beamerswitch — Convenient mode selection in Beamer documents

Alex Ball

Class v1.7 — 2020/06/06

This class is a wrapper around the beamer class to make it easier to use the same document to generate the different forms of the presentation: the slides themselves, an abbreviated slide set for transparencies or online reference, an n-up handout version, and a transcript or set of notes using the article class.

To contact the author about this package, please visit the GitHub page where the code is hosted: https://github.com/alex-ball/beamerswitch.

1 Quick start guide

Here are the key facts:

- The beamerswitch class makes it easier to switch between beamer modes.
- It is aimed at you if you want to generate handouts from your presentation, whether printouts of your slides or something more like a paper or article.
- It is not a drop-in replacement for beamer.

If you have an existing beamer presentation and want convert it to use beamerswitch instead, here’s what you need to do.

1. Have you specified class options other than ignorenonframetext? If so, start by rewriting your \documentclass line in terms of \PassOptionsToClass – note that the optional argument (in square brackets) becomes mandatory (in curly braces):

   Before
   \documentclass[10pt]{beamer}

   After
   \PassOptionsToClass{10pt}{beamer}

   Then add ‘\documentclass{beamerswitch}’ directly below.

   Otherwise, simply replace your \documentclass line with the beamerswitch one.

2. If you did not specify ignorenonframetext as one of your beamer options, add a ‘\mode<all>’ line directly after ‘\begin{document}’.
3. Wrap any preamble content intended only for your slides (like \usetheme or \usepackage commands) with ‘\mode<presentation>\{…\}\’ or something more specific, as required.

In summary, here is an example:

**Before**

\documentclass[10pt]{beamer}  
\usetheme{metropolis}  
\title{Test presentation}  
\begin{document}  
\maketitle  
\end{document}

**After**

\PassOptionsToClass{10pt}{beamer}  
\documentclass{beamerswitch}  
\mode<presentation>{\usetheme{metropolis}}  
\title{Test presentation}  
\begin{document}  
\mode<all>\maketitle  
\end{document}

At this point the document should compile exactly as before.

If you intend to use article mode at all, I strongly suggest that you proceed by converting your document body so it that works without the ‘\mode<all>\’ line. That means making sure all slide content is in a frame environment, new command definitions are moved to the preamble, and so on.

Lastly, read through the rest of this manual and see what beamerswitch can do for you!

## 2 Introduction

With beamer, it is possible to typeset the same document code in different ways to get different effects. The result you get depends on a potentially confusing mix of options, modes, and indeed classes.

Beamer has five modes for typesetting content. There are three modes that produce regular slides:

- The beamer mode relates to the normal, default slide set.
- The trans class option switches to the mode of the same name. It is intended for transparencies, but is really just an alternative mode that ignores ‘bare’ overlay specifications.
- The handout class option switches to the mode of the same name. It is intended for print-friendly versions, but is really just another alternative mode that ignores ‘bare’ overlay specifications.

The beamer manual shows how to use handout mode in conjunction with pgfpages to get several slides on a single side of A4 (or Letter) paper.

Beamer can also produce a double-height or double-width slide set, with the intention that each half will be shown on a different display (e.g. one for the audience, one for the speaker). There are three variations of this, activated using \setbeameroption:
• **show notes on second screen** uses the second screen for text marked up using `\note{<text>}`.

• **second mode text on second screen** uses the second screen for showing the slide typeset in second mode rather than beamer. Unlike trans and handout, second responds to ‘bare’ overlay specifications.

• **previous slide on second screen**, uses the second screen either for showing the previous slide or, if the slide has the `typeset second` option set, for showing the current slide typeset in second mode.

The final variation is to use a different class altogether, such as article, in conjunction with beamerarticle. In this case the content is set free-flowing in article mode, without any of the frame furniture.

The beamer manual suggests coping with all these variations by having the document code in one file, and using it as input to other files that each set up a different mode of operation. This is fine but a bit of a fiddle. It would be nice to be able to get the same effect using a single file and, ideally, a single command invocation.

The beamerswitch class addresses this issue by acting as a wrapper around the various options, and providing a common interface for switching between modes. More specifically, it has three main functions:

1. To provide more choice of handout-mode layouts than pgfpages gives you out-of-the-box. Additionally, I hope you will find the method for selecting them more memorable.

2. To enable you to switch to article mode with a simple change of class option, instead of having to fiddle with commented-out `\documentclass` and `\usepackage` lines.

3. To allow you to override the beamer mode from the command line, by choosing a given jobname suffix. Primarily this is to allow you to generate the different versions programmatically. Indeed, the class provides facilities for generating multiple versions with a single command.

### 3 Dependencies

To use beamerswitch, you will need to have the following packages available and reasonably up to date on your system. All of these ship with recent \TeX\X distributions.

- beamer
- etoolbox
- hyperref
- iftex
- pgf
- shellesc
- xkeyval
- xstring

One feature of the class uses latexmk by default, though you can configure it to use something else if you need or want to.
4 Loading the class

The class is loaded in the usual way:

\documentclass[\{options\}]{beamerswitch}

The various options are described below.

4.1 Choosing the mode of the current run

The beamerswitch class, as explained above, does not do much itself but rather helps you to switch between beamer modes, specifically article, beamer, trans and handout. Note that it loads beamer with the ignorenonframetext option, so that text outside frames is only shown in article mode.

The normal way of choosing the mode is to use the respective class options.

article\hspace{1cm} (no value, initially unset)

Switches to article mode, which uses the article class and resembles a normal article.

beamer\hspace{1cm} (no value, initially set)

Switches to beamer mode, which uses the beamer class and resembles a normal slideshow.

handout\hspace{1cm} (no value, initially unset)

Switches to handout mode, which uses the beamer class but uses a different series of overlay specifications. It resembles a set of printed pages with multiple slides shown on each page.

trans\hspace{1cm} (no value, initially unset)

Switches to trans mode, which uses the beamer class and resembles a normal slideshow, but uses a different series of overlay specifications.

There is, however, a sneaky second way of setting the mode that overrides the first, and that is to use the \jobname. By default, this is the name of your \LaTeX file minus the .tex extension, but you can set it to something else when you run \LaTeX. If you set the \jobname to end in one of the following suffixes, the mode will automatically switch:

- -article will switch to article mode.
- -handout will switch to handout mode.
- -slides will switch to beamer mode.
- -trans will switch to trans mode.

The idea is that you can keep your source document the same, but by running \LaTeX with an alternative \jobname, you can get a different version out with a meaningfully different file name.

Of course, you may not want to use those suffixes. Perhaps you want them in German; perhaps your document's file name already ends in one of them; perhaps ‘article’ or ‘trans’ doesn’t describe what you’re using those modes for. The suffixes are provided by the following commands:
\ArticleSuffix
Holds the \jobname suffix that triggers article mode.

\BeamerSuffix
Holds the \jobname suffix that triggers beamer mode.

\HandoutSuffix
Holds the \jobname suffix that triggers handout mode.

\TransSuffix
Holds the \jobname suffix that triggers trans mode.

The CamelCase is an admittedly rather obscure signal to you that, if you want to change them to something else, you should do so via \newcommand before loading the class:

\newcommand*{\ArticleSuffix}{-script}
\documentclass{beamerswitch}

4.2 Using more than one mode at once

Another handy feature of the class is that it can spawn parallel compilations, so you could in theory generate all four versions from a single command. It achieves this magic by escaping to the shell and running latexmk. Thus for it to work you need to run \TeX with shell escape enabled and you need latexmk to be installed.

alsoarticle (no value, initially unset)
Spawns a new compilation process in article mode.

alsobeamer (no value, initially unset)
Spawns a new compilation process in beamer mode.

alsohandout (no value, initially unset)
Spawns a new compilation process in handout mode.

alsotrans (no value, initially unset)
Spawns a new compilation process in trans mode.

also={\langle comma-separated list of modes\rangle} (no default, initially empty)
Spawns compilation processes in each of the specified modes. Note that the list has to be wrapped in braces, and only the four aforementioned modes are recognized.

If you would rather use a tool other than latexmk to managed your spawned compilation processes, it is possible to do that. Bear in mind, though, that beamerswitch is not clever enough to spot if you have already compiled the other version on a previous run so you have to take care of that yourself. The command that does the business is this:

\BeamerswitchSpawn{\langle suffix\rangle}
Spawns a new compilation process with \langle suffix\rangle appended to the \jobname.

By default, \BeamerswitchSpawn checks which \TeX engine you are running, and then runs the matching one of \SpawnedPDFTeX, \SpawnedXeTeX, \SpawnedLuaTeX, or if all the tests fail, \SpawnedTeX, with the converted \jobname. These commands all add an engine-selecting option to a base command, \SpawnedCompiler, which by default is
latexmk with synctex enabled, shell escape disabled and interaction set to batch mode. See the Implementation section below for details.

Any of these commands can be pre-defined to something else, depending on what you want to do. So, for example, if you want to change the base latexmk options, pre-define \SpawnedCompiler to latexmk plus your chosen options:

\newcommand*{\SpawnedCompiler}{%
  latexmk -silent -shell-escape -interaction=nonstopmode}
\documentclass{beamerswitch}

If you want to switch to using rubber, say, pre-define \SpawnedPDFTeX and friends accordingly (perhaps taking advantage of \SpawnedCompiler to reduce repetition):

\newcommand*{\SpawnedCompiler}{rubber -q --synctex}
\newcommand*{\SpawnedPDFTeX}{\SpawnedCompiler -m pdftex}
\newcommand*{\SpawnedXeTeX}{\SpawnedCompiler -m xelatex}
\documentclass{beamerswitch}

If you don't need the engine selection routine provided, simply pre-define \BeamerswitchSpawn to do whatever you need it to do:

\newcommand*{\BeamerswitchSpawn}[1]{%
  \ShellEscape{... -jobname=\jobname#1 \jobname}%
}
\documentclass{beamerswitch}

If you want to use arara to compile everything, the class options won't help you as arara does not let you change \jobnames on the command line. That's no problem, though, because you can do it manually within your list of directives:

% arara: lualatex
% arara: lualatex: { options: "-jobname beamerswitch-example-article" }
% arara: lualatex: { options: "-jobname beamerswitch-example-handout" }
% arara: lualatex: { options: "-jobname beamerswitch-example-trans" }
\documentclass{beamerswitch}

4.3 Improving compatibility

nohyperref (no value, initially unset)

The beamer class loads hyperref for you, but when you switch to article mode, beamerarticle doesn’t, so it is easy for you to get caught out. To protect you from this, beamerswitch does load hyperref in article mode, with the pleasing side effect that \subject and \keywords then work as intended instead of throwing errors.

To achieve this, the class has to load hyperref quite early on, which can cause trouble with certain other packages. If you would rather load hyperref yourself at a different point, use this option.
textinst

In all modes, patches the \inst command so that it prints its argument using \textsuperscript instead of a mathematical superscript. This helps avoid unnecessary font changes.

5 Setting the layout of the handouts

The class provides a simplified interface to the pgfpages package. By default, it prints six slides to a side of A4 paper, but you can change this by using the following command in the preamble.

\handoutlayout\{\textlangle options\rangle\}

Configures the layout of the page when using handout mode. It has no effect in other modes. The available \textlangle options\rangle are listed below.

\texttt{paper=\{paper size\}}

Selects the size of paper to use for handout mode. The value is passed directly to pgfpages, so consult the documentation of that package for the allowed values. You can find them under the documentation for the resize to layout.

\texttt{nup=1|1plus|2|3|3plus|4|4plus|6|8}

Selects how many slides are shown on a single page in handout mode. The ‘plus’ layouts leave blank space for recipients to make handwritten notes next to each slide. The effects of the various values are shown in Figure 1 on page 8. Note that the 2, 3plus, 4 and 6 layouts are intended for slides in the usual 4:3 aspect ratio, while the 3, 4plus and 8 layouts are intended for widescreen slides.

\texttt{pnos}

Adds page numbers to the bottom of each page. The number appears at a distance of \texttt{\handoutpnobaseline} (initially 5mm) from the bottom edge of the page. You can change this before calling \texttt{\handoutlayout}, if you like, using the usual mechanisms (e.g. \texttt{\setlength}).

\texttt{border=\{length\}}

Puts a rectangular border of thickness \texttt{\{length\}} around each slide.

\texttt{margin=\{length\}}

Sets how much extra space to leave around each slide on the page. To achieve this it uses the border shrink option from pgfpages. If you add page numbers, this value will increase if necessary to prevent the number colliding with the slides, so treat it as a minimum.

\texttt{align}

When margin is non-zero, the effective page margins are approximately half the size of the gaps between slides (some additional white space comes from preserving the aspect ratio of the slides). When set, this option tweaks the positions of the slides so that the page margins are approximately the same as the interior gaps.
6 Changing the look of article mode

Some additional options can be set by using the following command in the preamble.

\articlelayout{(options)}

Configures the appearance of article mode. It has no effect in other modes. The available \textit{options} are listed below.

\textbf{frametitles=para|margin|none} \\
In article mode, affects how frame titles are printed. By default, beamerarticle prints them as paragraph headings, represented by the value \texttt{para}. To have them printed in the margin (using \texttt{marginpar}), use \texttt{margin}. To omit them altogether, use the value \texttt{none}.

\textbf{maketitle} \\
(no value, initially unset)

In article mode, adjusts the \texttt{maketitle} routine:

\begin{itemize}
  \item The title is printed closer to the top margin.
\end{itemize}
7 Tips for further configuration

There are some other ways to customize the behaviour of the various modes. You can use the standard \LaTeX{} methods for customizing how the \texttt{article} and \texttt{beamer} classes are loaded:

\begin{verbatim}
\PassOptionsToClass{a4paper,11pt}{article} % for article mode
\PassOptionsToClass{utf8}{beamer} % for beamer, handout, trans modes
\documentclass{beamerswitch}
\end{verbatim}

And of course there is the standard \texttt{beamer} way of passing different options to different modes:

\begin{verbatim}
\documentclass{beamerswitch}
\mode<article>{
  \usepackage[utf8]{inputenc}
}
\mode<beamer>{
  \setbeameroption{second mode text on second screen}
}
\end{verbatim}

8 Feedback

I hope you find this class useful. Please report any bugs and add any suggestions for improvements or new features to the Issue Tracker on GitHub.
9 Implementation

9.1 Dependencies

We use the following packages:

- \texttt{xkeyval} with \texttt{xkvltxp} for setting options
- \texttt{etoolbox} for command patches and list processing
- \texttt{xstring} for comparisons
- \texttt{shellesc} for running parallel compilations
- \texttt{iftex} for determining which engine to use

\begin{verbatim}
\RequirePackage{xkeyval,xkvltxp,etoolbox,xstring,shellesc,iftex}
\end{verbatim}

9.2 Class options

We recognize four main modes of operation: ‘beamer’, ‘trans’, ‘handout’, and ‘article’.

The \texttt{beamer} option triggers beamer mode.

\begin{verbatim}
\define@boolkey[DC]{beamerswitch}{beamer}{true}{%
  \ifbool{DC@beamerswitch@beamer}{%
    \setkeys[DC]{beamerswitch}{trans=false}
    \setkeys[DC]{beamerswitch}{handout=false}
    \setkeys[DC]{beamerswitch}{article=false}
  }{%
}
\end{verbatim}

The \texttt{trans} option triggers trans mode.

\begin{verbatim}
\define@boolkey[DC]{beamerswitch}{trans}{true}{%
  \ifbool{DC@beamerswitch@trans}{%
    \setkeys[DC]{beamerswitch}{beamer=false}
    \setkeys[DC]{beamerswitch}{handout=false}
    \setkeys[DC]{beamerswitch}{article=false}
  }{%
}
\end{verbatim}

The \texttt{handout} option triggers handout mode.

\begin{verbatim}
\define@boolkey[DC]{beamerswitch}{handout}{true}{%
  \ifbool{DC@beamerswitch@handout}{%
    \setkeys[DC]{beamerswitch}{beamer=false}
    \setkeys[DC]{beamerswitch}{trans=false}
    \setkeys[DC]{beamerswitch}{article=false}
  }{%
}
\end{verbatim}

The \texttt{article} option triggers article mode.
The **also** option allows the user to specify a set of alternative modes to typeset in parallel, in a comma-separated list. Alternatively, the user can specify the Boolean **also* options directly.

The **nohyperref** option stops the class from loading the *hyperref* package in article mode.

The **textinst** option adjusts the superscript used for institution markers.

The default behaviour is to use beamer mode only.

Now we process the options given by the user.

### 9.3 Jobname-based mode switching

We define some default values for the special suffixes.
We provide a special routine for spawning new \TeX processes. We allow for the possibility of the user overriding this routine with another one, perhaps using a different automation tool; it should take one argument, being the jobname suffix.

We check for special jobnames and use them to override the above mode-related options. In certain circumstances, the jobname is wrapped in quote marks, so in that case we strip them off before running the tests.

Now we run the tests. Note that the \texttt{also*} options are only acted upon if the all the tests fail.
9.4 Setting up modes

For article mode, we load the `article` class and the `beamerarticle` support package. Apologies for anyone hoping for `scartcl` or `memoir` alternatives.

It seems as though `beamerarticle` expects `hyperref` to be loaded, but doesn’t actually do it itself. So we oblige, using the default options specified by `beamer`.

While `beamer` takes care of adding presentation metadata to the PDF properties, `beamerarticle` misses the title and author properties. (It does manage to set the subject and keywords, though.) We achieve parity with some additional \texttt{\hypersetup}. Note that \texttt{beamerarticle} appends the subtitle to \texttt{@title} with a linebreak and this does odd things in the context of \texttt{pdftitle}, so we fix it with \texttt{pdfstringdefDisableCommands}.
For the presentation modes, we load the `beamer` class with appropriate options. Since we are targeting users wanting different versions of their presentations with the same code, we activate `ignorenonframetext`.

```latex
\ifbool{DC@beamerswitch@handout}{
  \LoadClass[ignorenonframetext,handout]{beamer}
}{
\ifbool{DC@beamerswitch@trans}{
  \LoadClass[ignorenonframetext,trans]{beamer}
}{{
  \LoadClass[ignorenonframetext]{beamer}
}}
```

Handout mode lays multiple slides out on a single page. For this we use `pgfpages`. The actual configuration is handled later.

```latex
\RequirePackage{pgfpages}
```

We also activate `ignorenonframetext` for the other two modes.

```latex
9.5 Mode-independent layout

We implement the option that formats institution markers in text mode rather than math mode.

```latex
\ifbool{DC@beamerswitch@textinst}{
  \def\beamer@insttitle#1 {\textsuperscript{#1}}
  \def\beamer@instinst#1 {\textsuperscript{#1}\ignorespaces}
}{
}
```

```latex
9.6 Handout layout

In theory it would be nice to anticipate the paper size that the article mode would use, and pass that as an option to `pgfpagesuselayout` but as that's unlikely to be clean code, we settle here for setting it with an option.

```latex
\def\beamerswitch@handoutpaper{a4paper}
\define@key[HL]{beamer}{paper}{% 
  \def\beamerswitch@handoutpaper{#1} %
}
```

The `nup` option specifies how many slides to include per page. The 'plus' keyword indicates a layout with additional gaps for writing.
The \texttt{borders} option switches on borders around the slides on handout pages (and gaps where slides would appear if there were enough). The value is used to set the width of the border.

This is implemented as a PGF drawing command, which is initially set up to do nothing but gets filled out if the option is selected. As part of this, we calculate the number of the current slide (\texttt{handoutpno} gives the current page number minus one; \texttt{\beamerswitch@nup} is the number of slides per page; and \texttt{\pgf@cpn} gives the count of the current slide on the current page). If this exceeds the total number of slides, we suppress the border.

The \texttt{margin} option controls how much white space is added around each slide on handout pages. It corresponds to the \texttt{border shrink} option of \texttt{\pgfpagesuselayout}.

The \texttt{align} option chooses whether to rejig the positions of the slides on the page so that the effective page margin is the same as, rather than half, the space between adjacent slides.

The \texttt{pnos} option switches on page numbers for handout pages. The height at which
the baseline of the number sits, relative to the bottom edge of the paper, is given by the
length \handoutpnobaseline.

\newlength{\handoutpnobaseline}
\setlength{\handoutpnobaseline}{5mm}
\newcounter{handoutpno}
\define@boolkey[HL]{beamerswitch}{pno}{true}{}

In order to implement the page numbers, we need to add a hook to the \pgfpages
shipout routine.

\def\beamerswitch@footer{}
\patchcmd{\pgfpages@buildshipoutbox}{\pgfsys@endpicture}{\beamerswitch@footer\pgfsys@endpicture}{}{}

We set up the \handoutlayout command for applying these options.

The hook \beamerswitch@footer is used to step the page number counter handoutpno;
if page numering is active, it also places the number on the page centered and
\handoutpnobaseline up from the bottom.

The length \beamerswitch@pnoadjust is, conceptually, the height of the page number.
When page numbers are active, the margin around slides is automatically increased
to guarantee that the top of the page number will not be higher than the bottom of
the lowest slide. Currently this does not account for extra whitespace arising from aspect
ratio issues. When the align option is active, the margin at the bottom of the page is
increased, but to save a register we make the adjustment to \beamerswitch@pnoadjust
instead of a copy of \beamerswitch@margin.

\newlength{\beamerswitch@pnoadjust}
\newcommand*{\handoutlayout}[1]{%
\only<handout>{%
\setlength{\beamerswitch@pnoadjust}{1em}%
\setkeys[HL]{beamerswitch}{#1}%
\ifbool{HL@beamerswitch@pno}{%
\def\beamerswitch@footer{%
\stepcounter{handoutpno}%
\setbox0\vbox{\makebox[0pt][c]{\arabic{handoutpno}}}%
\pgfsys@beginscope
\pgflowlevel{\pgftransformshift{%
\pgflowlevel{\pgftransformshift{%
\pgflowlevel{\pgftransformshift{%
\pgflowlevel{\pgftransformshift{%
\pgflowlevel{\pgftransformshift{%
\pgfpoint{0.5\pgfphysicalwidth}{\handoutpnobaseline}}}%
\wd0=0pt%
\dp0=-\ht0%
\pgfsys@hbox0%
\pgfsys@endscope
}\ifbool{HL@beamerswitch@align}{%
\ifcase\value{beamerswitch@nupcase}\relax
\relax
\or
\addtolength{\beamerswitch@pnoadjust}{-0.333\beamerswitch@margin}%
\or

16
The value of the `nup` option is used both to select the layout and set the value of \texttt{\beamer@nup}. The latter is used to suppress unwanted borders and check if \texttt{\handoutlayout} has been called.

\ifcase\value{\beamer@nupcase}\relax
  \def\beamer@nup{1}\relax
  \pgfpagesuselayout{1 by 1}{\beamer@handoutpaper,\%
  landscape, border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{1}\relax
  \pgfpagesuselayout{1 by 1 narrow}{\beamer@handoutpaper,\%
  border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{2}\relax
  \pgfpagesuselayout{1 by 2}{\beamer@handoutpaper,\%
  border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{3}\relax
  \pgfpagesuselayout{1 by 3}{\beamer@handoutpaper,\%
  border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{3}\relax
  \pgfpagesuselayout{1 by 3 narrow}{\beamer@handoutpaper,\%
  border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{4}\relax
  \pgfpagesuselayout{2 by 2}{\beamer@handoutpaper,\%
  landscape, border shrink=\texttt{\the\beamer@margin}}\%
  \or
  \def\beamer@nup{4}\relax
  \pgfpagesuselayout{1 by 4 narrow}{\beamer@handoutpaper,\%
  border shrink=\texttt{\the\beamer@margin}}\%
\fi
We initialize the class with the defaults set above.

\AtEndPreamble{\ifdef\beamerswitch@nup{}{\handoutlayout}{}

Though \texttt{pgfpages} defines some perfectly fine layouts, we need to add configurability to the existing ones and provide some new ones.

The ‘1 by 1’ layout has a single slide per page.

\mode<handout>{
  \pgfpagesdeclarerelayout{1 by 1}
  {
    \edef\pgfpageoptionheight{\the\paperheight}
    \edef\pgfpageoptionwidth{\the\paperwidth}
    \edef\pgfpageoptionborder{0pt}
    \def\pgfpageoptionfirstshipout{1}
    \def\pgfpageoptionfirstshipout{1}
  }
}

\pgfpagesphysicalpageoptions
{%
  logical pages=1,%
  physical height=\pgfpageoptionheight,%
  physical width=\pgfpageoptionwidth,%
  current logical shipout=\pgfpageoptionfirstshipout%
%
}\pgfpageslogicalpageoptions{1}
{%
  border shrink=\pgfpageoptionborder,%
  border code=\beamerswitch@Border,%
  resized width=\pgfphysicalwidth,%
  resized height=\pgfphysicalheight,%
  center=\pgfpoint{.5\pgfphysicalwidth}{.5\pgfphysicalheight}%
%
}\pgfpagesdeclarerelayout{1 by 1 narrow}
The ‘1 by 2’ layout is similar to the normal 2 on 1 layout.
\begin{verbatim}
\pgfpagesphysicalpageoptions
{
  logical pages=2,
  physical height=\pgfpageoptionheight,%
  physical width=\pgfpageoptionwidth,%
  current logical shipout=\pgfpageoptionfirstshipout%
}
\ifdim\paperheight >\paperwidth\relax
% put side-by-side
\pgfpageslogicalpageoptions{1}
{
  border shrink=\pgfpageoptionborder,%
  border code=\beamerswitch@Border,%
  resized width=.5\pgfphysicalwidth,%
  resized height=\pgfphysicalheight,%
  center={\ifbool{HL@beamerswitch@align}{%
    \pgfpoint{.25\pgfphysicalwidth + \pgfmeta@center}{.5\pgfphysicalheight}}%
  \pgfpoint{.25\pgfphysicalwidth}{.5\pgfphysicalheight}}%
}
\pgfpageslogicalpageoptions{2}
{
  border shrink=\pgfpageoptionborder,%
  border code=\beamerswitch@Border,%
  resized width=.5\pgfphysicalwidth,%
  resized height=\pgfphysicalheight,%
  center={\ifbool{HL@beamerswitch@align}{%
    \pgfpoint{.25\pgfphysicalwidth + \pgfmeta@center}{.5\pgfphysicalheight}}%
  \pgfpoint{.25\pgfphysicalwidth}{.5\pgfphysicalheight}}%
}
\else
% stack on top of one another
\pgfpageslogicalpageoptions{1}
{
  border shrink=\pgfpageoptionborder,%
  border code=\beamerswitch@Border,%
  resized width=\pgfphysicalwidth,%
  resized height=.5\pgfphysicalheight,%
  center={\ifbool{HL@beamerswitch@align}{%
    \pgfpoint{.5\pgfphysicalwidth + \pgfmeta@center}{.75\pgfphysicalheight}}%
  \pgfpoint{.5\pgfphysicalwidth}{.75\pgfphysicalheight}}%
}
\pgfpageslogicalpageoptions{2}
{
  border shrink=\pgfpageoptionborder,%
  border code=\beamerswitch@Border,%
  resized width=\pgfphysicalwidth,%
  resized height=.5\pgfphysicalheight,%
}
\end{verbatim}
The '1 by 3' layout is similar to the '1 by 2', but with an extra row.
The ‘1 by 3 narrow’ layout is like the ‘1 by 3’ layout but restricted to the left (or top) half of the page.

```latex
\pgfpagesdeclarelayout{1 by 3 narrow}
{
  \edef\pgfpageoptionheight{\the\paperwidth} % landscaped by default
  \edef\pgfpageoptionwidth{\the\paperheight}
  \def\pgfpageoptionborder{0pt}
  \def\pgfpageoptionfirstshipout{1}
}
```
\begin{verbatim}
{\pgfpagesphysicalpageoptions
  \% logical pages=3,
  physical height=\pgfpageoptionheight,\%
  physical width=\pgfpageoptionwidth,\%
  current logical shipout=\pgfpageoptionfirstshipout\}%
\ifdim\paperheight >\paperwidth\relax % put side-by-side
  \pgfpageslogicalpageoptions{1}
  \%
  border shrink=\pgfpageoptionborder,\%
  border code=\beamerswitch@Border,\%
  resized width=.333\pgfphysicalwidth,\%
  resized height=.5\pgfphysicalheight,\%
  center=\ifbool{HL@beamerswitch@align}{\%
    \pgfpoint{.167\pgfphysicalwidth + 0.5\beamerswitch@margin}{.75\pgfphysicalheight - 0.5\beamerswitch@margin}\%
  }{\%
    \pgfpoint{.5\pgfphysicalwidth}{.75\pgfphysicalheight - 0.5\beamerswitch@margin}\%
  }\%
  \pgfpoint{.167\pgfphysicalwidth}{.75\pgfphysicalheight}\%
  \%}
  \pgfpageslogicalpageoptions{2}
  \%
  border shrink=\pgfpageoptionborder,\%
  border code=\beamerswitch@Border,\%
  resized width=.333\pgfphysicalwidth,\%
  resized height=.5\pgfphysicalheight,\%
  center=\ifbool{HL@beamerswitch@align}{\%
    \pgfpoint{.5\pgfphysicalwidth}{.75\pgfphysicalheight - 0.5\beamerswitch@margin}\%
  }{\%
    \pgfpoint{.833\pgfphysicalwidth - 0.5\beamerswitch@margin}{.75\pgfphysicalheight - 0.5\beamerswitch@margin}\%
  }\%
  \pgfpoint{.5\pgfphysicalwidth}{.75\pgfphysicalheight}\%
  \%}
  \pgfpageslogicalpageoptions{3}
  \%
  border shrink=\pgfpageoptionborder,\%
  border code=\beamerswitch@Border,\%
  resized width=.333\pgfphysicalwidth,\%
  resized height=.5\pgfphysicalheight,\%
  center=\ifbool{HL@beamerswitch@align}{\%
    \pgfpoint{.833\pgfphysicalwidth - 0.5\beamerswitch@margin}\%
  }{\%
    \pgfpoint{.833\pgfphysicalwidth}{.75\pgfphysicalheight}\%
  }\%
  \pgfpoint{.833\pgfphysicalwidth}{.75\pgfphysicalheight}\%
  \%}
}\else % stack on top of one another
  \pgfpageslogicalpageoptions{1}
  \%
  border shrink=\pgfpageoptionborder,\%
  border code=\beamerswitch@Border,\%
  resized width=.5\pgfphysicalwidth,\%
\end{verbatim}
The '2 by 2' layout is similar to the normal 4 on 1 layout.
The ‘1 by 4 narrow’ layout puts four slides in a column on the left half of the page (or in a row on the top half).

\pgfpagesdeclarelayout{1 by 4 narrow}
{\edef\pgfpageoptionheight{\the\paperwidth} \% landscaped by default} 25
\edef\pgfpageoptionwidth{\the\paperheight}
\def\pgfpageoptionborder{0pt}
\def\pgfpageoptionfirstshipout{1}
}
{
\pgfpagesphysicalpageoptions
{%
logical pages=4,
physical height=\pgfpageoptionheight,%
physical width=\pgfpageoptionwidth,%
current logical shipout=\pgfpageoptionfirstshipout%
}
\ifdim\paperheight>\paperwidth\relax % put side-by-side
\pgfpageslogicalpageoptions{1}
{%
border shrink=\pgfpageoptionborder,%
border code=\beamerswitch@Border,%
resized width=.25\pgfphysicalwidth,%
resized height=.5\pgfphysicalheight,%
center=\ifbool{HL@beamerswitch@align}{%\pgfpoint{.125\pgfphysicalwidth+0.6\beamerswitch@margin}{.75\pgfphysicalheight-0.333\beamerswitch@margin}%}{%\pgfpoint{.125\pgfphysicalwidth}{.75\pgfphysicalheight}%
}%
\pgfpageslogicalpageoptions{2}
{%
border shrink=\pgfpageoptionborder,%
border code=\beamerswitch@Border,%
resized width=.25\pgfphysicalwidth,%
resized height=.5\pgfphysicalheight,%
center=\ifbool{HL@beamerswitch@align}{%\pgfpoint{.375\pgfphysicalwidth+0.2\beamerswitch@margin}{.75\pgfphysicalheight-0.333\beamerswitch@margin}%}{%\pgfpoint{.375\pgfphysicalwidth}{.75\pgfphysicalheight}%
}%
\pgfpageslogicalpageoptions{3}
{%
border shrink=\pgfpageoptionborder,%
border code=\beamerswitch@Border,%
resized width=.25\pgfphysicalwidth,%
resized height=.5\pgfphysicalheight,%
center=\ifbool{HL@beamerswitch@align}{%\pgfpoint{.625\pgfphysicalwidth-0.2\beamerswitch@margin}{.75\pgfphysicalheight-0.333\beamerswitch@margin}%}{%\pgfpoint{.625\pgfphysicalwidth}{.75\pgfphysicalheight}%
}%
\pgfpageslogicalpageoptions{4}
{%
border shrink=\pgfpageoptionborder,
The ‘2 by 3’ layout positions the slides as three rows of two slides each.
\pgfpageslogicalpageoptions{3}
{
  \pgfpageslogicalpageoptions{1}
  \pgfpageslogicalpageoptions{2}
  \pgfpageslogicalpageoptions{4}
  \pgfpageslogicalpageoptions{5}
  \pgfpageslogicalpageoptions{6}
}\else
% stack on top of one another
\pgfpageslogicalpageoptions{1}
{\pgfpageslogicalpageoptions{2}}
The ‘2 by 4’ layout positions the slides as four rows of two slides each.
9.7 Article layout

We provide some options for configuring the appearance of article mode. The \texttt{maketitle} option triggers adjustments in how the title block is printed.

One will be to join the title and subtitle with a colon. There is an edge case where, if the user provides a title that ends in ‘!’ or ‘?’ and provides a subtitle while this option is in effect, they will end up with clashing punctuation in the middle of the displayed title (‘!’: or ‘?:”). We therefore introduce a toggle that, if set true, suppresses the additional colon.

Of course, we would rather not bother the user with this, so we introduce a command for testing the title for final punctuation.

The only way I can seem to do this is by switching to expl3 syntax. Rather than introduce extra hard dependencies to cope with what will probably be quite a rare issue, we make the dependency soft: it will only be applied if \expa\ is available. If there is
demand for it, we could introduce a class option to switch this code on or off, but let’s see how we go.

\begin{verbatim}
\IfFileExists{xparse.sty}{\@tempswatrue}{\@tempswafalse}
\if@tempswa
\RequirePackage{xparse}
\ExplSyntaxOn
\NewDocumentCommand{\bsw@punct@test}{m}{\l_bsw_punct_test:n {#1}}
\cs_new_protected:Nn \l_bsw_punct_test:n
{\str_case_e:nnTF { \str_item:nn {#1} {-1} }
{ , } { }
{ ; } { }
{ : } { }
{ . } { }
{ ! } { }
{ ? } { }
}{\global\toggletrue{titlepunct}}
{\global\togglefalse{titlepunct}}
\ExplSyntaxOff
\end{verbatim}

We insert this test into the definitions for \texttt{title} introduced by \texttt{beamer/beamerarticle}.

\begin{verbatim}
\mode<article>{%
\renewcommand{\title}[2][{}]{\beamer@origtitle{#2}\bsw@punct@test{#2}}
%
\mode<presentation>{%
\long\def\beamer@title[#1]#2{%
\def\inserttitle{#2}
\def\beamer@shorttitle{#1}%
\bsw@punct@test{#2}%
}
%
}%
}
\else
\wlog{Beamerswitch: Auto-detection of title punctuation not available.}
\fi

\end{verbatim}

We offer some alternatives for handling frame titles in article mode.

- \texttt{para} is what \texttt{beamerarticle} normally does.
- \texttt{margin} puts the frame titles in the margin.
- \texttt{none} gets rid of them entirely.

\begin{verbatim}
\define@choicekey+[AL]{beamerswitch}{frametitles}{para, margin, none}{%
\def\beamerswitch@articleframetitles{#1}%
}%
\ClassWarning{beamerswitch}{Value of `frametitles' not recognized. Allowed values are para, margin, and none.}%

\end{verbatim}

We provide a command for setting these options.
The following options are mode specific.

Personally I find slide titles somewhat intrusive in article mode. They can easily end up duplicating section headings in running text, or captions in figures. You may have other ideas, so we keep this behaviour configurable.

The margin value is implemented using \marginpar.

This is where we make our adjustments to \maketitle. We start by joining the subtitle to the title by means of a colon instead of a newline.

We add support for printing the institute information.

Our first change to \maketitle itself is to remove the initial vertical space.
The other is to add in a row for the institute information.

For consistency, if the \texttt{maketitle} option has been passed, we change the PDF metadata in the other modes to use the colon convention for joining the title and subtitle.

There is no more.
Change History

v1.1
\articlelayout: Remove personal customizations from \maketitle code. .......... 37
General: Fix \maketitle and \textinst options. ....................................... 35
Improve documentation of \BeamerswitchSpawn, \handoutlayout and \\
\articlelayout. ................................................................. 5
v1.2
General: Add ‘Quick start guide’ section. ........................................ 1
v1.3
\BeamerswitchSpawn: Decompose \BeamerswitchSpawn into further configurable
macros. ................................................................. 12
v1.4
General: Make margins around slides a configurable size ........................... 15
v1.5
\beamerswitch@Border: Remove empty frames from end of handout. ............ 15
v1.6
\JobName: Fix bug triggered by spaces in job names. ............................... 12
v1.6.1
\bsw@punct@test: Update expl3 syntax ........................................... 35
v1.7
General: Add nup=1plus option. ..................................................... 14
Add nup=1 option. ......................................................... 14
Add ‘1 by 1 narrow’ layout. .................................................. 18
Add ‘1 by 1’ layout. .......................................................... 18
Index

1plus value, 8, 14–15
3plus value, 8, 14–15
4plus value, 8, 14–15

1 value, 8, 14–15
2 value, 8, 14–15
3 value, 8, 14–15
4 value, 8, 14–15
6 value, 8, 14–15
8 value, 8, 14–15

align key, 7, 15
also key, 5, 11
alsoarticle key, 5, 11
alsobeamer key, 5, 11
alsohandout key, 5, 11
alsotrans key, 5, 11
article key, 4, 10–11
\articlelayout, 8, 36–38
\ArticleSuffix, 5, 11–12

beamer key, 4, 10
\BeamerSuffix, 5, 11–12
\beamerswitch@articleframetitles, 36
\beamerswitch@Border, 15
\beamerswitch@footer, 16
\beamerswitch@handoutpaper, 14
\beamerswitch@margin, 15
\beamerswitch@nup, 17–18
\beamerswitch@pnoadjust, 16–17
\beamerswitch@SetAlso, 11
\BeamerswitchSpawn, 5, 12
border key, 7, 15
\bsw@punct@test, 35–36

frametitles key, 8, 36

handout key, 4, 10
\handoutlayout, 7, 16–18
\handoutpnoadjust, 15–16
\HandoutSuffix, 5, 11–12
\JobName, 12

Values

1plus, 8, 14–15
3plus, 8, 14–15
4plus, 8, 14–15
1, 8, 14–15
2, 8, 14–15
3, 8, 14–15
4, 8, 14–15
6, 8, 14–15
8, 8, 14–15

also, margin, 8, 36–37
alsoarticle, margin, 8, 36–37
alsobeamer, 8, 36–37
alsohandout, 8, 36–37
alsotrans, 5, 11
article, 4, 10–11
beamer, 4, 10
border, 7, 15
frametitles, 8, 36
handout, 4, 10
maketitle, 8, 35
margin, 7, 15
nohyperref, 6, 11
nup, 7, 14–15
paper, 7, 14
pnos, 7, 15–16
textinst, 7, 11
trans, 4, 10
maketitle key, 8, 35
margin key, 7, 15
margin value, 8, 36–37
nohyperref key, 6, 11
none value, 8, 36–37
nup key, 7, 14–15
paper key, 7, 14
para value, 8, 36–37
pnos key, 7, 15–16
\SpawnedCompiler, 12
\SpawnedLuaTeX, 12
\SpawnedPDFTeX, 12
\SpawnedTeX, 12
\SpawnedXeTeX, 12
textinst key, 7, 11
trans key, 4, 10
\TransSuffix, 5, 11–12

Keys

align, 7, 15
also, 5, 11
alsoarticle, 5, 11
alsobeamer, 5, 11
alsohandout, 5, 11