TASKS

create horizontal columned lists

Clemens Niederberger

https://github.com/cgnieder/tasks/

contact@mychemistry.eu

Table of Contents

1 Motivation

Originally TASKS has been an integral part of the ExSheets package. However, users told me that it indeed could be useful to have it as a stand-alone package not having to load the whole ExSheets beast just for having the tasks environment available. Since I agree with this the environment has been extracted into a package if its own, TASKS. Since then TASKS has been

distributed as a package of its own but as part of the ExSheets bundle. With v0.10 I decided to make it a completely independent package. So the relation to ExSheets only is a historical one.

The reason for the tasks environment is an unwritten agreement in German maths textbooks (exspecially in (junior) high school textbooks) to organize exercises in columns counting horizontally rather than vertically. That is what tasks primarily is for. If you don't need this feature you're better off using traditional LaTeX lists and the enumitem package for customization.

2 License and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License (LPPL), version 1.3 or later (http://www.latex-project.org/lppl.txt). The software has the status "maintained."

TASKS requires the l3kernel [L3Pa] bundle, xparse,[1] xtemplate and l3keys2e[2] which are part of the l3packages [L3Pb] bundle, epic [Pod86], cntformats [Nie14], and environ [Rob13].

3 How it works

3.1 The Basics

The tasks environment is similar to a list like enumerate but not the same. Here are some of the differences:

- A first difference: there is no pagebreak possible inside an item but only between items.

- A second difference: the enumeration default is a), b), c) …

- A third difference: the body of the tasks environment is split at *every* occurrence of the item separator. For this reason the default separator is not \item but \task so it is unique to this environment only. This directly leads to…

- … a fourth difference: the tasks environment cannot be nested. You can, however, use an itemize environment or another "real" list in it.

- A fifth difference: verbatim material cannot be used in it. You'll have to use \string, \texttt or \detokenize. If this won't suffice then don't use tasks.

\begin{tasks}[⟨*options*⟩](⟨*num of columns*⟩)
  List like environment where the single items are introduced with \task.

Let's see an example:

```
1 % \Sample is defined to contain some sample text:
2 % \def\sample{This is some sample text we will use to create a somewhat
3 %   longer text spanning a few lines.}
4 % \def\Sample{\sample\ \sample\par\sample}
5 Some text before the list.
6 \begin{tasks}
7   \task \Sample
8   \task \Sample
9   \task \Sample
```

---

1. on CTAN as xparse: http://mirrors.ctan.org/macros/latex/contrib/xparse/
2. on CTAN as l3keys2e: http://mirrors.ctan.org/macros/latex/contrib/l3keys2e/

```
10 \end{tasks}
11 And also some text after it.
```

Some text before the list.

a) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

c) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

And also some text after it.

The environment takes the optional argument ($\langle$*num of columns*$\rangle$) with which the number of columns used by the environment is specified.

```
1 \begin{tasks}(2)
2   \task \Sample
3   \task \sample\ \sample
4   \task \sample
5   \task \Sample
6   \task \sample\par\sample
7 \end{tasks}
```

a) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

c) This is some sample text we will use to create a somewhat longer text spanning a few lines.

d) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

e) This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

3.2 Items Spanning More Than One Column

Introduced in version 0.10

Sometimes it may come in handy if an item is allowed to span more than one column. TASKS supports items using the remaining space by adding an optional star to \task:

```
1 \begin{tasks}(3)
2   \task \sample
3   \task* \sample
4   \task* \sample
5   \task \sample
6   \task \sample
7 \end{tasks}
```

a) This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines.

c) This is some sample text we will use to create a somewhat longer text spanning a few lines.

d) This is some sample text we will use to create a somewhat longer text spanning a few lines.

e) This is some sample text we will use to create a somewhat longer text spanning a few lines.

Introduced in version 0.10

TASKS also supports items that span *all* columns in any case by adding an optional bang to `\task`.

```
1 \begin{tasks}(3)
2   \task \sample
3   \task! \sample
4   \task! \sample
5   \task \sample
6   \task \sample
7 \end{tasks}
```

a) This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines.

c) This is some sample text we will use to create a somewhat longer text spanning a few lines.

d) This is some sample text we will use to create a somewhat longer text spanning a few lines.

e) This is some sample text we will use to create a somewhat longer text spanning a few lines.

The optional star has itself an optional argument with parentheses where you can specify the number of columns the item is supposed to span:

```
1  \settasks{debug}
2  \begin{tasks}(4)
3    \task the first
4    \task the second
5    \task the third
6    \task the fourth
7    \task*(3) the fifth item is way too long for this and needs three columns
8    \task the sixth
9    \task the seventh
10   \task*(2) the eighth item is way too long for this and needs two columns
11   \task the nineth
12   \task the tenth
13 \end{tasks}
```

a) the first b) the second c) the third d) the fourth

e) the fifth item is way too long for this and needs three columns f) the sixth

g) the seventh h) the eighth item is way too long for this and needs two columns i) the nineth

j) the tenth

If there are not enough columns left (say two columns but you said \task*(3)) the argument is ignored and the maximum number of remaining columns is used (two in case of our example).

Both optional star and optional bang can be combined with the optional argument for a custom label:

```
1 \begin{tasks}(3)
2   \task \sample
3   \task* \sample
4   \task*[(x)] \sample
5   \task \sample
6   \task \sample
7 \end{tasks}
```

a) This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines.

(x) This is some sample text we will use to create a somewhat longer text spanning a few lines.

c) This is some sample text we will use to create a somewhat longer text spanning a few lines.

d) This is some sample text we will use to create a somewhat longer text spanning a few lines.

Introduced in version 0.9

Forcing a new item line manually is also possible using the following command:

\startnewitemline

Introduce a new line in a tasks environment.

```
1  \begin{tasks}(4)
2    \task the first
3    \task the second
4    \task the third
5    \task the fourth
6    \task \rlap{the fifth item is way too long for this so we start a new row}
7      \startnewitemline
8    \task the sixth
9    \task the seventh
10   \task \rlap{the eighth item also is too long} \startnewitemline
11   \task the nineth
12   \task the tenth
```

```
13  \end{tasks}
```

a) the first   b) the second   c) the third   d) the fourth

e) the fifth item is way too long for this so we start a new row

f) the sixth   g) the seventh   h) the eighth item also is too long

i) the nineth   j) the tenth

While this works it also needs a bit of care since the width of the items doesn't change which means in order to use the full width you'd have to use trickery like `\rlap` which then means the danger of the item text sticking into the margin...

4 Available Options

Changed in version 0.10

The TASKS package does not have any package options.

The environment tasks has a number of options, though, namely the following ones that can be set using a setup command:

`\settasks{⟨options⟩}`
 Setup command for TASKS.

`style` = {⟨*instance*⟩}   (initially empty)
 Choose the instance to be used. Read more on this in section 8.1.

`counter-format` = {⟨*counter specs*⟩}   Default: tsk[a])

Introduced in version 0.9

 Sets a custom label. The letters `tsk` are replaced with the task-counter. An optional argument directly following these letters specifies the counter format: 1: `\arabic`, a: `\alph`, A: `\Alph`, r: `\roman` and R: `\Roman`.

`label-format` = {⟨*code*⟩}   (initially empty)

Changed in version 0.9

 Can be used to apply a formatting like, *e. g.*, `\bfseries` to the labels.

`label` = {⟨*code*⟩}   (initially empty)

Changed in version 0.9

 Overwrite the automatic label to a custom one.

`label-width` = {⟨*dim*⟩}   Default: 1em
 Sets the width of the item labels.

`label-offset` = {⟨*dim*⟩}   Default: .3333em

Introduced in version 0.7

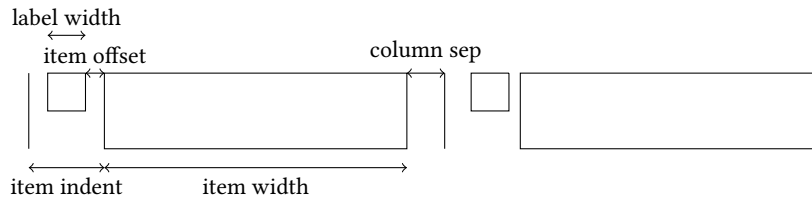 Sets the offset, *i. e.*, the distance between label and item.

FIGURE 1: A visual representation of the used lengths.

`item-format` = {⟨*code*⟩}   (initially empty)

Can be used to apply a formatting like, *e. g.*, `\bfseries` to the items. This may be a macro accepting the item as mandatory argument.

`item-indent` = {⟨*dim*⟩}   Default: `2.5em`

The indent of an item, *i. e.*, the horizontal space available for both label and label-offset. If

$$\text{indent} = \text{label-width} + \text{label-offset}$$

the label will align with the textblock above (if `label-align` = {left} is set). Please see figure 1 for a sketch of the available lengths and how they are set.

`column-sep` = {⟨*dim*⟩}   Default: `0pt`

A horizontal length that is inserted between columns ot items.

`label-align` = left|right|center   Default: `left`

Determines how the labels are aligned within the label-box whose width is set with `label-width`.

`before-skip` = {⟨*skip*⟩}   Default: `0pt`

Sets the skip before the list.

`after-skip` = {⟨*skip*⟩}   Default: `0pt`

Sets the skip after the list.

`after-item-skip` = {⟨*skip*⟩}   Default: `1ex plus 1ex minus 1ex`

This vertical skip is inserted between rows of items.

`resume` = <u>true</u>|false   Default: `false`

The enumeration will resume from a previous `tasks` environment. In order to use this option properly you shouldn't mix different `tasks` environments that both count their items.

`debug` = <u>true</u>|false   Default: `false`

If set to true `\fboxsep` is set to `0pt` inside the `tasks` environment and `\fbox` is used to draw a frame around the label boxes and the item boxes.

Now the same list as above but with three columns and a different label:

```
1 \begin{tasks}[counter-format=(tsk[r]),label-width=4ex](2)
2   \task \Sample
3   \task \sample\ \sample
4   \task \sample
5   \task \Sample
6   \task \sample\par\sample
7 \end{tasks}
```

(i) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

(ii) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

(iii) This is some sample text we will use to create a somewhat longer text spanning a few lines.

(iv) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

(v) This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

Let's use it inside a question, *i. e.*, inside ExSheets' question environment:

```
1 % since settings are local the following ones will be lost
2 % outside this example;
3 \settasks{
4   counter-format = qu.tsk ,
5   item-indent    = 2em ,
6   label-width    = 2em ,
7   label-offset   = 0pt
8 }
9 \begin{question}[type=exam]{4}
10   I have these two tasks for you. Shall we begin?
11   \begin{tasks}(2)
12     \task The first task: easy!
13     \task The second task: even more so!
14   \end{tasks}
15 \end{question}
16 \begin{solution}[print]
17   Now, let's see\ldots\ ah, yes:
18   \begin{tasks}
19     \task This is the first solution. Told you it was easy.
20     \task This is the second solution. And of course you knew that!
21   \end{tasks}
22 \end{solution}
```

Question 1.   4 P.

I have these two tasks for you. Shall we begin?

1.1   The first task: easy!   1.2   The second task: even more so!

Solution 1.

Now, lets see… ah, yes:

1.1   This is the first solution. Told you it was easy.

1.2   This is the second solution. And of course you knew that!

Finally let's see what the debug option does (you could see it already on page 6):

```
1 \settasks{debug}
2 \begin{tasks}(2)
```

```
3    \task \Sample
4    \task \Sample
5 \end{tasks}
```

a) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

b) This is some sample text we will use to create a somewhat longer text spanning a few lines. This is some sample text we will use to create a somewhat longer text spanning a few lines.

This is some sample text we will use to create a somewhat longer text spanning a few lines.

5 Available Instances

There are currently three additional instances for the `tasks` object available:

itemize uses `\labelitemi` as labels.

enumerate enumerates the items with 1., 2., …

multiplechoice a – well – 'multiple choice' list.

```
1  \begin{tasks}[style=itemize](2)
2    \task that's just how\ldots
3    \task \ldots we expected
4  \end{tasks}
5  \begin{tasks}[style=enumerate](2)
6    \task that's just how\ldots
7    \task \ldots we expected
8  \end{tasks}
9  \begin{tasks}[style=multiplechoice](2)
10   \task that's just how\ldots
11   \task \ldots we expected
12 \end{tasks}
```

- thats just how…   • …we expected

1. thats just how…   2. …we expected

☐ thats just how…   ☐ …we expected

6 Custom Labels

If you want to change a single label inside a list, you can use the optional argument of `\task`. This will temporarily overwrite the default label.

```
1 \begin{tasks}[style=itemize]
2   \task a standard item
3   \task another one
4   \task[+] a different one
5   \task and another one
6 \end{tasks}
```

- a standard item

- another one

+ a different one

- and another one

7 New Tasks

It is possible to add custom environments that work like the tasks environment.

`\NewTasks[⟨options⟩]{⟨name⟩}[⟨separator⟩](⟨cols⟩)`
  Define environment ⟨*name*⟩ that uses ⟨*separator*⟩ to introduce a new item. Default for ⟨*separator*⟩ is `\task`, default for ⟨*cols*⟩ is 1. The ⟨*options*⟩ are the ones described in section 4.

`\RenewTasks[⟨options⟩]{⟨name⟩}[⟨separator⟩](⟨cols⟩)`
  Renew environment previously defined with `\NewTasks`.

The tasks environment is defined as follows:

```
1 \NewTasks{tasks}
```

The separator does not have to be a control sequence:

```
1 % preamble:
2 % \usepackage{dingbat}
3 \NewTasks[label=\footnotesize\leftthumbsup,label-width=1.3em]{done}[*]
4 \begin{done}
5   * First task
6   * Second task
7 \end{done}
```

  👍 First task

  👍 Second task

Although this might seem handy or even nice I strongly advice against using something different than a command sequence. Remember that the items will be split at *every* occurrence of the separator. So in order to use the separator (here for example for a starred variant of a command) within an item it has to be hidden in braces. This is avoided of you use a command sequence which even doesn't have to be defined.

Please also keep in mind that the separator still has an optional star argument (see 4), an optional bang argument and the standard optional argument. Using * will prevent the optional star argument.

```
1 % preamble:
2 % \usepackage{dingbat}
3 \NewTasks[label=\footnotesize\leftthumbsup,label-width=1.3em]{done}[*]
4 \begin{done}(3)
5   * First task
6   * Second task
7   *! Third task spanning the full width available
```

```
8    * Fourth task
9  \end{done}
```

    👍 First task   👍 Second task

    👍 Third task spanning the full width available

    👍 Fourth task

Let's say you want a `multiplechoice` environment that has three columns in its default state. You could do something like this:

```
1  % preamble:
2  % \NewTasks[style=multiplechoice]{multiplechoice}[\choice](3)
3  % \newcommand*\correct{\PrintSolutionsTF{\checkedchoicebox}{\choicebox}}
4  %
5  % \PrintSolutionsTF and the {question} environment are provided
6  % by the ExSheets package
7  \begin{question}
8    \begin{multiplechoice}
9      \choice First choice
10     \choice Second choice
11     \choice[\correct] Third choice
12   \end{multiplechoice}
13 \end{question}
14 \begin{solution}[print]
15   \begin{multiplechoice}
16     \choice First choice
17     \choice Second choice
18     \choice[\correct] Third choice
19   \end{multiplechoice}
20 \end{solution}
```

Exercise 2.

    ☐ First choice   ☐ Second choice   ☐ Third choice

Solution 2.

☐ First choice ☐ Second choice ☒ Third choice

The last example shows you two additional commands:

`\choicebox` ☐
  Print an empty square.

`\checkedchoicebox` ☒
  Print a crossed-out square.

8 Styling TASKS

Equivalent to the styling of EXSHEETS TASKS uses xtemplate to declare additional instances for the lists.

8.1 The tasks Object

The object that's defined by TASKS is the 'tasks' object. This time there are four instances available for the one template (again 'default') that was defined.

8.1.1 Available Options

This section only lists the options that can be used when defining an instance of the 'default' template. The following subsections will give some examples of their usage.

```
1  \DeclareTemplateInterface{tasks}{default}{3}
2  {
3    % option        : type       = default
4    enumerate       : boolean    = true    ,
5    label           : tokenlist            ,
6    indent          : length     = 2.5em   ,
7    counter-format  : tokenlist  = tsk[a]) ,
8    label-format    : tokenlist            ,
9    label-width     : length     = 1em     ,
10   label-offset    : length     = .3333em ,
11   after-item-skip : skip       = 1ex plus 1ex minus 1ex
12  }
```

8.1.2 Predefined Instances

This is rather brief this time:

```
1  % ALPHABETIZE: a) b) c)
2  \DeclareInstance{tasks}{alphabetize}{default}{}
3  % available when `load-tasks=true':
4  % ITEMIZE:
5  \DeclareInstance{tasks}{itemize}{default}
6    {
7      enumerate   = false ,
8      label-width = 1.125em
9    }
10 % ENUMERATE:
11 \DeclareInstance{tasks}{enumerate}{default}
12   { counter-format = tsk. }
13 % MULTIPLECHOICE:
14 \DeclareInstance{tasks}{multiplechoice}{default}
15   {
16     enumerate = false       ,
17     label     = \choicebox  ,
18   }
```

References

[L3Pa] The LaTeX3 Project Team. l3kernel. version SVN 6377, Jan. 19, 2016.
        URL: http://mirror.ctan.org/macros/latex/contrib/l3kernel/.

[L3Pb] The LaTeX3 Project Team. l3packages. version SVN 6377, Jan. 19, 2016.
        URL: http://mirror.ctan.org/macros/latex/contrib/l3packages/.

[Nie14] Clemens Niederberger. cntformats. version 0.7, July 3, 2014.
        URL: http://mirror.ctan.org/macros/latex/contrib/cntformats/.

[Pod86] Sunil Podar. epic. version 1.2, July 14, 1986.
        URL: http://mirror.ctan.org/macros/latex/contrib/epic/.

[Rob13] Will Robertson. environ. version 0.3, Apr. 1, 2013.
        URL: http://mirror.ctan.org/macros/latex/contrib/environ/.

Index