\selectlabeltype[14]{avery_5162}
```

after `\begin{document}`. In this case, you would get a whole sheet of 14 labels (7 rows by 2 columns).

5 Example using additional label packages

As already mentioned in the abstract you can use `makelabels.lco` together with package `envlab`. In this case, it is important to load `makelabels.lco` before package `envlab`:

```
\documentclass[paper=letter]{scrletter}
\LoadLetterOptions{UScommercial9,makelabels}
```



John Doe
1 Lambda Street
Anyplace, NY 12345

Jane Doe
2 Alpha Street
Otherplace, NY 12346

August 16, 2021

Dear Jane,

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

With love

John Doe

JANE DOE
2 ALPHA STREET
OTHERPLACE NY 12346

John Doe
1 Lambda Street
Anyplace, NY 12345

Figure 2: The letter and the envelope sheet of the `envlab` example of section 5.

```
\usepackage[english]{babel}
\usepackage{blindtext}
\setkomavar{fromname}{John Doe}
\setkomavar{fromaddress}{1 Lambda Street\\Anyplace, NY 12345}

\usepackage[centerenvelopes,businessenvelope]{envlab}
\setlength{\FromAddressTopMargin}{0.25in}
\setlength{\FromAddressLeftMargin}{0.25in}
\makelabels

\begin{document}

\begin{letter}{Jane Doe\\2 Alpha Street\\Otherplace, NY 12346}
\opening{Dear Jane,}
\blindtext
\closing{With love}
\end{letter}
\end{document}
```

It would produce the two pages shown in Figure 2.

Please note: package `envlab` redefines macro `\@toaddressfont`, which is also an internal macro of `scrlttr2` or `scrletter`. The KOMA-Script classes resp. the KOMA-Script package uses this to store the font of the recipient’s address. So loading `envlab` will change the font of the recipient’s address. To avoid this you could use additional preamble code:

```
\makeatletter
\AtBeginDocument{%
  \immediate\write\@auxout{%
    \string\def\string\@toaddressfont{%
```

```
      \expandafter\detokenize\expandafter{\@toaddressfont}}}%
  \setkomafont{toaddress}{}% set the recepient’s address font
}
\makeatother
```

This delays the `envlab`’s redefinition of `\@toaddressfont` until the *labels* are generated.

6 Implementation

```
⟨*lco⟩
⟨@@=makelabels⟩
```

Load time actions

We need at least LaTeX 2020/10/01.

```
\newcommand*{\makelabels@fatal@format@error}{%
  \GenericError{(makelabels)\@spaces\@spaces\@spaces\@spaces}%
               {Fatal makelabels.lco error: LaTeX too old.}%
               {See the makelabels.lco documentation for explanation.}%
               {At least LaTeX 2020/10/01 is needed}%
  \endinput
}
\@ifundefined{IfFormatAtLeastTF}{%
  \makelabels@fatal@format@error
}{%
  \IfFormatAtLeastTF{2020/10/01}{}{\makelabels@fatal@format@error}%
}
```

With this version of LaTeX, we do not need to load `expl3` explicitly but can just switch to the syntax.

```
\ExplSyntaxOn
```

This LCO file can be used in the document preamble only. If we are already have begun the document, this would be fatal.

```
\msg_new:nnn { makelabels } { onlypreamble }
  { Sorry,~but~‘makelabels.lco’~can~be~used~in~the~document~preamble~only. }
\if@atdocument
  \msg_fatal:nnn { makelabels } { onlypreamble }
\fi
```

`\makelabels_add_label_type:nn` Labels have only a width and height. They are placed in a number of rows and columns at a label sheet. The sheet has also a width and height. There is a margin left of the first label, above the first label, right of the first label and below the last label. And there may be a horizontal and a vertical distance between the labels. See the definition of Avery 5162 labels for all properties.

Note: currently all properties have to be setup correctly.

```
\cs_new_nopar:Nn \makelabels_add_label_type:nn
  {
    \prop_new:c { g__makelabels_label_type_#1_prop }
    \prop_set_from_keyval:cn { g__makelabels_label_type_#1_prop } { #2 }
  }
```

(*End definition for* `\makelabels_add_label_type:nn`. *This function is documented on page* **??**.)

The first label type we define is `avery_5162`:

```
\makelabels_add_label_type:nn { avery_5162 }
  {
    sheet height              = 11 in,
    sheet width               = 8.5 in,
    sheet top margin          = 0.845 in,
    sheet bottom margin       = 0.845 in,
    sheet left margin         = 0.167 in,
    sheet right margin        = 0.167 in,
    label height              = 1.330 in,
    label width               = 4 in,
    horizontal distance       = 0.166 in,
    vertical distance         = 0 pt,
    label left margin         = 5 pt,
    label right margin        = 5 pt,
    label top margin          = 0 pt,
    label bottom margin       = 0 pt,
    rows                      = 7,
    columns                   = 2,
  }
```

We also have to take care that at the end of each letter the label is written to the `aux`-file.

`\__makelabels_Ifkomavarempty` ⋆

To be able to write to the `aux`-file, we need an expandable verion of KOMA-Script's `\Ifkomavarempty`. To make simple wrapping possible, we define it as an internal document command.

```
\NewExpandableDocumentCommand \__makelabels_Ifkomavarempty { s m m m }
  {
    \IfBooleanTF { #1 }
      {
        \exp_args:Nc \__makelabels_if_empty_var_or_name:NTF{scr@#2@name}{#3}{#4}
      }
      {
        \exp_args:Nc \__makelabels_if_empty_var_or_name:NTF{scr@#2@var}{#3}{#4}
      }
  }
```

`\__makelabels_if_empty_var_or_name_p:N` ★
`\__makelabels_if_empty_var_or_name:N`*`TF`* ★

This is the real internal command. Now, the first argument is not the variable any longer, but the macro storing the variable or name.

```
\prg_new_conditional:Nnn \__makelabels_if_empty_var_or_name:N { p, T, F, TF }
  {
    \if_cs_exist:N #1
      \if_meaning:w #1 \c_empty_tl
        \prg_return_true: \else: \prg_return_false: \fi:
    \else:
      \prg_return_false:
    \fi:
  }
```

`\__makelabels_usekomavar` ★

To be able to write to the `aux`-file, we need an expandable version of KOMA-Script's `\usekomavar`. To make simple wrapping possible, we define it as an internal document command.

```
\NewExpandableDocumentCommand \__makelabels_usekomavar { s o m }
  {
    \IfValueTF{#2}{#2}{\use:e}{\cs:w scr@#3@\IfBooleanTF{#1}{name}{var}\cs_end:}
  }
```

Now, after defining new, expandable versions of most of the usually needed not expandable KOMA-Script user command, we can write to the `aux`-file at the end of every letter. Note, we have to write immediately (using `\iow_now:Nx`), to use the current definition of the commands redefined in the local group.

```
\AtEndLetter
  {
    \if@filesw
      \group_begin:
        \cs_set:Npn \Ifkomavarempty { \__makelabels_Ifkomavarempty }
        \cs_set:Npn \usekomavar { \__makelabels_usekomavar }
        \iow_now:Nx \@mainaux
          {
            \token_to_str:N \@mlabel
            \iow_char:N \{ \usekomavar{backaddress} \iow_char:N \}
            \iow_char:N \{ \usekomavar{toname} \iow_char:N \\ \iow_char:N \\
                           \usekomavar{toaddress} \iow_char:N \}
          }
      \group_end:
    \fi
  }
```

The preamble commands

`\makelabels` Preamble only command to activate the label generation via the `aux`-file.

```
\cs_new:Npn \makelabels
  {
    \hook_gput_code:nnn { begindocument } { makelabels.lco }
      {
        \cs_set_eq:NN \@startlabels \startlabels
        \cs_set_eq:NN \@mlabel \mlabel
        \cs_set_eq:NN \@mlabeltype \mlabeltype
        \if@filesw
          \iow_now:Nn \@mainaux { \@startlabels }%
        \fi
      }
    \hook_gput_code:nnn { enddocument / afterlastpage } { makelabels.lco }
      {
        \if@filesw
          \iow_now:Nn \@mainaux { \clearpage } %
        \fi
      }
  }
\@onlypreamble \makelabels
```

(*End definition for* `\makelabels`*. This function is documented on page* **??**.)

`\selectlabeltype`

Used in the document preamble or inside the document to select another label type.

```
\newcommand*{\selectlabeltype}[2][1]{
  \cs_if_exist:cTF { g__makelabels_label_type_#2_prop }
    {
      \if@filesw
        \iow_now:Nn \@mainaux
          {
            \@mlabeltype { #1 } { #2 }
          }
      \fi
    }
    {
      \msg_error:nnn { makelabels } { unkown label type } { #1 }
    }
}
```

(*End definition for* `\selectlabeltype`*. This function is documented on page* **??**.)

The aux-file commands

`\@startlabels`
`\@mlabel`
`\@mlabeltype`

All these are dummies until `\makelabels` has been used.

```
\hook_gput_code:nnn { begindocument } { makelabels.lco }
  {
    \if@filesw
      \iow_now:Nn \@mainaux { \providecommand* { \@startlabels } { } }
      \iow_now:Nn \@mainaux { \providecommand* { \@mlabel }[2] { } }
      \iow_now:Nn \@mainaux { \providecommand* { \@mlabeltype } [ 2 ] { } }
    \fi
  }
```

(*End definition for* `\@startlabels`*,* `\@mlabel`*, and* `\@mlabeltype`*. These functions are documented on page* **??**.)

`\mlabeltype`

Select generating #1 labels of type #2.

```
\int_new:N \g__makelabels_label_repeat_int
\int_set_eq:NN \g__makelabels_label_repeat_int \c_one_int
```

```
\str_new:N \g__makelabels_label_type_str
\str_set:Nn \g__makelabels_label_type_str { avery_5162 }
\cs_new:Npn \mlabeltype #1#2
  {
    \int_set:Nn \g__makelabels_label_repeat_int { #1 }
    \str_set:Nn \g__makelabels_label_type_str { #2 }
  }
```

(*End definition for* `\mlabeltype`*. This function is documented on page* **??***.*)

`\startlabels` Start a new label page. We have to setup several page layout parameter depending on the current label type `\g__makelabels_label_type_str`.

```
\cs_new:Npn \startlabels
  {
    \clearpage
    \if@twocolumn \onecolumn \fi
    \pagestyle{empty}
    \cs_set_eq:NN \@texttop \relax
    \dim_set_eq:NN \headheight \c_zero_dim
    \dim_set_eq:NN \headsep \c_zero_dim
    \dim_set_eq:NN \lineskip \c_zero_dim
    \__makelabels_prop_get_dim:nN { sheet height } \paperheight
    \__makelabels_prop_get_dim:nN { sheet top margin } \topmargin
    \dim_sub:Nn \topmargin { 1in }
    \__makelabels_prop_get_dim:nN { sheet width } \paperwidth
    \__makelabels_prop_get_dim:nN { sheet left margin } \oddsidemargin
    \dim_sub:Nn \oddsidemargin { 1in }
    \dim_set_eq:NN \evensidemargin \oddsidemargin
    \__makelabels_prop_get_int:nN { rows } \g__makelabels_rows_int
    \__makelabels_prop_get_int:nN { columns } \g__makelabels_columns_int
    \__makelabels_prop_get_dim:nN { label height } \g__makelabels_label_height_dim
    \__makelabels_prop_get_dim:nN { label width } \g__makelabels_label_width_dim
    \__makelabels_prop_get_dim:nN { vertical distance } \g__makelabels_vertical_skip_dim
    \__makelabels_prop_get_dim:nN { horizontal distance } \columnsep
    \dim_set:Nn \textheight
      {
        ( \g__makelabels_label_height_dim + \g__makelabels_vertical_skip_dim ) * \g__makelabe
        - \g__makelabels_vertical_skip_dim
      }
    \dim_set:Nn \textwidth
      {
        ( \g__makelabels_label_width_dim + \columnsep ) * \g__makelabels_columns_int
        - \columnsep
      }
    \__makelabels_prop_get_dim:nN { label top margin } \g__makelabels_label_top_margin_dim
    \__makelabels_prop_get_dim:nN { label bottom margin } \g__makelabels_label_bottom_margin_
    \__makelabels_prop_get_dim:nN { label left margin } \g__makelabels_label_left_margin_dim
    \__makelabels_prop_get_dim:nN { label right margin } \g__makelabels_label_right_margin_di
    \activateareas
    \fontsize{10pt}{12pt}\selectfont
    \dim_set_eq:NN \boxmaxdepth \c_max_dim
    \int_set_eq:NN \g__makelabels_row_int \c_one_int
    \int_set_eq:NN \g__makelabels_column_int \c_one_int
    \dim_set:Nn \g__makelabels_label_height_effective_dim
```

```
      {
        \g__makelabels_label_height_dim
        - \g__makelabels_label_top_margin_dim
        - \g__makelabels_label_bottom_margin_dim
      }
    \dim_set:Nn \g__makelabels_label_width_effective_dim
      {
        \g__makelabels_label_width_dim
        - \g__makelabels_label_left_margin_dim
        - \g__makelabels_label_right_margin_dim
      }
    \raggedright
  }

\int_new:N \g__makelabels_rows_int
\int_new:N \g__makelabels_columns_int
\int_new:N \g__makelabels_row_int
\int_new:N \g__makelabels_column_int
\dim_new:N \g__makelabels_label_height_dim
\dim_new:N \g__makelabels_label_width_dim
\dim_new:N \g__makelabels_label_top_margin_dim
\dim_new:N \g__makelabels_label_bottom_margin_dim
\dim_new:N \g__makelabels_label_left_margin_dim
\dim_new:N \g__makelabels_label_right_margin_dim
\dim_new:N \g__makelabels_label_height_effective_dim
\dim_new:N \g__makelabels_label_width_effective_dim
\dim_new:N \g__makelabels_vertical_skip_dim
```

(*End definition for* `\startlabels`. *This function is documented on page* **??**.)

`\__makelabels_prop_get_dim:nN` `\__makelabels_prop_get_dim:nN {⟨property string⟩} {⟨dimension variable⟩}`
`\__makelabels_prop_get_int:nN` `\__makelabels_prop_get_int:nN {⟨property string⟩} {⟨integer variable⟩}`

Get a property from the property list of the current label type `\g__makelabels_label_type_str` and store it as a dimension resp. integer. Unkown properties result in a warning message. Unkown dimenionss are assumed to be zero, unkown integers are assumed to be one.

```
\msg_new:nnn { makelabels } { undefined property }
  { Property~‘#1’~undefined~for~label~type~‘\g__makelabels_label_type_str’.~
    Value~#2~assumed. }

\cs_new:Nn \__makelabels_prop_get_dim:nN
  {
    \prop_get:cnNTF { g__makelabels_label_type_ \g__makelabels_label_type_str _prop } { #1 }
      { \dim_set:Nn #2 \l_tmpa_tl }
      {
        \msg_warning:nnnn { makelabels } { undefined property } { #1 } { zero }
        \dim_set_eq:NN #2 \c_zero_dim
      }
  }

\cs_new:Nn \__makelabels_prop_get_int:nN
  {
    \prop_get:cnNTF { g__makelabels_label_type_ \g__makelabels_label_type_str _prop } { #1 }
      { \int_set:Nn #2 \l_tmpa_tl }
      {
        \msg_warning:nnnn { makelabels } { undefined property } { #1 } { one }
        \int_set_eq:NN #2 \c_one_int
      }
  }
```

`\__makelabels_print_one_label:` Currently we do not support different output routines for different label types. So this command is always the same.

```
\cs_new:Nn \__makelabels_print_one_label:nn
  {
%     \frame{
    \parbox[b][\g__makelabels_label_height_dim]{\g__makelabels_label_width_dim}{
      \skip_vertical:N \g__makelabels_label_top_margin_dim
      \skip_horizontal:N \g__makelabels_label_left_margin_dim
      \parbox[c][\g__makelabels_label_height_effective_dim]{\g__makelabels_label_width_effect
        \ignorespaces #2
      }
      \skip_vertical:N \g__makelabels_label_bottom_margin_dim
    }
%    }
    \int_incr:N \g__makelabels_column_int
    \if_int_compare:w \g__makelabels_column_int > \g__makelabels_columns_int
      \par
      \skip_vertical:N \g__makelabels_vertical_skip_dim
      \int_set_eq:NN \g__makelabels_column_int \c_one_int
      \int_incr:N \g__makelabels_row_int
      \if_int_compare:w \g__makelabels_row_int > \g__makelabels_rows_int
        \clearpage
        \int_set_eq:NN \g__makelabels_row_int \c_one_int
      \fi:
    \else:
      \skip_horizontal:N \columnsep
    \fi:
  }
```

(*End definition for* `\__makelabels_print_one_label:`*.*)

`\mlabel` Output the configurated number of labels.

```
\cs_new:Npn \mlabel #1#2
  {
    \int_step_inline:nnn { 1 } { \g__makelabels_label_repeat_int }
      {
        \__makelabels_print_one_label:nn { #1 } { #2 }
      }
  }
```

(*End definition for* `\mlabel`*. This function is documented on page* ??*.*)

We need to not forget to switch of `expl3` syntax, because this is not a package but a LCO.

```
\ExplSyntaxOff
⟨/lco⟩
```

Change History

v0.5
General: First version released as `mlabel.lco` at https://komascript.de/mlabel.lco . . . 1

v1.0
General: Reimplementation using `expl3` syntax 1

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols

\\ 6, 9, 20, 29, 112
\{ 111, 112
\} 111, 113

I

\Ifkomavarempty *7*, 106

M

\makelabels *2*
\makelabels *9*, 7, 25, 118
makelabels commands:
  \makelabels_add_label_type:nn *3*, 55, 60
makelabels internal commands:
  \g__makelabels_column_int 208, 227, 272, 273, 276
  \g__makelabels_columns_int 185, 197, 225, 273
  __makelabels_if_empty_var_or_name:N 89
  __makelabels_if_empty_var_or_-

name:NTF *8*, 83, 86
__makelabels_if_empty_var_or_name_p:N *8*
__makelabels_Ifkomavarempty *7*, 79, 106
\g__makelabels_label_bottom_margin_dim 201, 213, 231, 269
\g__makelabels_label_height_dim 186, 192, 211, 228, 263
\g__makelabels_label_height_effective_dim 209, 234, 266
\g__makelabels_label_left_margin_dim 202, 218, 232, 265
\g__makelabels_label_repeat_int 159, 160, 165, 288
\g__makelabels_label_right_margin_dim 203, 219, 233
\g__makelabels_label_top_margin_dim 200, 212, 230, 264
\g__makelabels_label_type_str *10*, *12*, 161, 162, 166, 238, 243, 253
\g__makelabels_label_width_dim 187, 197, 217, 229, 263
\g__makelabels_label_width_effective_dim 215, 235, 266
__makelabels_print_one_label: . 260
__makelabels_print_one_label:nn 260, 290
__makelabels_prop_get_dim:nN *12*, 177, 178, 180, 181, 186, 187, 188, 189, 200, 201, 202, 203, 241
__makelabels_prop_get_int:nN *12*, 184, 185, 251
\g__makelabels_row_int 207, 226, 277, 278, 280
\g__makelabels_rows_int 184, 192, 224, 278
__makelabels_usekomavar . . *8*, 98, 107
\g__makelabels_vertical_skip_dim 188, 192, 193, 236, 275
\makelabels_add_label_type:nn *2*
\mlabel 123, 286
\mlabeltype 124, 159

S

\selectlabeltype *2*
\selectlabeltype *2*, 137
\startlabels 122, 168

T

TEX and LATEX 2ε commands:
    \@mlabel 110, 123, 151
    \@mlabeltype 124, 143, 151
    \@startlabels 122, 126, 151
    \makelabels@fatal@format@error 37, 45, 47

U

\usekomavar *8*, 107, 111, 112, 113