THE EXSHEETS BUNDLE

v0.21i   2017/02/08

the packages ExSheets and ExSheets-listings
or
Yet another package for the creation of exercise sheets and exams.

Clemens NIEDERBERGER

http://www.mychemistry.eu/forums/forum/exsheets/

contact@mychemistry.eu

ExSheets provides means to create exercises or questions and their corresponding solutions. The questions can be divided into classes and can be printed selectively. Meta-data to questions can be added and recovered.

The solutions may be printed where they are, can be collected and printed at a later point in the document alltogether or section-wise or selectively by ID.

ExSheets provides a comprehensive interface for styling the headings of questions and solutions.

Table of Contents

1

Part I.
Preliminaries

1. Licence 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."

ExSheets loads and needs the following packages: l3kernel[1] [L3Pb], xparse, xtemplate, l3keys2e[2] [L3Pc], l3sort[3] [L3Pa], xcolor[4] [Ker07], ulem[5] [Ars11], etoolbox[6] [Leh15], environ[7] [Rob13], and pgfcore[8] [Tan13]. ExSheets calls `\normalem` (from the ulem package).

2. Motivation

There are already quite a number of packages that allow the creation of exercise sheets or written exams. Just to name the most common ones: eqexam [Sto11], exam [Hir15], examdesign [Ale01], exercise [Pic12], probsoln [Tal12], answers [Pif10], esami [MV13], exsol [Dae13] (and many more . . .).

One thing I missed in all packages that I've tried out[9] was a high flexibility in choosing which questions and solutions should be printed, where which solutions should be printed and so on,

---

1. on CTAN as l3kernel: `http://mirrors.ctan.org/macros/latex/contrib/l3kernel/`

2. all three on CTAN as l3packages: `http://mirrors.ctan.org/macros/latex/contrib/l3packages/`

3. on CTAN as l3experimental: `http://mirrors.ctan.org/macros/latex/contrib/l3experimental/`

4. on CTAN as xcolor: `http://mirrors.ctan.org/macros/latex/contrib/xcolor/`

5. on CTAN as ulem: `http://mirrors.ctan.org/macros/latex/contrib/ulem/`

6. on CTAN as etoolbox: `http://mirrors.ctan.org/macros/latex/contrib/etoolbox/`

7. on CTAN as environ: `http://mirrors.ctan.org/macros/latex/contrib/environ/`

8. on CTAN as pgf: `http://mirrors.ctan.org/graphics/pgf/`

9. Well, probably I didn't try hard enough…

combined with the possibility to assign questions to different classes so one could for example create two versions of an exam out of the box. And – I can't get enough – I also want to be able to use/design different layouts for questions additional to a standard section-like format. All these points are realized in ExSheets.

Additionally one should be able to assign some sort of meta-data to questions that of course should be easily reusable. How this can be done is explained in section 11.

Then there is – at least in Germany – the habit of having lists of exercises aligned in columns but counting from the left to the right instead from up to down. That's why the tasks package was developed as part of ExSheets and was distributed as part of the bundle. Now it is a package of its own but is loaded by ExSheets automatically with the necessary setup to make them work together nicely.

Changed in version 0.15

ExSheets has no native support for multiple choice tests but that doesn't mean that you can't create them with ExSheets. It just means that they may be a bit more work with ExSheets than with other packages.

I had the idea for this package in 2008. Back then my TeX skills were by far not good enough to write it. Actually, even today I wouldn't have been able to realize it without all the l3 packages like l3kernel and l3packages. I actively began to develop ExSheets in spring 2011 but it wasn't until now (September 2012) that I consider it stable enough for wider usage. At the time of writing (February 9, 2017) there still are probably lots of rough edges let alone bugs so I am very interested in all kinds of feedback.

3. Additional Packages

ExSheets actually bundles two packages: ExSheets, ExSheets-listings. ExSheets-listings is an add-on to ExSheets that offers some functionality to use listings with ExSheets. It is presented in part III.

Changed in version 0.9i

Changed in version 0.15

ExSheets used to bundle the translations package, too, but doesn't any more. You can find the translations package as a package of it's own on the Comprehensive TeX Archive Network (CTAN). It also used to bundle the packages tasks and cntformats. They're available now as packages of their own as well.

4. Thanks

I need to thank the many users who gave me feedback so far! For one thing this shows me that ExSheets is useful to people. It also led to many improvements like new features and countless bug fixes.

Part II.
The ExSheets package

5. Setup

The ExSheets package has three different types of options, kind of. The first type are the classic package options which are used when you load ExSheets:

```
1 \usepackage[<options>]{exsheets}
```

All general options can be used this way and most of them are described in section 6. All of those options also can be set via the setup command:

\SetupExSheets[⟨module⟩]{⟨options⟩}

The second type are options that belong to a specific environment or command. These options are either used directly with the environment/command

```
1 \begin{env}[<options>]
2 ...
3 \end{env}
```

or can also be set with the setup command. In the first case they only act upon the environment or command where they're used. In the second case they are set for all following uses of the corresponding environment or command.

The options of the second type all belong to `modules`. Let's say you want to specify some options of the `question` environment. You can then say the following:

```
1 \SetupExSheets[question]{option1,option2=value2}
2 % or:
3 \SetupExSheets{question/option1,question/option2=value2}
```

The `module` an option belongs to is written in the left margin next to the when the option is described.

The third type aren't options at all, actually. However, thanks to the great xtemplate package you are able to define your own instances of some of the objects used by EXSHEETS. This is explained in a little more detail in part 18 on page 39 ff. This third type, however, brings in a possible instability: the xtemplate package is in an experimental and developing state. This means that the sytax of the package may and possibly will change sometime in the future. I cannot foresee what any consequences of that will be for EXSHEETS.

6. General Options

The package EXSHEETS has some options, namely the following ones:

counter-format = {⟨*counter-format*⟩}   Default: qu.
  Formatting of the counter of the questions. This option takes a special kind of string that is described in section 7.6.

counter-within = {⟨*counter*⟩}   (initially empty)
  Resets the question counter with every step of ⟨*counter*⟩.

auto-label = <u>true</u>|false   Default: false
  If set to true EXSHEETS will automatically place a \label{qu:⟨*id*⟩} for each question. See section 15 for ways to customize this. It will also create the question properties ref and pageref, see section 11 for more on this.

headings = {⟨*instance*⟩}   Default: block
  Choose the style of the questions' and solutions' headings. There are two predefined styles: block and runin.

headings-format = {⟨*code*⟩}   Default: \normalsize\bfseries
  This code is placed immediately before the headings of the questions and solutions.

subtitle-format = {⟨*code*⟩}   Default: \normalsize\itshape
  This code is placed immediately before the subtitle of the questions and solutions. It only has an effect with a title instance that uses the subtitle coffin, see section 18.2.

skip-below = {⟨*dim*⟩}   Default: .5\baselineskip
  Sets the vertical space that is inserted after the question and solution environments.

no-skip-below = <u>true</u>|false   Default: false
  Disables the insertion of vertical space after the question and solution environments.

totoc = <u>true</u>|false   Default: false
  This option adds the questions and solutions with their names and numbers to the table of contents.

questions-totoc = <u>true</u>|false   Default: false
  This option adds the questions with their names and numbers to the table of contents.

`solutions-totoc` = <u>`true`</u>`|false`   Default: `false`
   This option adds the solutions with their names and numbers to the table of contents.

`toc-level` = {⟨*toc level*⟩}   Default: `subsection`
   This option sets the level in which questions and solutions should appear in the table of contents.

`questions-toc-level` = {⟨*toc level*⟩}   Default: `subsection`
   This option sets the level in which questions should appear in the table of contents.

`solutions-toc-level` = {⟨*toc level*⟩}   Default: `subsection`
   This option sets the level in which solutions should appear in the table of contents.

`use-ref` = <u>`true`</u>`|false`   Default: `false`
   enable referencing to sections and chapters in a way that the references can be used with
   `\printsolutions`, see section 9.2 for details.

   The `toc` options are demonstrated with section A and the solutions printed there being listed in
the table of contents.

7. Create Questions/Exercises and their Solutions

Now, let's start with the most important part: the questions and (possibly) their respective
solutions.

7.1. The question Environment

Questions are written inside the `question` environment:

`\begin{question}[`⟨*options*⟩`]{`⟨*points*⟩`}`
   The main environment: creates a new exercise/question. Both arguments are optional!

```
1 \begin{question}
2   This is our very first very difficult to solve question!
3 \end{question}
```

Exercise 1.
This is our very first very difficult to solve question!

As you can see a heading is automatically created and the question is numbered. You can of
course change both the numbering and the naming, but more on that later.
   The `question` environment takes an optional argument {⟨*points*⟩} that can be used to assign
points to the question (as is common in written exams):

```
1 \begin{question}{3}
2   This is our first difficult question that is worth 3 points!
3 \end{question}
```

Exercise 2.   3 P.

This is our first difficult question that is worth 3 points!

These points are saved internally (see section 8 for reasons why) and are written to the right margin next to the question heading in the default setting.

You can also assign bonus points by inserting ⟨*point*⟩+⟨*bonus points*⟩ as argument.

```
1 \begin{question}{1+1}
2   This question is worth 1 point and 1 bonus point.
3 \end{question}
4 \begin{question}{+3}
5   This question is a bonus question. It is worth 3 bonus points.
6 \end{question}
```

Exercise 3.   1 (+1) P.

This question is worth 1 point and 1 bonus point.

Exercise 4.   (+3 P.)

This question is a bonus question. It is worth 3 bonus points.

The points are counted and added to the total sum of points, see section 8 for details on this.

Introduced in version 0.12

Should you want that the points of a specific question *should not be added* to the total sum then precede it with a bang !:

```
1 \begin{question}{!3}
2   This question's points won't be added to the total sum.
3 \end{question}
```

Exercise 5.   3 P.

This question's points won't be added to the total sum.

Beware that this also prevents bonus points. The points simply will be written where the heading instance puts them.

One additional thing: you might want to define custom commands that should behave differently if they're inside or outside of the question environment. In this case you can use these commands:

* `\IfInsideQuestionTF{`⟨*true code*⟩`}{`⟨*false code*⟩`}`
  Check if inside of a question and either leave ⟨*true code*⟩ or ⟨*false code*⟩ in the input stream.

* `\IfInsideQuestionT{`⟨*true code*⟩`}`
  Check if inside of a question and either leave ⟨*true code*⟩ in the input stream if true.

* `\IfInsideQuestionF{`⟨*false code*⟩`}`
  Check if inside of a question and either leave ⟨*false code*⟩ in the input stream if not.

7.2. Options to the question Environment

The question environment takes one or more of the following options:

question » `type` = exam|exercise   Default: `exercise`
Determines the type of question and changes the default name of a question from "Exercise" to "Question". These default names are language dependent.

If you use `\usepackage[`⟨*ngerman*⟩`]{`⟨*babel*⟩`}`, for example, then the names are "ÃIJbung" and "Aufgabe".

question » `name` = {⟨*name*⟩}   (initially empty)
Sets a custom name. All predefined names are discarded.

question » `subtitle` = {⟨*subtitle*⟩}   (initially empty)
Adds a subtitle ⟨*subtitle*⟩ for the question that is used by headings instances that make use of the subtitle coffin, see section 18.2.

question » `skip-below` = {⟨*dim*⟩}   Default: `.5\baselineskip`
Sets the vertical space that is inserted after the question environment.

question » `no-skip-below` = true|false   Default: `false`
Disables the insertion of vertical space after the question environment.

question » `print` = true|false   Default: `true`
Prints or hides the question.

question » `ID` = {⟨*id*⟩}   (initially empty)
Assigns a custom ID to the question. See section 9.3 for further information.

question » `label` = {⟨*label*⟩}   (initially empty)
Places a `\label{`⟨*label*⟩`}` for the question. This will overwrite any label that is placed by the `auto-label` option.

9

question » class = {⟨*class*⟩}   (initially empty)
        Assigns a class ⟨*class*⟩ to the question. See section 10.1 for further information.

question » topic = {⟨*topic*⟩}   (initially empty)
        Assigns a topic ⟨*topic*⟩ to the question. See section 10.2 for further information.

question » use = <u>true</u>|false   Default: true
        Discards the question. Or not.

question » pre-hook = {⟨*code*⟩}   (initially empty)
Changed in   Adds ⟨*code*⟩ directly before the question title.
version 0.16
question » post-hook = {⟨*code*⟩}   (initially empty)
Changed in   Adds ⟨*code*⟩ directly after the question.
version 0.16
question » pre-body-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly before the question body.
version 0.16
question » post-body-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly after the question body.
version 0.16

```
1  \begin{question}[type=exam]
2    This question has the type \keyis{type}{exam}. The default name has changed
3    from ``Exercise'' to ``Question''.
4  \end{question}
5  \begin{question}[name=Fancy name]
6    This question has a custom name.
7  \end{question}
8  \begin{question}[print=false]
9    This question is not printed.
10 \end{question}
```

Question 6.
This question has the type type = {exam}. The default name has changed from "Exercise"
to "Question".

Fancy name 7.
This question has a custom name.

The difference between print and use lies behind the scenes: with print = {false} the
question is not printed, but it still gets an individual ID, is numbered, and a possible solution
is saved. This is for example useful when you want to print a sample solution for an exam.
With use = {false} it is fully discarded which means it is not accessible through an ID and a
possible solution will not be saved.

7.3. Subtitles to Questions

The subtitle option mentioned in section 7.2 can be used to add a subtitle to a question. However, unless you choose a suitable heading (see section 18.2) it won't be printed. Currently there is *one* heading instance that uses the subtitles but it should be easy to create a custom heading using one of the existing ones as a starter example. When creating such a heading you may want to distinguish between the cases when a subtitle has been given and when no subtitle is present. This can be done with the following commands:

* \IfQuestionSubtitleTF{⟨*true code*⟩}{⟨*false code*⟩}

Tests if the current question has a subtitle. Leaves either ⟨*true code*⟩ or ⟨*false code*⟩ in the input stream.

* \IfQuestionSubtitleT{⟨*true code*⟩}

Tests if the current question has a subtitle. Leaves ⟨*true code*⟩ in the input stream if it has.

* \IfQuestionSubtitleF{⟨*false code*⟩}

Tests if the current question has a subtitle. Leaves ⟨*false code*⟩ in the input stream if it hasn't.

A subtitle is also a property of a question in the sense of section 11. That means if a subtitle is given it can be retrieved with \GetQuestionProperty.

As an example you could define your own heading instance that prints the ID of a question and (if given) the subtitle:

```
1  \DeclareInstance{exsheets-heading}{QE}{default}{
2    join = {
3      title[r,B]number[l,B](.333em,0pt) ;
4      title[r,B]subtitle[l,B](1em,0pt)
5    } ,
6    attach = {
7      main[l,vc]title[l,vc](0pt,0pt) ;
8      main[r,vc]points[l,vc](\marginparsep,0pt)
9    } ,
10   subtitle-post-code = {ID: \CurrentQuestionID} ,
11   number-post-code   = {\IfQuestionSubtitleF{ID: \CurrentQuestionID}}
12 }
```

Please see section 18.2 for more details on heading instances.

7.4. The solution Environment

If you want to save/print (more on the exact usage in section 9) a solution you have to use the solution environment *after* the question it belongs to and *before* the next question.

\begin{solution}[⟨*options*⟩]

The main environment for adding solutions to exercises/questions.

11

```
1 \begin{question}[ID=first]\label{qu:question_with_solution}
2   This is our first question that gets a solution!
3 \end{question}
4 \begin{solution}
5   This is the solution to exercise~\ref{qu:question_with_solution}!
6 \end{solution}
```

Exercise 9.

This is our first question that gets a solution!

You can see that in the default settings the solution is *not* written to the document. It has been saved, though, for possible later usage. We will see the solution later!

7.5. Options to the solution Environment

The `solutions` environment also has options, namely these:

solution » name = {⟨*name*⟩}   (initially empty)
>   Sets a custom name.

solution » print = <u>true</u>|false   Default: `false`
>   Prints or hides the solution.

solution » skip-below = {⟨*dim*⟩}   Default: .5\baselineskip
Introduced in   Sets the vertical space that is inserted after the solution environment.
version 0.18
solution » no-skip-below = <u>true</u>|false   Default: `false`
Introduced in   Disables the insertion of vertical space after the solution environment.
version 0.18
solution » pre-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly before the solution title.
version 0.16
solution » post-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly after the solution.
version 0.16
solution » pre-body-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly before the solution body.
version 0.16
solution » post-body-hook = {⟨*code*⟩}   (initially empty)
Introduced in   Adds ⟨*code*⟩ directly after the solution body.
version 0.16

Their meaning is the same as those for the question environment.

```
1  \begin{question}{5}
2    The solution to this questions gets printed where it is.
3  \end{question}
4  \begin{solution}[print]
5    See? This solution gets printed where you have put it in the code of
6    your document.
7  \end{solution}
8  \begin{question}{2.5}
9    The solution to this questions gets printed where it is \emph{and}
10   has a fancy name. Have you noticed that you can assign partial
11   points?
12 \end{question}
13 \begin{solution}[print,name=Fancy name]
14   See? This solution gets printed where you have put it and has a fancy
15   name!
16 \end{solution}
```

Exercise 10.   5 P.

The solution to this questions gets printed where it is.

Solution 10.

See? This solution gets printed where you have put it in the code of your document.

Exercise 11.   2.5 P.

The solution to this questions gets printed where it is *and* has a fancy name. Have you noticed that you can assign partial points?

Fancy name 11.

See? This solution gets printed where you have put it and has a fancy name!

7.6. Setting the Counter

The package option `counter-format` allows you to specify how the question counter (a counter unsurprisingly name question) is formatted.

The input is an arbitrary string which means you can have anything as counter number. However, the letter combinations ch, se, qu and tsk are replaced with the counters for the chapter, section, question or tasks (see the TASKS package), respectively. While the last one is not really useful in this case the others allow for a combined numbering. Each of these letter combinations can have an optional argument that specifies the format of the respective counter. 1: \arabic, a: \alph, A: \Alph, r: \roman and R: \Roman.

```
1 \SetupExSheets{counter-format=Nr~se~(qu[a])}
2 \begin{question}
3   A question with a differently formatted number.
4 \end{question}
```

Exercise Nr 7 (l)
A question with a differently formatted number.

Since the strings associated with the counters are replaced one has to hide them if they are actually wanted in the counter format. The easiest way would to hide them in braces.

```
1 \SetupExSheets{counter-format={section}\,se~{question}\,(qu[a])}
2 \begin{question}
3   A question with a yet differently formatted number.
4 \end{question}
```

Exercise section 7 question (m)
A question with a yet differently formatted number.

7.7. Language Settings

The names of the questions and solutions are language dependent. If you use babel or polyglossia EXSHEETS will adapt to the document language. EXSHEETS has a number of translations but surely not all! If you miss a language please drop me a line in an email [10] containing the babel language name and the correct translations for questions (possibly distinguishing between exercises and exam questions) and solutions.

Until I implement it you can add something like this to your preamble (example for Danish) and try if it works:

```
1 \DeclareTranslation{Danish}{exsheets-exercise-name}{\O{}velse}
2 \DeclareTranslation{Danish}{exsheets-question-name}{Opgave}
3 \DeclareTranslation{Danish}{exsheets-solution-name}{Opl\o{}sning}
```

If this isn't working it means that the language you're using is unknown to the translations

---

10. contact@mychemistry.eu

package. In this case please notify me, too. You then can still use the name options.

8. Counting Points

8.1. The Commands

You have seen in section 7.1 that you can assign points to a question. If you do so these points are printed into the margin[11] and are counted internally. But there are additional commands to assign points or bonus points and a number of commands to retrieve the sum of points and/or bonus points.

`\addpoints*{⟨num⟩}`
This command can be used to add points assigned to subquestions. `\addpoints` will print the points (with "unit") *and* add them to the sum of all points, `\addpoints*` will only add them but print nothing.

`\points*{⟨num⟩}`
This command will only print the points (with "unit") but won't add them to the sum of points.

`\addbonus*{⟨num⟩}`
This command can be used to add bonus points assigned to subquestions. `\addbonus` will print the points (with "unit") *and* add them to the sum of all bonus points, `\addbonus*` will only add them but print nothing.

`\bonus*{⟨num⟩}`
This command will only print the bonus points (with "unit") but won't add them to the sum of bonus points.

`\pointssum*`
Prints the sum of all points with or without (starred version) "unit": 64.75 P.

`\currentpointssum*`
Prints the current sum of points with or without (starred version) "unit": 11.5 P.

`\bonussum*`
Prints the sum of all bonus points with or without (starred version) "unit": 4 P.

`\currentbonussum*`
Prints the current sum of bonus points with or without (starred version) "unit": 4 P.

`\totalpoints*`
prints the sum of the points *and* the sum of the bonus points with "unit": 64.75 (+4) P. The starred version prints the sum of the points without "unit": 64.75 (+4).

---

11. Well, not necessarily. It depends on the heading style you have chosen.

The commands \pointssum, \bonussum and \totalpoints need at least *two* LATEX runs to get the sum right.

Suppose you have an exercise worth 4 P. which consists of four questions listed with an enumerate environment that are all worth 1 P. each. You have two possibilities to display and count them:

```
1 % uses package `enumitem'
2 \begin{question}{4}
3   \begin{enumerate}[label=\alph*)]
4     \item blah (\points{1})
5     \item blah (\points{1})
6     \item blah (\points{1})
7     \item blah (\points{1})
8   \end{enumerate}
9 \end{question}
10 \begin{question}
11   \begin{enumerate}[label=\alph*)]
12     \item blah (\addpoints{1})
13     \item blah (\addpoints{1})
14     \item blah (\addpoints{1})
15     \item blah (\addpoints{1})
16   \end{enumerate}
17 \end{question}
```

Exercise 14.   4 P.

    a) blah (1 P.)

    b) blah (1 P.)

    c) blah (1 P.)

    d) blah (1 P.)

Exercise 15.

    a) blah (1 P.)

    b) blah (1 P.)

    c) blah (1 P.)

    d) blah (1 P.)

8.2. **Options**

points » name = {⟨*name*⟩}   Default: P.
>   Choose the "unit" for the points. If you like to differentiate between a single point and more
>   than one point you can give a plural ending separated with a slash: name = {point/s}. This
>   sets also the name of the bonus points.

points » name-plural = {⟨*plural form of name*⟩}   (initially empty)
>   Instead of forming the plural form with an ending to the singular form this option allows to set
>   an extra word for it. This sets also the plural form for the bonus points.

points » bonus-name = {⟨*name*⟩}   Default: P.
>   Choose the "unit" for the bonus points. If you like to differentiate between a single point
>   and more than one point you can give a plural ending separated with a slash: bonus-name =
>   {⟨*point/s*⟩}.

points » bonus-plural = {⟨*plural form of name*⟩}   (initially empty)
>   Instead of forming the plural form with an ending to the singular form this option allows to set
>   an extra word for it.

points » use-name = <u>true</u>|false   Default: true
>   Don't display the name at all. Or do.

points » format = {⟨*code*⟩}   Default: \@firtsofone

Introduced in
version 0.9d
>   Format number plus name as a whole. Ideally ⟨*code*⟩ would end with a command that takes an
>   argument. Else number plus name will be braced.

points » number-format = {⟨*any code*⟩}   (initially empty)
>   This option allows formatting of the number, *e. g.*, italics: number-format = {\textit}.

points » bonus-format = {⟨*any code*⟩}   (initially empty)
>   This option allows formatting of the number of the bonus points, *e. g.*, italics: bonus-format =
>   {\textit}.

points » parse = <u>true</u>|false   Default: true
>   If set to false the points are not counted and the \totalpoints, \pointssum and \bonussum
>   commands won't know their value.

points » separate-bonus = <u>true</u>|false   Default: false
>   This option determines whether points and bonus points each get their own unit when they
>   appear together (in the margin or with \totalpoints).

points » pre-bonus = {⟨*tokens*⟩}   Default: \space(+
>   Code to be inserted before the bonus points when they follow normal points.

points » post-bonus = {⟨*tokens*⟩}   Default:)
>   Code to be inserted after the bonus points when they follow normal points.

```
1 \SetupExSheets[points]{name=point/s,number-format=\color{red}}
2 \begin{question}{1}
3   This one's easy so only 1 point can be earned.
4 \end{question}
5 \begin{question}{7.5}
6   But this one's hard! 7.5 points are in there for you!
7 \end{question}
```

Exercise 16.   1 point

This one's easy so only 1 point can be earned.

Exercise 17.   7.5 points

But this one's hard! 7.5 points are in there for you!

9. Printing Solutions

You have already seen that you can print solutions where they are using the `print` option. But ExSheets offers you quite more possibilities.

In the next subsections the usage of the following command is discussed.

`\printsolutions[`⟨*setting*⟩`]`
  Print solutions of questions/exercises.

Before we do that a hint: remember that you can set the option `print` globally:

```
1 % in the preamble
2 \SetupExSheets{solution/print=true}
```

Now if you want to typeset some text depending on the option being true or not you can use the following commands:

* `\PrintSolutionsTF{`⟨*true code*⟩`}{`⟨*false code*⟩`}`
  Either leaves ⟨*true code*⟩ or ⟨*false code*⟩ in the input stream depending on wether solutions are printed or not, *i. e.*, on the value of the solution's option `print`. Inside a `solution` environment this always prints ⟨*true code*⟩.

* `\PrintSolutionsT{`⟨*true code*⟩`}`
  Either leaves ⟨*true code*⟩ or nothing in the input stream depending on wether solutions are printed or not, *i. e.*, on the value of the solution's option `print`. Inside a `solution` environment this always prints ⟨*true code*⟩.

* \PrintSolutionsF{⟨*false code*⟩}

Either leaves nothing or ⟨*false code*⟩ in the input stream depending on wether solutions are printed or not, *i. e.*, on the value of the solution's option `print`. Inside a `solution`environment this always prints nothing.

They might come in handy if you want two versions of an exercise sheet, one with the exercises and one with the solutions, and you want to add different titles to these versions, for instance.

9.1. Print all

The first and easiest usage of \printsolutions is the following:

```
1 \printsolutions
```

There is nothing more to say, really. It prints all solutions you have specified except those belonging to a question with option `use` = {false}. Yes, there's one more point: \printsolutions only knows the solutions that have been set *before* its usage! This is also true for every usage explained in the next sections.

```
1 \printsolutions
```

Solution 9.
This is the solution to exercise 9!

Solution 10.
See? This solution gets printed where you have put it in the code of your document.

Fancy name 11.
See? This solution gets printed where you have put it and has a fancy name!

Two options allow to add code to the list of solutions when used with \printsolutions[all] (which is the same as using it without option):

`chapter-hook` = {⟨*code*⟩}

Introduced in
version 0.13

Adds ⟨*code*⟩ to the list of solutions every time solutions from a new chapter are printed (before the solutions of the corresponding chapter are printed).

`section-hook` = {⟨*code*⟩}

Introduced in
version 0.13

Adds ⟨*code*⟩ to the list of solutions every time solutions from a new section are printed (before the solutions of the corresponding section are printed).

9.2. Print per chapter/section

Current chapter/section

If you are not creating an exercise sheet or an exam but are writing a textbook you maybe want a section at the end of each chapter showing the solution to the exercises presented in that chapter. In this case use the command as follows:

```
1 \printsolutions[section]
2 % or
3 \printsolutions[chapter]
```

Again, this is pretty much self-explaining. The solutions to the questions of the current chapter [12] or section are printed.

```
1 \begin{question}
2   This is the first and only question in this section.
3 \end{question}
4 \begin{solution}
5   This will be one of a few solutions printed by the following call of
6   \cs{printsolutions}.
7 \end{solution}
8 And now:
9 \printsolutions[section]
```

Exercise 18.

This is the first and only question in this section.

  And now:

Solution 18.

This will be one of a few solutions printed by the following call of `\printsolutions`.

Specific chapter/section

You can also print only the solutions from chapters or sections other than the current ones. The syntax is fairly easy:

---

12. Only if the document class you're using *has* chapters, of course!

```
1 \printsolutions[section={1-7,10}]
2 % the same for chapters:
3 % \printsolutions[chapter={1-7,10}]
```

Solution 9.
This is the solution to exercise 9!

Solution 10.
See? This solution gets printed where you have put it in the code of your document.

Fancy name 11.
See? This solution gets printed where you have put it and has a fancy name!

Don't forget that \printsolutions cannot know the solutions from section 10 yet. It is just used to demonstrate the syntax. You can also use an open range, *e. g.*, something like

```
1 \printsolutions[section={-4,10-}]
```

This would print the solutions from sections 1–4 and from all sections with number 10[13] and greater.

There is an obvious disadvantage: you have to know the section numbers! But there is a solution: use the package option use-ref = {true}. Then you can do something like

```
1 % in the preamble:
2 \usepackage[use-ref]{exsheets}
3 % somewhere in your code after \section{A really cool section title}:
4 \label{sec:ReallyCool}
5 % somewhere later in your code:
6 \printsolutions[section={-\S{sec:ReallyCool}}]
7 % which will print all solutions from questions up to and
8 % including the really cool section
```

With the package option use-ref = {true} each usage of \label will create additional labels (one preceded with exse: and another one with exch:) which store the section number and the chapter number, respectively. These are used internally by two commands \S and \C which

---

13. Or rather where \value{section} is 10 or greater – the actual counter formatting is irrelevant.

refer to the section number and the chapter number the label was created in. *These commands are only available as arguments of* \printsolutions.

Since some packages like the well known hyperref for example redefine \label use-ref won't work in together with it. In this case don't use use-ref and set \exlabel{⟨*label*⟩} instead to remember the section/the chapter number. Its usage is just like \label. So the safest way is as follows:

```
1 % in the preamble:
2 \usepackage{exsheets}
3 % somewhere in your code after \section{A really cool section title}:
4 \exlabel{sec:ReallyCool}
5 % somewhere later in your code:
6 \printsolutions[section={-\S{sec:ReallyCool}}]
7 % which will print all solutions from questions up to and
8 % including the really cool section
```

Please be aware that the labels must be processed in a previous LATEX run before \S and \C can pass them on to \printsolutions.

9.3. Print by ID

Now comes the best part: you can also print selected solutions! Every question has an ID. To see which ID a question has you can call the following command:

* \CurrentQuestionID

Introduced in
version 0.4a

Expands to the current question ID.

debug = <u>true</u>|false
Enable or disable visual ExSheets' debugging.

Let's create some more questions and take a look what this command does:

```
1 \SetupExSheets{debug=true}
2 \begin{question}[ID=nice!]
3   A question with a nice \acs{id}!
4 \end{question}
5 \begin{solution}
6   The solution to the question with the nice \acs{id}.
7 \end{solution}
8 \begin{question}{3.75}
9   Yet another question. But this time with quarter points!
10 \end{question}
11 \begin{solution}
```

```
12    Yet another solution.
13  \end{solution}
```

> **ID**: *nice!*; **not tagged**
>
> **Exercise 19.**
>
> A question with a nice ID!
>
> **ID**: *20*; **not tagged**
>
> **Exercise 20.**
>
> Yet another question. But this time with quarter points!

3.75 P.

So now we can call some specific solutions:

```
1  \printsolutions[byID={first,nice!,10,14}]
```

Solution 9.

This is the solution to exercise 9!

Solution 10.

See? This solution gets printed where you have put it in the code of your document.

Solution 19.

The solution to the question with the nice ID.

This makes use of the l3sort package which at the time of writing is still considered experimental. In case you wonder where solution 14 is: question 14 has no solution given.

If you don't want that the solutions are sorted automatically but appear in the order given you can use the option

solution » sorted = <u>true</u>|false   Default: true

   Sort solutions given by ID or don't.

10. Conditional Printing of Questions

10.1. Using Classes

For creating different variants of a written exam or different difficulty levels of an exercise sheet it comes in handy if one can assign certain classes to questions and then tell ExSheets only to use one ore more specific classes.

use-classes = {⟨*list of classes*⟩}   (initially empty)

When this option is used only the questions belonging to the specified classes are printed and have their solutions saved.

```
1 \SetupExSheets{use-classes={A,C}}
2 \begin{question}[class=A]
3   Belonging to class A.
4 \end{question}
5 \begin{question}[class=B]
6   Belonging to class B.
7 \end{question}
8 \begin{question}[class=C]
9   Belonging to class C!
10 \end{question}
```

Exercise 21.
Belonging to class A.

Exercise 22.
Belonging to class C!

Questions of classes that are not used are fully discarded. *This also means that questions that don't have a class assigned are discarded.*

10.2. Using Topics

Similarly to classes one can assign topics to questions. The usage is practically identical, the semantic meaning is different.

use-topics = {⟨*list of topics*⟩}   (initially empty)

When this option is used only the questions belonging to the specified topics are printed and have their solutions saved.

```
1 \SetupExSheets{use-topics={trigonometry}}
2 \begin{question}[topic=trigonometry]
3   A trigonometry question.
4 \end{question}
5 \begin{question}[topic=arithmetics]
6   A arithmetics question
7 \end{question}
```

> **Exercise 23.**
> A trigonometry question.

Questions of topics that are not used are fully discarded. *This also means that questions that don't have a topic assigned are discarded.*

If you set both `use-classes` and `use-topics` then only questions will be used that *match both categories*.

Ideally one could assign more than one topic to a question but this is *not* supported yet.

10.3. **Own Dividing Concepts**

Introduced in version 0.8

Actually both classes and topics are introduced into EXSHEETS internally this way:

```
1 \DeclareQuestionClass{class}{classes}
2 \DeclareQuestionClass{topic}{topics}
```

which means you can do the same introducing your own dividing concepts.

`\DeclareQuestionClass{`⟨*singular name*⟩`}{`⟨*plural name*⟩`}`

Introduces a new dividing concept and defines both new options for the `question` environment and new global options.

For example you could decide you want to group your questions according to their difficulty. You could place the following line in your preamble:

```
1 \DeclareQuestionClass{difficulty}{difficulties}
```

This would define an option `use-difficulties` analogous to `use-classes` and `use-topics`. It would also define an option `difficulty` for the `question` environment. This means you could now do something like the following:

```
1 \SetupExSheets{use-difficulties={easy,hard}}
2 \begin{question}[difficulty=easy]
3   An easy question.
4 \end{question}
5 \begin{question}[difficulty=medium]
```

```
6    This one's a bit harder.
7  \end{question}
8  \begin{question}[difficulty=hard]
9    Now let's see if you can solve this one.
10 \end{question}
```

Exercise 24.
An easy question.

Exercise 25.
Now let's see if you can solve this one.

10.4. Retrieving the Class Value in a Question

Sometimes it may be desirable to retrieve the value of a class defined by \DeclareQuestionClass that a question has in order to be able to print, say. This is possible with the following commands:

* \GetQuestionClass{⟨*class*⟩}
  Prints the value of ⟨*class*⟩ a question has. The command is expandable. If the class does not exist or the value is empty the command expands to nothing.

\PrintQuestionClassTF{⟨*class*⟩}{⟨*true*⟩}{⟨*false*⟩}
  Test if a question has a non-empty value for class ⟨*class*⟩ and either leaves ⟨*true*⟩ or ⟨*false*⟩ in the input stream. In the ⟨*true*⟩ argument you can refer to the value with #1 where you want it printed.

\PrintQuestionClassT{⟨*class*⟩}{⟨*true*⟩}
  Like \PrintQuestionClassTF but only has the ⟨*true*⟩ branch.

\PrintQuestionClassF{⟨*class*⟩}{⟨*false*⟩}
  Like \PrintQuestionClassTF but only has the ⟨*false*⟩ branch.

```
1  \begin{question}[difficulty=hard]
2    This question has the difficulty level
3    ``\PrintQuestionClassTF{difficulty}{#1}{??}''.
4  \end{question}
```

Exercise 26.
This question has the difficulty level "hard".

10.5. Tagging Questions

There is another way of dividing questions: you can assign tags to questions:

```
1 \begin{question}[tags={foo,bar,baz}]
2   ...
3 \end{question}
```

You can then decide to print only questions with certain tags by using the following option:

use-tags = {⟨*csv list of tags to include*⟩}

Select tags. When used only questions being tagged with at least one of the tags in ⟨*csv list of tags to include*⟩ are printed.

11. Adding and Using Additional Information to Questions

11.1. Question Properties – the Basics

For managing lots of questions and corresponding solutions it can be very useful to be able to save and recover additional information to the questions. This is possible with the following commands. First the ones for saving:

\DeclareQuestionProperty{⟨*name*⟩}

This command defines a question property ⟨*name*⟩. It can only be used in the document preamble.

\SetQuestionProperties{⟨*name*⟩=⟨*value*⟩,...}

Set the properties for a specific question. this command can only be used inside the question environment.

Now the commands for recovering the properties:

\QuestionNumber{⟨*id*⟩}

Recover the number of the question with the ID ⟨*id*⟩. The number is displayed according to the format set with counter-format.

* \GetQuestionProperty{⟨*name*⟩}{⟨*id*⟩}

Recover the property ⟨*name*⟩ of the question with the ID ⟨*id*⟩. Of course the property must have been declared before. The command is expandable. Since the properties of a question are written to the aux file it is possible to retrieve them before the corresponding question environment has been used.

* \IfQuestionPropertyTF{⟨*name*⟩}{⟨*id*⟩}{⟨*true*⟩}{⟨*false*⟩}

A command that returns ⟨*true*⟩ if the question with the ID ⟨*id*⟩ has the property ⟨*name*⟩ and ⟨*false*⟩ otherwise. The variants \IfQuestionPropertyT and \IfQuestionPropertyF also exist which only have the ⟨*true*⟩ or the ⟨*false*⟩ branch.

Let's say we have declared the properties `notes`, `reference` and `topic`. By default the property `points` is available and gets the value of the optional argument of the `question` environment.

We can now do the following:

```
% uses `biblatex'
\begin{question}[ID=center,topic=LaTeX]{3}
  Explain how you could center text in a \LaTeX\ document.
  \SetQuestionProperties{
     topic     = \TeX/\LaTeX ,
     notes     = {How to center text.},
     reference = {\textcite{companion}}}
\end{question}
\begin{solution}
  To center a short part of the text body one can use the \env*{center}
  environment (\points{1}). Inside an environment like \env*{table} one
  should use \cs*{centering} (\points{1}). For single lines there is also
  the \cs*{centerline} command (\points{1}).
\end{solution}
\begin{question}[ID=knuthbooks,topic=LaTeX]{2}
  Name two books by D.\,E.\,Knuth.
  \SetQuestionProperties{
     topic     = \TeX/\LaTeX ,
     notes     = {Books by Knuth.},
     reference = {\textcite{knuth:ct:a,knuth:ct:b,knuth:ct:c,knuth:ct:d,knuth:
  ct:e}}}
\end{question}
\begin{solution}
  For example two volumes from \citetitle{knuth:ct}:
  \citetitle{knuth:ct:a,knuth:ct:b,knuth:ct:c,knuth:ct:d,knuth:ct:e}. Each
  valid answer is worth \points{1}
\end{solution}
```

Exercise 27.   3 P.
Explain how you could center text in a LATEX document.

Exercise 28.   2 P.
Name two books by D. E. Knuth.

It is now possible to recover these values later:

```
1  % uses `booktabs'
2  \begin{center}
3    \begin{tabular}{lll}
4      \toprule
5        Question & Property & \\
6      \midrule
7      \QuestionNumber{center}
8        & Points     & \GetQuestionProperty{points}{center} \\
9        & Topic      & \GetQuestionProperty{topic}{center} \\
10       & References & \GetQuestionProperty{reference}{center} \\
11       & Note       & \GetQuestionProperty{notes}{center} \\
12     \midrule
13     \QuestionNumber{knuthbooks}
14       & Points     & \GetQuestionProperty{points}{knuthbooks} \\
15       & Topic      & \GetQuestionProperty{topic}{knuthbooks} \\
16       & References & \GetQuestionProperty{reference}{knuthbooks} \\
17       & Note       & \GetQuestionProperty{notes}{knuthbooks} \\
18     \bottomrule
19   \end{tabular}
20 \end{center}
```

Question	Property	
27.	Points	3
	Topic	TEX/LATEX
	References	Goossens, Mittelbach, and Samarin [GMS94]
	Note	How to center text.
28.	Points	2
	Topic	TEX/LATEX
	References	Knuth [Knu84b; Knu86a; Knu86b; Knu86c; Knu86d]
	Note	Books by Knuth.

Please note that properties *are not the same* as the dividing concepts explained in section 10 although they may seem similar in meaning or even have the same name.

When properties are set they are also written to the aux file which means they can be retrieved *before* the corresponding question. Of course this means that two compilation runs are necessary.

11.2. Pre-defined Properties

A few properties are already defined by ExSheets:

- `counter`: this property holds the actual question number formatted according to the formatting set with option `counter-format`.

- `subtitle`: this property holds the subtitle of the question if given.

- `question-body`: this property holds the body of the corresponding `question` environment. Unlike the other properties it is per default *not* written to the aux file.

- `points`: this property holds the sum of points given to a question.

- `bonus-points`: this property holds the sum of bonus points given to a question.

- `ref`: when the option `auto-label` is used this property is defined and expands to the corresponding `\ref`. Also see section 15.

- `page-ref`: when the option `auto-label` is used this property is defined and expands to the corresponding `\pageref`. Also see section 15.

There is one option affecting the property `question-body`:

`question` » `save-to-aux` = <u>true</u>|false   Default: `false`

When set to `true` the property `question-body` is also written to the aux file.

11.3. Advanced Usage

There are additional commands that might prove useful. They allow advanced usage of defined properties. Below an example is shown how they can be used to generate a grading table.

`\ForEachQuestion{`⟨*code to be executed for each used question*⟩`}`

Inside the argument one can refer to the ID of a question with #1. You can also refer to the number of the question with #2. *Number* means that if you *use* seven questions then those questions have numbers 1 to 7.

* `\numberofquestions`

returns the complete number of used questions.

* `\iflastquestion{`⟨*true code*⟩`}{`⟨*false code*⟩`}`

Although this command is available in the whole document it is only useful inside `\ForEachQuestion`. It tells you if the end of the loop is reached or not.

One could use these commands to create a grading table, for instance:

```
1 \begin{tabular}{|l|*{\numberofquestions}{c|}c|}\hline
2   Question &
3     \ForEachQuestion{\QuestionNumber{#1}\iflastquestion{}{&}} &
4     Total \\ \hline
5   Points   &
6     \ForEachQuestion{\GetQuestionProperty{points}{#1}\iflastquestion{}{&}} &
7     \pointssum* \\ \hline
```

```
 8   Reached  &
 9     \ForEachQuestion{\iflastquestion{}{&}} & \\ \hline
10 \end{tabular}
```

For four questions the table now would look similar to figure 1.

Question	1.	2.	3.	4.	Total
Points	3	5	10	8	26
Reached					

FIGURE 1: An example for a grading table. (Actually this is a fake. See the `grading-table.tex` file shipped with exsheets for the real use case.)

12. Variations of an Exam

It is a quite common task to design an exam in two different variants. This is of course possible with EXSHEETS' classes (see section 10.1). However, often not the whole question is to be different but only small details, the numbers in a maths exam, say. For this purpose EXSHEETS provides the following commands:

`\SetVariations{⟨num⟩}`
Set the number of different variants. This will determine how many arguments the command `\vary` will get. ⟨num⟩ must at least be 2 and is initially set to 2.

`\variant{⟨num⟩}`
Choose the active variant. The argument must be a number between 1 and the number set with `\SetVariations`. Initially set to 1.

`\vary{⟨variant 1⟩}{⟨variant 2⟩}`
This command is the one actually used in the document. It has a number of required arguments equal to the number set with `\SetVariations`. All of its arguments are discarded except the one specified with `\variant`.

`\lastvariant`

Each time `\vary` is called it stores the value it chose in `\lastversion`. This might be convenient to use if one otherwise would have to repeatedly write the same `\vary`.

```
1 \SetVariations{6}%
2 \variant{6}\vary{A}{B}{C}{D}{E}{F}
3 (last variant: \lastvariant)
4 \variant{1}\vary{A}{B}{C}{D}{E}{F}
5 (last variant: \lastvariant)
```

```
6 \variant{5}\vary{A}{B}{C}{D}{E}{F}
7 (last variant: \lastvariant)
8 \variant{2}\vary{A}{B}{C}{D}{E}{F}
9 (last variant: \lastvariant)
10 \variant{4}\vary{A}{B}{C}{D}{E}{F}
11 (last variant: \lastvariant)
12 \variant{3}\vary{A}{B}{C}{D}{E}{F}
13 (last variant: \lastvariant)
```

F (last variant: F) A (last variant: A) E (last variant: E) B (last variant: B) D (last variant: D) C (last variant: C)

13. A Grade Distribution

Probably this is a rather esoteric feature but it could proof useful in some cases. Suppose you are a German math teacher and want to grade exactly corresponding to the number of points relative to the sum of total points, regardless of how big that might be. You could do something like this to present your grading decisions for the exam:

```
1 % preamble:
2 % \DeclareRelGrades{
3 %    1      = 1 ,
4 %    {1,5} = .9167 ,
5 %    2      = .8333 ,
6 %    {2,5} = .75 ,
7 %    3      = .6667 ,
8 %    {3,5} = .5833 ,
9 %    4      = .5
10 % }
11 \small\setlength\tabcolsep{2pt}
12 \begin{tabular}{r|*8c}
13   Punkte
14   & $\grade*{1}$      & $\le\grade*{1}$ & $\le\grade*{1,5}$ & $\le\grade*{2}$
15   & $\le\grade*{2,5}$ & $\le\grade*{3}$ & $\le\grade*{3,5}$ & $<\grade*{4}$ \\
16   Note
17   & 1 & 1--2 & 2 & 2--3 & 3 & 3--4 & 4 & 5
18 \end{tabular}
```

Punkte	64.75	≤ 64.75	≤ 59	≤ 54	≤ 49	≤ 43	≤ 38	< 32
Note	1	1–2	2	2–3	3	3–4	4	5

These are the available commands and options:

`\DeclareRelGrades{⟨grade⟩=⟨num⟩,...}`
This command is used to define grades and assign the percentage of total points to them.

`\grade*{⟨grade⟩}`
Gives the number of points corresponding to a grade depending on the value of `\pointssum` with or without (starred version) "unit".

grades » round = {⟨num⟩}   Default: 0
The number of decimals the points of a grade are rounded to. This doesn't apply to the maximum number of points if the rounded number would be bigger than the actual sum.

grades » half = true|false   Default: false
If set to true points are rounded either to full or to half points.

14. Selectively Include Questions from External Files

14.1. Caveat

I need to say some words of caution: the `\includequestions` that will be presented shortly is probably ExSheets' most experimental one at the time of writing (February 9, 2017). Thanks to feedback of users it is constantly improved and bugs are fixed. It is not a very efficient way to insert question regarding performance and you shouldn't wonder if compilation slows down when you use it. It probably needs to be re-written all over but on the one hand that would introduce new bugs and on the other hand for the time being I don't have the capacities, anyway, so you'll have to live it, I'm afraid.

14.2. How it works

Suppose you have one or more files with questions prepared to use them as a kind of database. One for class A, say, one for class B, one for class C and so one, something like this:

```
1 % this is file classA.tex
2 \begin{question}[class=A,ID=A1,topic=X]
3   First question of class A, topic X.
4 \end{question}
5 \begin{solution}
6   First solution of class A.
7 \end{solution}
8 \begin{question}[class=A,ID=A2,topic=Y]
9   Second question of class A, topic Y.
10 \end{question}
11 \begin{solution}
12   Second solution of class A.
13 \end{solution}
14 ...
```

```
15 % end of file classA.tex
16 \endinput
```

You can of course just \input or \include it but that would of course include the whole file into your document. But would't it be nice to just include selected questions? Or maybe a five random questions from the file? That is possible with the following command:

\includequestions[⟨*options*⟩]{⟨*list of filenames*⟩}
  Include questions from external files.

If you use it without options it will have the same effect as \input. There are however the following options:

include » all = <u>true</u>|false

include » IDs = {⟨*list of IDs*⟩}   (initially empty)
  Includes only the specified questions.

include » random = {⟨*num*⟩}   (initially empty)
  Includes ⟨*num*⟩ randomly selected questions. This option uses the pgfcore package to create the pseudo-random numbers.

include » exclude = {⟨*list of IDs*⟩}   (initially empty)
  Questions who's IDs are specified here are *not* included. This option can be combined with the random option.

The usage should be self-explainable:

```
1 % include questions A1, A3 and A4:
2 \includequestions[IDs={A1,A3,A4}]{classA.tex}
3 % or include 3 random questions:
4 \includequestions[random=3]{classA}
```

In order to be able to select the questions ExSheets needs to \input the file twice. The first time the available questions are determined, the second time the selected questions are used. This unfortunately means that anything that is *not* part of a question or solution is also input twice. Either don't put anything else into the file or use one of the following commands for control:

\PrintIfIncludeActiveTF{⟨*true code*⟩}{⟨*false code*⟩}
  Checks if the questions are actively included or not and puts ⟨*true code*⟩ or ⟨*false code*⟩ in the input stream depending on the answer.

`\PrintIfIncludeActiveT`{⟨*true code*⟩}

Checks if the questions are actively included or not and puts ⟨*true code*⟩ in the input stream if the answer is yes.

`\PrintIfIncludeActiveF`{⟨*false code*⟩}

Checks if the questions are actively included or not and puts ⟨*false code*⟩ in the input stream if the answer is no.

The selection can be refined further by selecting questions belonging to a specific class of questions (see section 10) before using `\includequestions`.

Introduced in version 0.8

After you've used `\includequestions` the IDs of the included questions is available as an unordered comma separated list in the following macro:

`\questionsincludedlast`

Unordered comma separated list of question IDs included with the last usage of `\includequestions`.

15. The `auto-label` Option

Introduced in version 0.12

The package option `auto-label` sets a `\label`{qu:⟨*id*⟩} every time the question environment is used. Both the used command and the automated label can be customized using the following options:

`label-format` = {⟨*code*⟩}   Default: qu:#1

The pattern for generating the automatic label. #1 gets replaced by the ID of the corresponding question.

`label-cmd` = {⟨*macro*⟩}   Default: `\label`

The command used for generating the label. A command that should take one mandatory argument.

`ref-cmd` = {⟨*macro*⟩}   Default: `\ref`

The command used in the `ref` property created by the `auto-label` option, also see section 11. The command should take one mandatory argument.

`pageref-cmd` = {⟨*macro*⟩}   Default: `\pageref`

The command used in the `pageref` property created by the `auto-label` option, also see section 11. The command should take one mandatory argument.

16. Own Question/Solution Pairs

Changed in version 0.9

EXSHEETS provides the possibility to create new environments that behave like the question and solution environments. This would allow, for example, to define a question∗/solution∗ environment pair for bonus questions. The following commands may be used in the document preamble:

\NewQuSolPair{⟨*question*⟩}[⟨*question options*⟩][⟨*general options*⟩]{⟨*solution*⟩}[⟨*solution options*⟩][⟨*general options*⟩]

Define a new pair of question and solution environments.

\RenewQuSolPair{⟨*question*⟩}[⟨*question options*⟩][⟨*general options*⟩]{⟨*solution*⟩}[⟨*solution options*⟩][⟨*general options*⟩]

Redefine an existing pair of question and solution environments.

The standard environments are defined as follows:

```
1 \NewQuSolPair{question}{solution}
```

Let's say we want the possibility to add bonus questions. A simple way would be to define starred variants that add a star in the margin left to the title:

```
1  % preamble:
2  % - \texorpdfstring is provided by `hyperref'
3  % - \bigstar is provided by `amssymb'
4  % \DeclareRobustCommand*\questionstar{\texorpdfstring{\bonusquestionsign}{* }}
5  % \DeclareRobustCommand*\bonusquestionsign{\llap{$\bigstar$\space}}
6  %
7  % \NewQuSolPair
8  %   {question*}[name=\questionstar Bonus Question]
9  %   {solution*}[name=\questionstar Solution]
10 \begin{question*}
11   This is a bonus question.
12 \end{question*}
13 \begin{solution*}[print]
14   This is what the solution looks like.
15 \end{solution*}
```

★ **Bonus Question 29.**
This is a bonus question.

★ **Solution 29.**
This is what the solution looks like.

As you can see the environments take the same options as are described for the standard `question` and `solution` environments.

17. Filling in the Blanks

17.1. Cloze

Both in exercise sheets and in exams it is sometimes desirable to be able to create _____ that have
to be filled in. Or maybe some more lines: _____

_____

_____

_____

_____

_____

`\blank*[⟨`*options*`⟩]{⟨`*text to be filled in*`⟩}`

creates a blank in normal text or in a question but fills the text of its argument if inside a solution.
If used at the *begin of a paragraph* `\blank` will do two things: it will set the linespread according
to an option explained below and will insert `\par` after the lines. If you don't want that use the
starred version.

The options are these:

`blank »` `style` = `line|wave|dline|dotted|dashed` Default: `line`
The style of the line. This uses the corresponding command from the ulem package and is the
whole reason why ExSheets loads it in the first place.

`blank »` `scale` = `{⟨`*num*`⟩}` Default: `1`
Scales the width of the blank by factor ⟨*num*⟩ unless the width is explicitly set.

`blank »` `width` = `{⟨`*dim*`⟩}` (initially empty)
The width of the line. If it is not used the width of the filled in text is used.

`blank »` `linespread` = `{⟨`*num*`⟩}` Default: `1`
Set the linespread for the blank lines. This only has an effect if `\blank` is used at the begin of a
paragraph.

`blank »` `line-increment` = `{⟨`*dim*`⟩}` Default: `1pt`

When the blank line ist built it is built in multiples of this value. If the value is too large you
may end up with uneven lines. If the value is too small you may end up with a non-ending
compilation.

`blank »` `line-minimum-length` = `{⟨`*dim*`⟩}` Default: `2em`

The minimal length a line must have before it is built step by step.

```
1  \begin{question}
2    Try to fill in \blank[width=4cm]{these} blanks. All of them
3    \blank[style=dotted]{are created} by using the \cs{blank}
4    \blank[style=dashed]{command}.
```

```
 5 \end{question}
 6 \begin{solution}[print]
 7   Try to fill in \blank[width=4cm]{these} blanks. All of them
 8   \blank[style=dotted]{are created} by using the \cs{blank}
 9   \blank[style=dashed]{command}.
10 \end{solution}
```

Exercise 30.

Try to fill in _____ blanks. All of them . . . . by using the \blank

_ _ _ _ _ _ _.

Solution 30.

Try to fill in <u>these</u> blanks. All of them are created by using the \blank command.

A number of empty lines are easily created by setting the width option:

```
 1 \blank[width=4.8\linewidth,linespread=1.5]{}
```

_____

_____

_ _

_ _

_____

_____

17.2. Vertical Space for answers

When you're creating an exam you might want to add some vertical space where the students can write down their answers. While you can always use \vspace this is not always handy when the space left on the page is less than you want. In this case it would be nice if a) there would be no warning and b) the rest of the space would be added at the top of the next page. This is what the following command is for:

\examspace*{⟨*dim*⟩}

Add space as specified in ⟨*dim*⟩. If the space available on the current page is not enough the rest of the space will be added at the top of the next page. The starred version will silently drop any leftover space instead of adding it to the next page.

> **Exercise 31.**
> What do you think of this feature?
>
> ```
> 1 \begin{question}
> 2  What do you think of this feature?
> 3  \examspace{3cm}
> 4 \end{question}
> 5 This line comes after the space.
> ```
>
>
> This line comes after the space.

18. Styling your Exercise/Exam Sheets

18.1. Background

The ExSheets package makes extensive use of LaTeX3's coffins[14] as well as its templates concept.[15] The latter allows a rather easy extension and customization of some of ExSheets' environments. To be more precise: you can define your own instances for the headings used for questions and solutions.

What this package doesn't provide is changing the background of questions or framing them. But this is easily possible using the mdframed package and its \surroundwithmdframed command.

ExSheets also provides the options `pre-hook`, `post-hook`, `pre-body-hook` and `post-body-hook` to both the question and the solution environment. With them it is rather straightforward to add a mdframed frame for instance:

```
1 \SetupExSheets{
2   solution/pre-hook = \mdframed ,
3   solution/post-hook = \endmdframed
4 }
```

Introduced in version 0.18

Then there is the macro \exsheetsprintsolution{⟨*heading*⟩}{⟨*body*⟩} which may be redefined to suit your needs. The default definition is equivalent to

---

14. See the documentation to the xcoffins package for more information on that.
15. Have a look into the documentation to the xtemplate package.

```
1 \newcommand\exsheetsprintsolution[2]{#1#2}
```

18.2. The exsheets-headings Object

ExSheets defines the object exsheets-headings and one template for it, the 'default' template. The package also defines two instances of this template, the 'block' instance and the 'runin' instance.

```
1 \SetupExSheets{headings=block}
2 \begin{question}{1}
3   a `block' heading
4 \end{question}
5 \SetupExSheets{headings=runin}
6 \begin{question}{1}
7   a `runin' heading
8 \end{question}
```

Exercise 32.   1 P.

a 'block' heading

Exercise 33. a 'runin' heading   1 P.

18.2.1. 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 loads of examples of their usage. The options are listed in the definition for the template interface:

```
1 \DeclareTemplateInterface{exsheets-heading}{default}{3}{
2   % option            : type      = default
3   inline              : boolean   = false ,
4   runin               : boolean   = false ,
5   indent-first        : boolean   = false ,
6   toc-reversed        : boolean   = false ,
7   vscale              : real      = 1     ,
8   above               : length    = 2pt   ,
9   below               : length    = 2pt   ,
10  main                : tokenlist =       ,
11  pre-code            : tokenlist =       ,
```

```
12    post-code         : tokenlist =         ,
13    title-format      : tokenlist =         ,
14    title-pre-code    : tokenlist =         ,
15    title-post-code   : tokenlist =         ,
16    number-format     : tokenlist =         ,
17    number-pre-code   : tokenlist =         ,
18    number-post-code  : tokenlist =         ,
19    subtitle-format   : tokenlist =         ,
20    subtitle-pre-code : tokenlist =         ,
21    subtitle-post-code : tokenlist =        ,
22    points-format     : tokenlist =         ,
23    points-pre-code   : tokenlist =         ,
24    points-post-code  : tokenlist =         ,
25    join              : tokenlist =         ,
26    attach            : tokenlist =
27 }
```

Each heading is built with at most five coffins available with the names 'main', 'title', 'subtitle', 'number' and 'points'. Those coffins place possibly the whole heading, the title, the subtitle, the question number and the assigned points. The only coffin that's always typeset is the 'main' coffin, which is empty per default.

Coffins can be joined (two become one, the first extends its bounding box to contain the second) using the following syntax:

```
1 join = coffin1[handle11,handle12]coffin2[handle21,handle22](x-offset,y-offset)
```

The syntax for attaching (two become one, the first does *not* extend its bounding box around the second) is the same.

More on coffin handles is described in the documentation for the xcoffins. Figure 2 briefly demonstrates the available handle pairs.
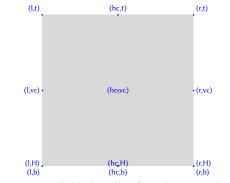

FIGURE 2: Available handles for a horizontal coffin.

It is possible to add own static coffins:

`\DeclareExSheetsHeadingContainer{⟨name⟩}{⟨code⟩}`

Defines a new coffin ⟨*name*⟩ containing ⟨*code*⟩. You can refer to the current question's ID with `\CurrentQuestionID`.

The following subsections will show all definitions of the instances available and how they look. This will hopefully give you enough ideas to create your own instance if you want to have another heading style than the ones available. Each of the following instances is available through the option `headings` = {⟨*instance*⟩}.

The following examples use a sample text defined as follows:

```
1 \def\s{This is some sample text we will use to create a somewhat
2   longer text spanning a few lines.}
3 \def\sample{\s\ \s\par\s}
```

All of the following examples use the same question call:

```
1 \SetupExSheets{headings=<name>}
2 \begin{question}[subtitle=The subtitle of the question]{1}
3   A `<name>' heading. \sample
4 \end{question}
```

18.2.2. The 'block' Instance

```
1 \DeclareInstance{exsheets-heading}{block}{default}{
2   join            = { title[r,B]number[l,B](.333em,0pt) } ,
3   attach          =
4     {
5       main[l,vc]title[l,vc](0pt,0pt) ;
6       main[r,vc]points[l,vc](\marginparsep,0pt)
7     }
8 }
```

Exercise 34. 1 P.

A 'block' heading. 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.

18.2.3. The 'runin' Instance

```
1 \DeclareInstance{exsheets-heading}{runin}{default}{
2   runin           = true ,
3   number-post-code = \space ,
4   attach          =
5     { main[l,vc]points[l,vc](\linewidth+\marginparsep,0pt) } ,
6   join            =
7     {
8       main[r,vc]title[r,vc](0pt,0pt) ;
9       main[r,vc]number[l,vc](.333em,0pt)
10     }
11 }
```

Exercise 35. A 'runin' heading. This is some sample text we will use to create a somewhat 1 P. 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.

18.2.4. The 'simple' Instance

```
1 \DeclareInstance{exsheets-heading}{simple}{default}{
2   title-format     = \normalsize ,
3   points-pre-code  = ( ,
4   points-post-code = ) ,
5   attach           = { main[l,t]number[l,t](0pt,0pt) } ,
6   join             =
7     {
8       number[r,b]title[l,b](.333em,0pt) ;
9       main[l,b]points[l,t](1em,0pt)
10     }
11 }
```

36. Exercise
   (1 P.)

A 'simple' heading. 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.
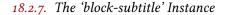
18.2.5. The 'empty' Instance

Introduced in
version 0.9a

```
1 \DeclareInstance{exsheets-heading}{empty}{default}{
2   runin  = true ,
3   above  = \parskip ,
4   below  = \parskip ,
5   attach = { main[l,vc]points[l,vc](\linewidth+\marginparsep,0pt) }
6 }
```

An 'empty' heading. This is some sample text we will use to create a somewhat longer text   1 P.
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.

18.2.6. The 'block-rev' Instance

```
1 \DeclareInstance{exsheets-heading}{block-rev}{default}{
2   toc-reversed    = true ,
3   join            = { number[r,B]title[l,B](.333em,0pt) } ,
4   attach          =
5     {
6       main[l,vc]number[l,vc](0pt,0pt) ;
7       main[r,vc]points[l,vc](\marginparsep,0pt)
8     }
9 }
```

38. Exercise   1 P.

A 'block-rev' heading. 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.

18.2.7. The 'block-subtitle' Instance

```
1 \DeclareInstance{exsheets-heading}{block-subtitle}{default}{
2   join = {
3     title[r,B]number[l,B](.333em,0pt) ;
4     title[r,B]subtitle[l,B](1em,0pt)
5   } ,
6   attach = {
7     main[l,vc]title[l,vc](0pt,0pt) ;
8     main[r,vc]points[l,vc](\marginparsep,0pt)
9   }
10 }
```

Exercise 39. *The subtitle of the question*   1 P.

A 'block-subtitle' heading. 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.

18.2.8. The 'block-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{block-wp}{default}{
2   points-pre-code  = ( ,
3   points-post-code = ) ,
4   join             =
5     {
6       title[r,B]number[l,B](.333em,0pt) ;
7       title[r,B]points[l,B](.333em,0pt)
8     } ,
9   attach           = { main[l,vc]title[l,vc](0pt,0pt) }
10 }
```

Exercise 40. (1 P.)

A 'block-wp' heading. 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.

18.2.9. The 'block-wp-rev' Instance

```
1  \DeclareInstance{exsheets-heading}{block-wp-rev}{default}{
2    toc-reversed     = true ,
3    points-pre-code  = ( ,
4    points-post-code = ) ,
5    join             =
6      {
7        number[r,B]title[l,B](.333em,0pt) ;
8        number[r,B]points[l,B](.333em,0pt)
9      } ,
10   attach           = { main[l,vc]number[l,vc](0pt,0pt) }
11 }
```

41. Exercise (1 P.)

A 'block-wp-rev' heading. 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.

18.2.10. The 'block-nr' Instance

```
1  \DeclareInstance{exsheets-heading}{block-nr}{default}{
2    attach           =
3      {
4        main[l,vc]number[l,vc](0pt,0pt) ;
5        main[r,vc]points[l,vc](\marginparsep,0pt)
6      }
7  }
```

42.   1 P.

A 'block-nr' heading. 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.

18.2.11. The 'block-nr-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{block-nr-wp}{default}{
2   points-pre-code  = ( ,
3   points-post-code = ) ,
4   join             = { number[r,vc]points[l,vc](.333em,0pt) } ,
5   attach           = { main[l,vc]number[l,vc](0pt,0pt) }
6 }
```

43. (1 P.)
A 'block-nr-wp' heading. 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.

18.2.12. The 'runin-rev' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-rev}{default}{
2   toc-reversed    = true ,
3   runin           = true ,
4   title-post-code = \space ,
5   attach          =
6     { main[l,vc]points[l,vc](\linewidth+\marginparsep,0pt) } ,
7   join            =
8     {
9       main[r,vc]number[r,vc](0pt,0pt) ;
10      main[r,vc]title[l,vc](.333em,0pt)
11    }
12 }
```

44. Exercise A 'runin-rev' heading. This is some sample text we will use to create a somewhat   1 P.
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.

18.2.13. The 'runin-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-wp}{default}{
2   runin           = true ,
3   points-pre-code = ( ,
```

```
4   points-post-code = )\space ,
5   join            =
6     {
7       main[r,vc]title[r,vc](0pt,0pt) ;
8       main[r,vc]number[l,vc](.333em,0pt) ;
9       main[r,vc]points[l,vc](.333em,0pt)
10     }
11 }
```

Exercise 45. (1 P.) A 'runin-wp' heading. 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.

18.2.14. The 'runin-wp-rev' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-wp-rev}{default}{
2   toc-reversed     = true ,
3   runin            = true ,
4   points-pre-code  = ( ,
5   points-post-code = )\space ,
6   join             =
7     {
8       main[r,vc]number[r,vc](0pt,0pt) ;
9       main[r,vc]title[l,vc](.333em,0pt) ;
10      main[r,vc]points[l,vc](.333em,0pt)
11     }
12 }
```

46. Exercise (1 P.) A 'runin-wp-rev' heading. 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.

18.2.15. The 'runin-nr' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-nr}{default}{
2   runin            = true ,
3   number-post-code = \space ,
4   attach           =
```

```
5      { main[l,vc]points[l,vc](\linewidth+\marginparsep,0pt) } ,
6   join              = { main[r,vc]number[l,vc](0pt,0pt) }
7 }
```

47. A 'runin-nr' heading. This is some sample text we will use to create a somewhat longer text   1 P.
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.

18.2.16. The 'runin-fixed-nr' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-fixed-nr}{default}{
2   runin             = true ,
3   number-pre-code   = \hbox to 2em \bgroup ,
4   number-post-code  = \hfil\egroup ,
5   attach            =
6      { main[l,vc]points[l,vc](\linewidth+\marginparsep,0pt) } ,
7   join              = { main[r,vc]number[l,vc](0pt,0pt) }
8 }
```

48. A 'runin-fixed-nr' heading. This is some sample text we will use to create a somewhat   1 P.
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.

18.2.17. The 'runin-nr-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{runin-nr-wp}{default}{
2   runin             = true ,
3   points-pre-code   = ( ,
4   points-post-code  = )\space ,
5   join              =
6      {
7        main[r,vc]number[l,vc](0pt,0pt) ;
8        main[r,vc]points[l,vc](.333em,0pt)
9      }
10 }
```

49. (1 P.) A 'runin-nr-wp' heading. 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.

18.2.18. The 'inline' Instance

```
1 \DeclareInstance{exsheets-heading}{inline}{default}{
2   inline           = true ,
3   number-pre-code  = \space ,
4   number-post-code = \space ,
5   join             =
6     {
7       main[r,vc]title[r,vc](0pt,0pt) ;
8       main[r,vc]number[l,vc](0pt,0pt)
9     }
10 }
```

Text before **Exercise 50.** An 'inline' heading. 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.Text after

18.2.19. The 'inline-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{inline-wp}{default}{
2   inline           = true ,
3   number-pre-code  = \space ,
4   number-post-code = \space ,
5   points-pre-code  = ( ,
6   points-post-code = )\space ,
7   join             =
8     {
9       main[r,vc]title[r,vc](0pt,0pt) ;
10      main[r,vc]number[l,vc](0pt,0pt) ;
11      main[r,vc]points[l,vc](0pt,0pt)
12    }
13 }
```

Text before **Exercise 51.** (1 P.) An 'inline-wp' heading. 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.Text after

18.2.20. The 'inline-nr' Instance

Introduced in
version 0.5

```
1 \DeclareInstance{exsheets-heading}{inline-nr}{default}{
2   inline            = true ,
3   number-post-code  = \space ,
4   join              = { main[r,vc]number[l,vc](0pt,0pt) }
5 }
```

Text before **52.** An 'inline-nr' heading. 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.Text after

18.2.21. The 'centered' Instance

```
1 \DeclareInstance{exsheets-heading}{centered}{default}{
2   join              = { title[r,B]number[l,B](.333em,0pt) } ,
3   attach            =
4     {
5       main[hc,vc]title[hc,vc](0pt,0pt) ;
6       main[r,vc]points[l,vc](\marginparsep,0pt)
7     }
8 }
```

<div align="center">

Exercise 53.   1 P.

</div>

A 'centered' heading. 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.

18.2.22. The 'centered-wp' Instance

```
1 \DeclareInstance{exsheets-heading}{centered-wp}{default}{
2   points-pre-code  = ( ,
3   points-post-code = ) ,
4   join             =
5     {
6       title[r,B]number[l,B](.333em,0pt) ;
7       title[r,B]points[l,B](.333em,0pt)
8     } ,
9   attach           = { main[hc,vc]title[hc,vc](0pt,0pt) }
10 }
```

Exercise 54. (1 P.)

A 'centered-wp' heading. 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.

18.2.23. The 'margin' Instance

```
1 \DeclareInstance{exsheets-heading}{margin}{default}{
2   runin            = true ,
3   number-post-code = \space ,
4   points-pre-code  = ( ,
5   points-post-code = )\space ,
6   join             = { title[r,b]number[l,b](.333em,0pt) } ,
7   attach           =
8     {
9       main[l,vc]title[r,vc](0pt,0pt) ;
10      main[l,b]points[r,t](0pt,0pt)
11    }
12 }
```

Exercise 55. A 'margin' heading. This is some sample text we will use to create a somewhat longer text (1 P.) 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.

18.2.24. The 'margin-nr' Instance

```
1 \DeclareInstance{exsheets-heading}{margin-nr}{default}{
2   runin  = true ,
3   attach =
4     {
5       main[l,vc]number[r,vc](-.333em,0pt) ;
6       main[r,vc]points[l,vc](\linewidth+\marginparsep,0pt)
7     }
8 }
```

56. A 'margin-nr' heading. This is some sample text we will use to create a somewhat longer text   1 P.
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.

18.2.25. The 'raggedleft' Instance

```
1 \DeclareInstance{exsheets-heading}{raggedleft}{default}{
2   join            = { title[r,B]number[l,B](.333em,0pt) } ,
3   attach          =
4     {
5       main[r,vc]title[r,vc](0pt,0pt) ;
6       main[r,vc]points[l,vc](\marginparsep,0pt)
7     }
8 }
```

Exercise 57.   1 P.

A 'raggedleft' heading. 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.

18.2.26. The 'fancy' Instance

```
1 \DeclareInstance{exsheets-heading}{fancy}{default}{
2   toc-reversed    = true ,
```

```
3    indent-first    = true ,
4    vscale          = 2 ,
5    pre-code        = \rule{\linewidth}{1pt} ,
6    post-code       = \rule{\linewidth}{1pt} ,
7    title-format    = \large\scshape\color{rgb:red,0.65;green,0.04;blue,0.07} ,
8    number-format   = \large\bfseries\color{rgb:red,0.02;green,0.04;blue,0.48} ,
9    points-format   = \itshape ,
10   join            = { number[r,B]title[l,B](.333em,0pt) } ,
11   attach          =
12      {
13        main[hc,vc]number[hc,vc](0pt,0pt) ;
14        main[l,vc]points[r,vc](-\marginparsep,0pt)
15      }
16 }
```

---

1 P.   **58. EXERCISE**

A 'fancy' heading. 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.

18.2.27. The 'fancy-wp' Instance

```
1  \DeclareInstance{exsheets-heading}{fancy-wp}{default}{
2    toc-reversed    = true ,
3    indent-first    = true ,
4    vscale          = 2 ,
5    pre-code        = \rule{\linewidth}{1pt} ,
6    post-code       = \rule{\linewidth}{1pt} ,
7    title-format    = \large\scshape\color{rgb:red,0.65;green,0.04;blue,0.07} ,
8    number-format   = \large\bfseries\color{rgb:red,0.02;green,0.04;blue,0.48} ,
9    points-format   = \itshape ,
10   points-pre-code  = ( ,
11   points-post-code = ) ,
12   join            =
13      {
14        number[r,B]title[l,B](.333em,0pt) ;
15        number[r,B]points[l,B](.333em,0pt)
16      } ,
17   attach          = { main[hc,vc]number[hc,vc](0pt,0pt) }
18 }
```

A 'fancy-wp' heading. 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.

18.3. Using an EXSHEETS Heading in Custom Code

It can be useful to have access to EXSHEETS headings in custom code. This is possible with the following command:

`\ExSheetsHeading{⟨instance⟩}{⟨title⟩}{⟨number⟩}{⟨points⟩}{⟨bonus⟩}{⟨id⟩}`

The meaning of the arguments is as follows:

- •⟨*instance*⟩: the name of the headings instance to be used.
- •⟨*title*⟩: the contents of the `title` coffin.
- •⟨*number*⟩: the contents of the `number` coffin.
- •⟨*points*⟩: The number of points given to the question. If non-zero this will cause the points to be printed in the `points` coffin.
- •⟨*bonus*⟩: the same as ⟨*points*⟩ but for bonus points.
- •⟨*id*⟩: the ID of the question this heading belongs to.

In combination with `\ForEachQuestion` the command can be used to build a custom list of questions. An example of its usage can be seen at the German Q&A site TEXwelt: `http://texwelt.de/wissen/fragen/6698#6738`.

18.4. Load Custom Configurations

If you have custom configurations you want to be loaded automatically then save them in a file `exsheets_configurations.cfg`. If this file is present it will be loaded `\AtBeginDocument`.

Part III.
The EXSHEETS-LISTINGS Package

19. The Problem

I knew the day would come when people would ask how to include verbatim material in the `question` and `solution` environments. Since they're defined with the environ package they're reading their environment bodies like macros read their arguments. This makes it impossible to

use verbatim material inside them.[16] Now the day has come [Bra13]. Soon after the first question appeared I wrote the first draft for ExSheets-listings for a question on TeX.sx [Ass13].

20. The Proposed Solution

The ExSheets-listings package defines listings environments that place their contents inside `question` and `solution` environments. They do this by writing the listing to a unique auxiliary file – questions to `\jobname-ex`⟨*num*⟩`.lst` and solutions to `\jobname-sol`⟨*num*⟩`.lst` where ⟨*num*⟩ is an increasing integer that ensures that each listing gets a unique file name. Those files are then included with `\lstinputlisting` if and when the question or solution is printed.

`\begin{lstquestion}[`⟨*options*⟩`]`
  A listings environment placed in a `question`.

`\begin{lstsolution}[`⟨*options*⟩`]`
  A listings environment placed in a `solution`.

  All you have to do to use the package is loading it the usual way:

```
1 \usepackage{exsheets-listings}
```

This will also load the packages ExSheets and listings if they're not loaded already.

```
1 % this uses my listings style used in this documentation for all pieces of
2 % code:
3 \begin{lstquestion}[%
4     pre=Explain what this piece of \TeX\ code does:,
5     listings={style=cnltx}]
6   \begingroup\expandafter\expandafter\expandafter\endgroup
7   \expandafter\ifx\csname foo\endcsname\relax
8   ...
9   \else
10  ...
11  \fi
12 \end{lstquestion}
```

1

---

16. See the TeX FAQ http://www.tex.ac.uk/cgi-bin/texfaq2html?label=verbwithin for reasons why.

> **Exercise 60.**
>
> Explain what this piece of TEX code does:
>
> ```
> 1 \begingroup\expandafter\expandafter\expandafter\endgroup
> 2 \expandafter\ifx\csname foo\endcsname\relax
> 3 ...
> 4 \else
> 5 ...
> 6 \fi
> ```

The example already shows two options of these environments. Here is the complete list:

pre = {⟨*text*⟩}
  ⟨*text*⟩ is placed before the code in the question or solution environment.

post = {⟨*text*⟩}
  ⟨*text*⟩ is placed after the code in the question or solution environment.

options = {⟨*options*⟩}
  Options passed to underlying the question or solution environment.

points = {⟨*points*⟩}
  The points assigned to the underlying question environment.

listings = {⟨*options*⟩}
  Options passed to the underlying listings environment.

  There are also two new options for ExSheets that can be set with \SetupExSheets:

question » listings = {⟨*options*⟩}
  Options passed to the underlying listings environment of lstquestion.

solution » listings = {⟨*options*⟩}
  Options passed to the underlying listings environment of lstsolution.

21. Own Environments

\NewLstQuSolPair[⟨*options for both environments*⟩]{⟨*lst question env*⟩}{⟨*question env*⟩}[⟨*options for lst question env*⟩]{⟨*lst solution env*⟩}{⟨*solution env*⟩}[⟨*options for lst solution env*⟩]
Defines two new listings environments that place the listing in a question environment ⟨*question env*⟩ or a solution environment ⟨*solution env*⟩. Those underlying environments should be environments as defined by \NewQuSolPair. The different options allow to preset options for the newly defined environments.

  The existing environments have been defined like this:

```
1  \NewLstQuSolPair{lstquestion}{question}{lstsolution}{solution}
```

Part IV.
Appendix

A. A List of all Solutions used in this Manual

Solution 9.
This is the solution to exercise 9!

Solution 10.
See? This solution gets printed where you have put it in the code of your document.

Fancy name 11.
See? This solution gets printed where you have put it and has a fancy name!

Solution 18.
This will be one of a few solutions printed by the following call of `\printsolutions`.

Solution 19.
The solution to the question with the nice ID.

Solution 20.
Yet another solution.

Solution 27.
To center a short part of the text body one can use the `center` environment (1 P.). Inside an environment like `table` one should use `\centering` (1 P.). For single lines there is also the `\centerline` command (1 P.).

Solution 28.
For example two volumes from *Computers & Typesetting*: *TEXbook*; *TEX*; *METAFONTbook*; *METAFONT*; *Computer Modern Typefaces*. Each valid answer is worth 1 P.

★ **Solution 29.**
This is what the solution looks like.

Solution 30.
Try to fill in these blanks. All of them are created by using the `\blank` command.

B. Bibliography

[Ale01]   Jason ALEXANDER. examdesign. version 1.1, Mar. 26, 2001 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/examdesign/.

[Ars11]   Donald ARSENEAU. ulem. Mar. 18, 2011 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/ulem/.

[Ass13]   Forkrul ASSAIL.
          How do I extract repeated functionality (ExSHEETS workaround) to make it reusable?
          Sept. 18, 2013.
          URL: http://tex.stackexchange.com/q/133907/5049 (visited on 09/22/2013).

[Bra13]   Stefano BRAGAGLIA.
          Fixing lstlisting inside ExSHEETS question and solution environments.
          Sept. 4, 2013.
          URL: http://tex.stackexchange.com/q/131546/5049 (visited on 09/22/2013).

[Dae13]   Walter DAEMS. exsol. version 0.6, May 12, 2013 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/exsol/.

[GMS94]   Michel GOOSSENS, Frank MITTELBACH, and Alexander SAMARIN.
          The LaTeX Companion. 1st ed. Reading, Mass.: Addison-Wesley, 1994. 528 pp.

[Hir15]   Philip HIRSCHHORN. exam. version 2.5, May 7, 2015 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/exam/.

[Ker07]   Uwe KERN. xcolor. version 2.11, Jan. 21, 2007 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/xcolor/.

[Knu84a] Donald E. KNUTH. *Computers & Typesetting.* 5 vols.
          Reading, Mass.: Addison-Wesley, 1984–1986.

[Knu84b] Donald E. KNUTH. *Computers & Typesetting.* Vol. A: *The TEX book.*
          Reading, Mass.: Addison-Wesley, 1984.

[Knu86a] Donald E. KNUTH. *Computers & Typesetting.* Vol. B: *TEX: The Program.*
          Reading, Mass.: Addison-Wesley, 1986.

[Knu86b] Donald E. KNUTH. *Computers & Typesetting.* Vol. C: *The METAFONTbook.*
          Reading, Mass.: Addison-Wesley, 1986.

[Knu86c] Donald E. KNUTH. *Computers & Typesetting.* Vol. D: *METAFONT: The Program.*
          Reading, Mass.: Addison-Wesley, 1986.

[Knu86d] Donald E. KNUTH. *Computers & Typesetting.* Vol. E: *Computer Modern Typefaces.*
          Reading, Mass.: Addison-Wesley, 1986.

[L3Pa]   THE LATEX3 PROJECT TEAM.
          l3experimental. version SVN 6377, Jan. 19, 2016 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/l3experimental/.

[L3Pb]   THE LATEX3 PROJECT TEAM. l3kernel. version SVN 6377, Jan. 19, 2016 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/l3kernel/.

B. Bibliography

[L3Pc]   THE LATEX3 PROJECT TEAM.
          l3packages. version SVN 6377, Jan. 19, 2016 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/l3packages/.

[Leh15]   Philipp LEHMAN, current maintainer: Joseph WRIGHT.
          etoolbox. version 2.2a, Aug. 2, 2015 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/etoolbox/.

[MV13]   Grazia MESSINEO and Salvatore VASSALLO.
          esami. version 1.0, Mar. 8, 2013 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/esami/.

[Pic12]   Paul PICHAUREAU. exercise. version 1.58, May 8, 2012 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/exercise/.

[Pif10]   Mike PIFF, current maintainer: Joseph WRIGHT.
          answers. version 2.13, Oct. 11, 2010 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/answers/.

[Rob13]   Will ROBERTSON. environ. version 0.3, Apr. 1, 2013 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/environ/.

[Sto11]   D. P. STORY. eqexam. version 3.0k, Sept. 1, 2011 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/eqexam/.

[Tal12]   Nicola L. C. TALBOT. probsoln. version 3.04, Aug. 23, 2012 (or newer).
          URL: http://mirror.ctan.org/macros/latex/contrib/probsoln/.

[Tan13]   Till TANTAU. TikZ/pgf. version 3.0.0, Dec. 13, 2013 (or newer).
          URL: http://mirror.ctan.org/graphics/pgf/.

C. Index