THE worksheet DOCUMENT CLASS
SAMUEL S. WATSON
MAY 17, 2018

Name:

The document class `worksheet` is designed for formatting problems and solutions using the `tcolorbox` package. It supports formatting a problems-only and solutions-included document from a single source file. The `solutionbox` environment is for blank solution space, while the `solution` environment is for the actual solutions.

1 Basic commands

The main environments are `problem` and `solution`. Standard `tcolorbox` options, like `height = 5cm` or `height fill` can be passed to the tcolorbox.

Typically one would have a `problem` environment followed by a `solutionbox` environment followed by a `solution` environment. Only one of the latter two will be displayed, according to whether the `solutions` document option is invoked (that is `\documentclass[solutions]{worksheet}` will show solutions, while `\documentclass[]{worksheet}` will show empty boxes).

```
\begin{problem}
  $\vdots$
\end{problem}
```

Problem 1

$\vdots$

```
\begin{solution}[height = 5cm]
  $\vdots$
\end{solution}
```

Solution

$\vdots$

Equations, like (1.1) below, are numbered on a per-problem basis.

$$\int \frac{1}{1+x^2}\,\mathrm{d}x = \arctan x. \tag{1.1}$$

You can put a final answer box in the lower right-hand corner with

```
\finalanswer[$\frac{5}{2}$]
% or
\finalanswer[$\frac{5}{2}$][3cm][4cm]
```

or, for an empty final answer box, just `\finalanswer`. The macro `\solfinalanswer` gives a height-filling empty solution box with an empty final answer box.

2 Header commands

The header is produced with the `\header` command. A name can be included via `\headerwithname`, which takes two arguments: the header content and the width of the name box. So the header above was produced with

```
\headerwithname{
    The \texttt{worksheet} document class \\
    Samuel S.\ Watson \\ \today
  }{
    3 cm
  }
```

3 Other commands

We can also make a problem using the `\pb` shortcut

```
\pb{Another example problem}
```

Problem 2

Another example problem

Likewise, we can produce a solutionbox using `\sol`, which takes an optional argument of tcolorbox options, like `height = 5cm`.

We can put two problems side by side using the `problempair` environment (in which one must place two `problem` environments) or using `twoproblems`.

```
\twoproblems{first problem contents}{second problem contents}
```

Problem 3

first problem contents

Problem 4

second problem contents

We can also put two empty solution boxes side by side. The argument is any list of `tcolorbox` options.

```
\twosolutions{height = 5cm}
```

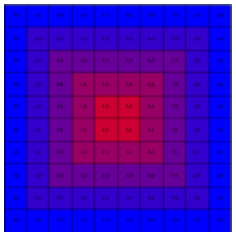
Solution

Solution

4 Inset figures

The `wrapfigure` package doesn't play so nicely with tcolorboxes. But `worksheet.cls` has its own `insetfigure` command which juxtaposes boxes containing its first and second arguments, with alignment specified by the optional third argument (`c` for center, `t` for top, and `b` for bottom, `t` by default) and spacing specified by the optional fourth argument (`16pt` by default). Here's an example:

```
\insetfigure{
  \lipsum[1]
}{
  \includegraphics[width=3cm]{example-grid-100x100pt}
}
\lipsum[2-3]
```



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.
Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

5 Solutions

The accompanying Python script `processsolutions.py` can be used to simultaneously produce a pdf with solutions and one without. The only caveat here is that the square brackets in the `documentclass` command must be present, even if they're empty.

6 Julia integration

The `pythontex` package can be used to weave Julia code into the worksheets. Combined with SymPy, which can be accessed from Julia, this can be pretty powerful. For example:

```
\begin{jlcode}
  A = rand(-5:5,5,10)
  x = rand(-5:5,10)
\end{jlcode}

\[
  \jltex{A} \jltex{x} = \jltex{A*x}.
\]
```

$$\begin{bmatrix} -2 & -3 & 5 & 0 & -3 & -4 & -5 & -5 & 5 & -5 \\ -5 & 3 & -3 & 2 & -2 & 2 & -3 & -4 & -5 & -2 \\ 2 & 0 & 4 & 5 & 4 & -3 & 0 & 2 & -5 & -2 \\ 5 & -1 & -1 & -5 & 4 & -1 & -5 & 4 & 3 & 2 \\ 4 & 0 & 4 & 1 & 3 & 4 & -2 & 4 & 4 & -5 \end{bmatrix} \begin{bmatrix} -2 \\ -3 \\ -2 \\ -3 \\ 5 \\ -4 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ -24 \\ -2 \\ 39 \\ -21 \end{bmatrix}.$$

```
\begin{jlcode}
  I = Integral(exp(x^2),x)
\end{jlcode}
```

```
\[
  \jltex{I} = \jltex{doit(I)}.
\]
```

$$\int e^{x^2}\,dx = \frac{\sqrt{\pi}}{2}\,\operatorname{erfi}(x).$$

The Julia function `p`, which simply returns its argument in string form, also increments a global Julia variable `POINTS`, which can be used to keep a running total of the points assigned so far.

Problem 5 (5 points)

This is the first problem.

Problem 6 (8 points)

This is the second problem.

Total points so far: 13

You can use `Plots.jl` with its `pgfplots` backend to generate figures which can then be included.