

BEAMER-REVEAL

Walter Daems (`walter.daems@uantwerpen.be`)

Release 2026-04-08 — v1.10

1 Introduction

BEAMER [1] is a very powerful and convenient document class to create presentations and slides. However, integrating multimedia in it, is still a bit of a faff. On one side you have the integrated multimedia facility of BEAMER [1] and on the other side the `movie15` [2] and `media9` [3] packages. But unless you use the stock acrobat reader on Microsoft Windows, these things barely work, if at all.

Next to this \LaTeX ecosystem for slides, you have the `reveal.js` framework [4], that allows easy integration of multimedia content. Why not combine both worlds? With that I mean:

- make your slides in BEAMER, using all the nice packages that come with \LaTeX ,
- include any browser-compatible multimedia content you'd like, even generate some \LaTeX animations,
- convert that presentation to the `reveal.js` framework.

That's exactly what this package does.

Note that you are an *early adopter* when using this package. Expect frequent changes.

2 Rationale

Let's talk about the elephant in the room: why not work in the `reveal.js` framework directly, e.g. using Quarto [5]? I see two reasons:

- It avoids a new learning curve, allowing you to continue to build on your \LaTeX and BEAMER expertise.
- It avoids that you have to convert your thousands of slides into a new format.

For me, those are all the reasons I need.

3 Synopsis

Figure 1 shows the overall flow. You start by making slides (frames) the usual way using the `beamer` class. Your source file (`slides.tex`) uses the package `beamer-reveal.sty` and references the multimedia files of your choice. These multimedia files (e.g. videos) can be super-imposed on any slide you like. Your favorite \LaTeX -compiler typesets your slides to PDF and produces an auxiliary `.rv1`-file containing extra information for the conversion script. The conversion script `beamer-reveal.pl` fuses the PDF and the auxiliary file into a `reveal.js` presentation, using the templates provided by the BEAMER-REVEAL package. In fact, this is done by converting your slides to JPG format and using them as the background on the REVEAL-slides. The multimedia content appears as HTML5 elements on that background.

Notes are also converted to JPG format and available in the speaker view that is based on notes plugin of `reveal.js`.

You then can use your favorite browser as your viewer instead of the good old PDF reader.

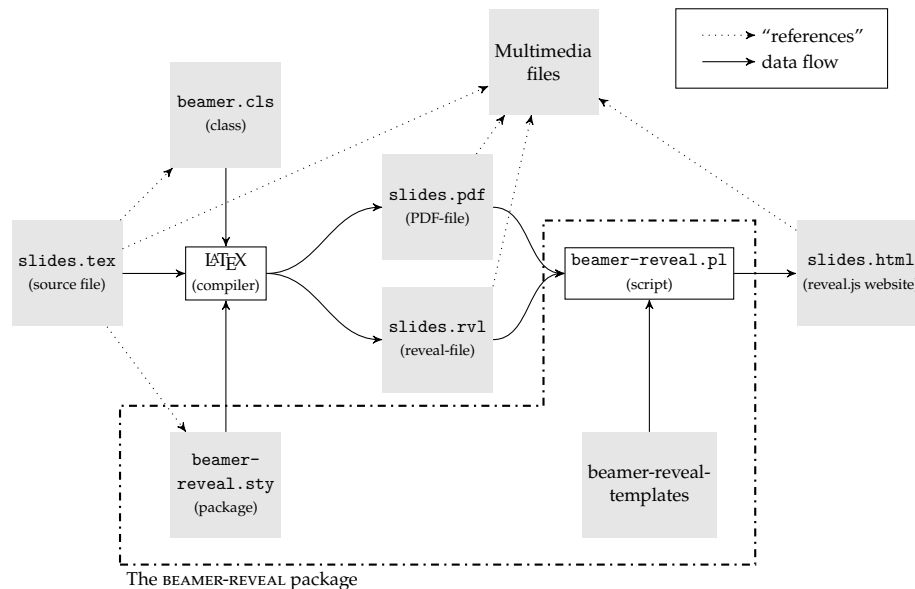


Figure 1: Workflow imposed by the BEAMER-REVEAL package

As a nice bonus, the BEAMER-REVEAL package allows you to generate \LaTeX animations. It's not on the level of Manim [6], but for shorter animations inlined in your deck of slides it is more than functional!

Finally, you can also include material in iframes. Iframes are HTML constructs that act as a mini-browser, allowing to incorporate multiple multimedia blocks in HTML. As an example, this allows incorporating asymptote material that has been generated for displaying as HTML using WebGL. Note that in many cases, your browser will prohibit these iframes to load external web content. To solve this, you can run a local webserver and access your presentation through it, or you can embed the iframe in the main HTML file. This is explained in Section 7.

4 Quirks

Combining BEAMER and REVEAL posed one major challenge: how to make sure that the HTML5 elements appear exactly where you want them to be, i.e. perfectly aligned with your \LaTeX content as it has been converted to PDF. The core of the problem is twofold:

- In BEAMER the aspectratio of the slides is determined by the class option `aspectratio`. Your PDF viewer uses letterboxing (black bars on the side) if the aspectratio of your presentation does not correspond to the aspectratio of your screen. To the other hand, REVEAL puts your slides fullscreen without letterboxing, and therefore the aspectratio is determined by the screen resolution.
- Given the vector-nature of \LaTeX and PDF, resolution is not a parameter you normally care about (everything is vector graphics anyway), while given the pixel-based nature of traditional multimedia files, resolution is an issue that you need to think about carefully.

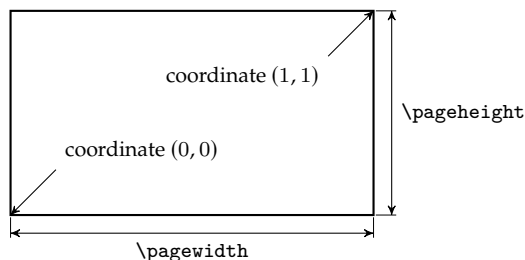
There is only one good solution: ask the users to (1) set the correct aspectratio using the beamer class options and (2) specify the width or height of their displaying screen using a BEAMER-REVEAL class option.

The latter ensures that (a) the background content and the multimedia files that are generated based on the \LaTeX source, are generated with sufficient resolution, and that (b) the file sizes stay within acceptable limits.

Actually, there is a one more problem, and that is that the canvas size and aspectratio of the display area in your browser is dependent on whether you are viewing full screen or not. Therefore, an extra requirement arises: (3) when presenting a slide-deck only you must use your browser in full-screen mode. Otherwise, the alignment is off.

In order not to have to care about resolutions or the actual `\pagewidth` or `\pageheight` of your presen-

tation, I've chosen to impose another constraint: (4) if you want to add multimedia material in overlay mode, the user must specify locations on the screen (where to put the multimedia content) as relative fractions (x, y) where $(0, 0)$ corresponds to the bottom left of the screen and $(1, 1)$ corresponds to the top right of the screen.



However, if you have to specify width and heights of the multimedia boxes that you want to overlay on your slides, you don't want to specify width and height as relative numbers (of the slide width and height). In that scenario, displaying a 16:9 video on a 16:10 slide with a width equal to a quarter of the slide width, would require you to specify: `width=0.25,height=0.225`, which is weird. In addition, if you'd have to present at a venue where the projector has a different aspectratio from the one you anticipated, you would be forced to recalculate all the widths and heights of every video or image. You don't want that. Presenting on itself already brings sufficient stress without that extra worry.

Therefore, I imposed a fifth constraint: (5) the user must always specify either width or height of the box (as a fraction of the respective slide width or height) together with the aspectratio. Given width and aspectratio the BEAMER-REVEAL package can correctly determine its relative height; likewise height and aspectratio can be converted to the correct relative width. Therefore, it is illegal to specify both width and height at the same time. As the package knows the aspectratio of the screen (set correctly by the user) no recalculations need to be done. In the example above, that would mean specifying: `width=0.25,aspectratio=16/9`, which is very logical.

When confronted with the situation where you need to present on an old 4:3 projector, instead of the 16:10 you are used to, this allows you to just change the beamer aspectratio to 43, recompile, rerun `beamer-reveal.pl` and you are good to go on stage.

So summarized, these are the five rules to go by:

1. Set the correct aspectratio as a beamer class option.
2. Specify the X or Y-resolution of your displaying screen as a package option to the BEAMER-REVEAL package.
3. Always put your browser in full-screen-mode when presenting.
4. Specify positions relative to the slide width and height, $(0, 0)$ being bottom left and $(1, 1)$ being top right.
5. Specify box sizes of the HTML5 content always as the width or length in combination with the aspectratio.

One final remark: you can put multimedia boxes on your slides in two modes:

overlay mode in this mode the box is put on your slide like tikz puts an image in overlay mode on a document: it does not consume any space on the slide;

insert mode in this mode the box is actually typeset by L^AT_EX on your slide. The actual content (video, image, ...) will not be there, but the space will be consumed.

Note that for both modes — to work properly — your compiler must be run twice before running `beamer-reveal.pl`.

5 Portability

These class files should be ready to use with all common modern \LaTeX compilers that produce PDF (pdf \LaTeX , Xe \LaTeX , Lua \LaTeX , ...) from the major \TeX -distributions (TeX \TeX , TexLive, MikTeX). If you experience problems with one of these, please inform the author.

The script `beamer-reveal.pl` is a Perl script, that works on all major platforms (UNIX, Linux, BSD, Debian, MS-Windows, MAC-OS, ...). It makes use of the Poppler library and its `pdftoppm` tool, which is also available for those platforms. In case you want to use \LaTeX animations, it also uses your very own favorite \LaTeX -compiler, `pdfcrop` (which is part of the major \TeX -distributions) and `ffmpeg` a well-known video conversion tool.

If these tools are available on your platform, then all should be well.

6 Installing the `beamer-reveal.pl` script

6.1 On Linux-like operating systems

Open a terminal with a shell. Below '\$' represents your shell prompt. If needed, update your package list first. Then install the required tools:

- on a Debian-like OS:

```
$ sudo apt install -y perl cpanminus poppler-utils \
  texlive-extra-utils ffmpeg
```
- on a Redhat-like OS:

```
$ sudo dnf install -y epel-release
$ sudo dnf install -y \
  https://download1.rpmfusion.org/free/el/rpmfusion-free-release-$(rpm -E %rhel).noarch.rpm
$ sudo dnf install -y perl perl-App-cpanminus poppler-utils \
  texlive-pdfcrop ffmpeg
```
- on openSUSE:

```
$ sudo zypper install -y perl perl-App-cpanminus \
  poppler-tools texlive-pdfcrop ffmpeg
```
- on an Arch-like OS:

```
$ sudo pacman -Syu --needed perl cpanminus poppler \
  texlive-binextra ffmpeg
```
- on Alpine:

```
$ sudo apk add perl cpan-app-cpanminus poppler-utils \
  texlive-extra-utils ffmpeg
```
- on macOS:

```
$ brew install perl cpanminus poppler texlive ffmpeg
```

6.2 On MS windows operating systems

Open a powershell. Below '\$' represents the powershell prompt. Install the required tools as follows:

```
$ winget install StrawberryPerl.StrawberryPerl
$ winget install -e --id oschwartz10612.Poppler
$ winget install MiKTeX.MiKTeX
$ winget install Gyan.FFMpeg
$ winget install --id Python.Python.3 --source winget
```

You probably need to answer 'Yes' quite some times. If you're asked to restart, you only need that before moving over to section 6.4 'Checking your setup'. You probably already have MiKTeX installed. In that case you can skip that. However, make sure the 'pdfcrop' program (part of MiKTeX) is installed and available on your path.

6.3 Install BEAMER-REVEAL

Open a shell on Linux/macOS, or a powershell prompt on MS-Windows. Run the following command at the prompt:

```
$ cpanm BeamerReveal
```

This will take a while as this needs to install a number of extra Perl libraries that have been used in BEAMER-REVEAL. You can use the same command to update the script if a new release has been made.

The number of the Perl package is: 20260408.1240

You will see that number appear during the installation. Any version more recent version number than this one will work. New releases will fix bugs that are reported. If there is a breaking change in the Perl package, a new CTAN update will be released as well.

6.4 Checking your setup

Open a shell on Linux/macOS, or a command line or powershell prompt on MS-Windows. Then see whether you can invoke the help information for the tools.

```
$ cpanm -h
$ pdftoppm -h
$ pdfcrop --help
$ ffmpeg -h
```

To test BEAMER-REVEAL on MS-Windows, run:

```
$ beamer-reveal -h
```

On any other platform, run:

```
$ beamer-reveal.pl -h
```

If they all display correct help info, you're good to go.

7 Using the beamer-reveal.pl script

If you prepared your BEAMER-presentation according to Section 12, then converting it into a reveal.js HTML presentation, is as simple as running the following command. Make sure the current directory of the shell is the directory your L^AT_EX-source file and your compiled PDF-file is in. For all commands below, stay in that same working directory!

If your document is called `jobname.tex`, then just run:

```
$ lualatex jobname.tex
$ lualatex jobname.tex
```

On MS-Windows, run:

```
$ beamer-reveal jobname
```

On any other platform, run:

```
$ beamer-reveal.pl jobname
```

Conversion is very fast. Of course, if you need to render some L^AT_EX animations, it may take a little more than a jiffy. Especially if you are working on MS-Windows, because there the generation is fully single-threaded.

Some side notes:

- you need to run `beamer-reveal.pl` from the same working directory as your L^AT_EX-compiler; the script needs to be able to access your source files from the same viewpoint as your compiler.
- `beamer-reveal.pl` also has some convenient command-line options to set the `aux` directory (that contains your `.rvl` file), the `pdf` directory that is the output directory of your L^AT_EX-run and the

output directory (in which it will place the resulting reveal site). This allows easy integration with some build frameworks such as `latexmk`. Checkout the help message to learn about it.

```
$ beamer-reveal.pl -h
```

Next, you can load your document in your browser, e.g.:

```
$ firefox jobname.html
```

If your presentation contains `iframe` content, you need to start a local webserver. You don't need a network connection for that. This is how:

```
$ python -m http.server
```

Then, you can access it through: <http://localhost:8000>.

As of v1.06 there is an alternative, and that is to make the `iframe` element 'embedded'. This will cause the `iframe` content to live as Base64-encoded ASCII string in the main `\jobname.html` file. Therefore it will be trusted by your browser and will load without any complaint.

8 Demo

If you want to see and try out the result of the example that is part of this documentation, check:

- a 16-by-9 version on: <https://www.digmanwaves.net/beamer-reveal/169>
- a 16-by-10 version on: <https://www.digmanwaves.net/beamer-reveal/1610>

9 Open issues

Below, you'll find a list of open issues. I appreciate any input I can get on these matters.

- Regarding browsers, I'm a big fan of Firefox. However, I must admit that the Chrome/chromium line does a better job regarding smooth slide transitions. I can't seem to get preloading to work in Firefox browser, while in Chrome/chromium, it just works out of the box.
- I did not yet see any solutions to perform robust letterboxing on a browser (i.e. putting black stripes on the top and bottom if you are displaying a 16:9 presentation on a 16:10 screen). With robust, I mean a solution in which you don't have to change the content of your presentation html-file.

10 Reach out

Is there any feature you are missing? Some problems you encounter? Some inconsistencies in the interface or the documentation? Some additional features that Reveal supports, but are not in BEAMER-REVEAL? Let me know by dropping me an e-mail.

If you think BEAMER-REVEAL makes no sense, let me know why you think so. I'm keen to learn.

On the other hand, if you like BEAMER-REVEAL and are using it, just send me a kind word. It keeps me going much better than wine or pizza.

11 Thanks

Thanks to Paul Levrie for proofreading this documentation and testing the package.

12 Usage

12.1 Package options

12.1.1 Regarding dimensions

The following package options are available:

<code>width</code>	the width (in pixels) of the screen you will display the presentation on
<code>height</code>	the height (in pixels) of the screen you will display the presentation on

Only specify one of the two options. The other dimension will be deduced from the `aspectratio` option that passed onto the beamer class.

Higher values will give higher resolution of the slides (in the background), but also larger file sizes. A safety factor is already used when converting the slides to jpg-format, so taking the true height or width is recommended. Only when you are bothered with jpg-artifacts in the final result, you should consider increasing the width- or height-value.

12.1.2 Reveal specific stuff

The following options are opening up some cool stuff that reveal enables:

<code>embed</code>	this causes all graphical material in your slides (<code>.jpg</code> , <code>.png</code> , <code>.mp4</code> , <code>.ogg</code> , <code>.wav</code> , ...) including the slide backgrounds and the notes slides to be embedded as Base64-encoded ASCII string into the HTML. Note however that this still is not the case for animations, stills and the <code>reveal.js</code> boilerplate javascripts.
<code>autoslide=<int></code>	This causes your presentation to progress automatically, going from slides to slide every <code><int></code> milliseconds. You can override this value on an individual basis per slide, by providing the same option to the beamer frame environment.
<code>loop</code>	This causes your presentation to loop; this makes sense if you <i>autoslide</i> it, enabling to run your presentation in <i>kiosk</i> mode.

12.2 An enhanced frame environment

`frame (env.)` The frame environment is defined by `BEAMER`. However, it is equipped with four new environment options by the `BEAMER-REVEAL` package:

<code>titleslide</code>	specifies that this slide is a title slide. It will be on the top level of the reveal menu. This menu can be invoked by pressing 'm' in your presentation (or clicking on the pan-cake icon on the bottom left of your presentation).
<code>sectionslide</code>	specifies that this slide is a section slide. It will be on the second level of the reveal menu.
<code>subsectionslide</code>	specifies that this slide is a subsection slide. It will be on the third level of the reveal menu.
<code>transition</code>	one of <code>none</code> (default), <code>fade</code> , <code>zoom</code> , <code>concave</code> or <code>convex</code> ; these correspond to the available <code>reveal.js</code> transitions. Not recommended for use. Why? Animation without purpose is bad practice.
<code>embed</code>	This will cause the slide background, the notes and all media elements on your slide to be embedded as a Base64-encode ASCII string in your HTML file.
<code>autoslide=<int></code>	This will set the individual time for this slide to <code><int></code> milliseconds. This option only takes effect if you set the package option <code>autoslide</code> as well. If you set the value to 0, the presentation will stop the <code>autoslide</code> mode at that

slide. If your slide contains a video or audio fragment, or an animation that plays for a specific amount of time, then this time will override the value you specify here. The slide show will only progress as soon as the animation or video/audio has finished!

Slides that are not titleslides, sectionslides or subsectionslides are ordinary slides. They will appear on the lowest level in the reveal menu.

12.3 The macros to use inside the frame environment

To understand the operation of the macros, it is important to realize that the slide content of your beamer-generated PDF will be put as a background image onto the reveal.js slides. The extra material such as videos, images and audio will be put as an overlay on top of that background. The macros allow you to define what material to put in overlay and where and how big it should appear. All macros take options. Some of the options need a value, some not. Options that take no value are marked with a dagger-symbol in superscript (†).

`\video` This macro will create a video box on the current slide. Of course a video box can cover the entire slide if desired.

Its syntax is:

- overlay mode:
`\video<overlay-spec>[options] (nodename) \at (x,y) { filename }`
- insert mode:
`\video<overlay-spec>[options] (nodename) { filename }`

The arguments are:

`overlay-spec` a standard beamer overlay specification that allows you to determine on which overlays the video is to appear; this argument will end up in a traditional `\only<>{ }` clause that beamer provides.

`options` the following keys are available. In general they require a value: `key=value`.

Size options:

<code>width</code>	the width of the video box (a fraction relative to the width of the slide)
<code>height</code>	the height of the video box (a fraction relative to the height of the slide)
<code>aspectratio</code>	the aspectratio of the video box
Remember that you never specify both <code>width</code> and <code>height</code> , only one of those two in combination with the <code>aspectratio</code> .	

Placement options:

<code>anchor</code>	value is one of center, north, west, south, east, north east, north west, south east, south west; this specifies where the anchor of the video is positioned; the anchor will be positioned at (x,y)
<code>above</code> [†]	synonym to <code>anchor=south</code>
<code>below</code> [†]	synonym to <code>anchor=north</code>
<code>left</code> [†]	synonym to <code>anchor=west</code>
<code>right</code> [†]	synonym to <code>anchor=east</code>
<code>above left</code> [†]	synonym to <code>anchor=south east</code>
<code>above right</code> [†]	synonym to <code>anchor=south west</code>
<code>below left</code> [†]	synonym to <code>anchor=north east</code>
<code>below right</code> [†]	synonym to <code>anchor=north west</code>

Appearance and sound options:

<code>fit</code>	the way the video should occupy the box: <code>fill</code> , <code>cover</code> or <code>fit</code>
<code>background</code>	the color of the background of the box
<code>draw</code> [†]	generates an outline around the box that allows you to inspect where the video will end up in your PDF-file
<code>autoplay</code> [†]	causes the video to start playing as soon as it appears on the slide
<code>controls</code> [†]	causes player controls to appear below your video
<code>loop</code> [†]	causes the video to run in loop mode
<code>muted</code> [†]	silences the audio of the player

Various options (only one for now):

<code>embed</code> [†]	causes the content of the video-file to be embedded as a Base64-encoded ASCII string in the main html file the script produces. I don't recommend this for video files.
---------------------------------	---

`nodename` (optional) name to assign to the rectangular node corresponding to the bounding box of the video, such that you can later refer to that node (and its derived anchors (e.g. as `'(nodename.south)'`) to position other material.

`x, y` the x - and y -coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

`filename` a filename that leads to the video file (e.g. an mp4 file). Any file playable by your browser will work. If the filename starts with `http://` or `https://`, it will be considered to be a URL and your presentation will link to it.

Note that you can pause and (re)start your video by pressing 'v' when presenting your slide deck.

`\audio` This macro will create an audio block on the current slide. This block is rather an abstract concept, unless you activate the controls of the player. Indeed, the audio block will be invisible unless you specify the option to display the controls of the player.

Its syntax is:

- overlay mode:
`\audio<overlay-spec>[options] (nodename) \at (x,y) { filename }`
- insert mode:
`\audio<overlay-spec>[options] (nodename) { filename }`

The arguments are:

`overlay-spec` a standard beamer overlay specification that allows you to determine on which overlays the audio is to appear; this argument will end up in a traditional `\only<>{}` clause that beamer provides.

`options` the following keys are available. In general they require a value: `key=value`.

Size options:

<code>width</code>	the width of the audio box (a fraction relative to the width of the slide)
<code>height</code>	the height of the audio box (a fraction relative to the height of the slide)
<code>aspectratio</code>	the aspectratio of the audio box

Placement options:

<code>anchor</code>	value is one of <code>center</code> , <code>north</code> , <code>west</code> , <code>south</code> , <code>east</code> , <code>north east</code> , <code>north west</code> , <code>south east</code> , <code>south west</code> ; this specifies where the anchor of the audio box is positioned; the anchor will be positioned at (x, y)
<code>above</code> [†]	synonym to <code>anchor=south</code>

<code>below[†]</code>	synonym to <code>anchor=north</code>
<code>left[†]</code>	synonym to <code>anchor=east</code>
<code>right[†]</code>	synonym to <code>anchor=west</code>
<code>above left[†]</code>	synonym to <code>anchor=south east</code>
<code>above right[†]</code>	synonym to <code>anchor=south west</code>
<code>below left[†]</code>	synonym to <code>anchor=north east</code>
<code>below right[†]</code>	synonym to <code>anchor=north west</code>

Appearance and sound options:

<code>fit</code>	the way the audio block should occupy the box: <code>fill</code> , <code>cover</code> or <code>fit</code>
<code>background</code>	the color of the background of the box
<code>draw[†]</code>	generates an outline around the box that allows you to inspect where the audio box will end up in your PDF-file
<code>autoplay[†]</code>	causes the audio to start playing as soon as it appears on the slide
<code>controls[†]</code>	causes player controls to appear
<code>loop[†]</code>	causes the audio to run in loop mode
<code>muted[†]</code>	silences the audio of the player

Various options (only one for now):

<code>embed[†]</code>	causes the content of the audio-file to be embedded as a Base64-encoded ASCII string in the main html file the script produces. I don't recommend this for audio files.
--------------------------------	---

`nodename` (optional) name to assign to the rectangular node corresponding to the bounding box of the audio, such that you can later refer to that node (and its derived anchors (e.g. as `'(nodename.south)'`) to position other material.

`x, y` the *x*- and *y*-coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

`filename` a filename that leads to the audio file (e.g. an mp4 file). Any file playable by your browser will work. If the filename starts with `http://` or `https://`, it will be considered to be a URL and your presentation will link to it.

Note that you can pause and (re)start your audio fragment by pressing 'a' when presenting your slide deck.

`\iframe` This macro will create an iframe box on the current slide. Of course an iframe box can cover the entire slide if desired.

Its syntax is:

- overlay mode:
`\iframe<overlay-spec>[options] (nodename) \at (x,y) { filename }`
- insert mode:
`\iframe<overlay-spec>[options] (nodename) { filename }`

The arguments are:

`overlay-spec` a standard beamer overlay specification that allows you to determine on which overlays the iframe is to appear; this argument will end up in a traditional `\only<>{}` clause that beamer provides.

`options` the following keys are available. In general they require a value: `key=value`.

Size options:

<code>width</code>	the width of the iframe box (a fraction relative to the width of the slide)
--------------------	---

height the height of the iframe box (a fraction relative to the height of the slide)

aspectratio the aspectratio of the iframe box

Remember that you never specify both width and height, only one of those two in combination with the aspectratio.

Placement options:

anchor value is one of center, north, west, south, east, north east, north west, south east, south west; this specifies where the anchor of the video is positioned; the anchor will be positioned at (x,y)

above synonym to anchor=south;

below synonym to anchor=north;

left[†] synonym to anchor=west;

right[†] synonym to anchor=west;

above left[†] synonym to anchor=south east;

above right[†] synonym to anchor=south west;

below left[†] synonym to anchor=north east;

below right[†] synonym to anchor=north west;

Appearance options:

fit the way the iframe should occupy the box: fill, cover or fit

background the color of the background of the box

draw generates an outline around the box that allows you to inspect where the iframe will end up in your PDF-file;⁰

Various options (only one for now):

embed[†] causes the content of the iframe html-file to be embedded as a Base64-encoded ASCII string in the main html file the script produces. For iframe material, I think this makes sense if you want to avoid using an auxiliary webbrowser (e.g. if you are presenting using someone else's computer).

nodename (optional) name to assign to the rectangular node corresponding to the bounding box of the iframe, such that you can later refer to that node (and its derived anchors (e.g. as '(nodename.south)') to position other material.

x, y the x- and y-coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

filename a filename that leads to the iframe file (e.g. an HTML file generated by asymptote). Note that for the content to work, you might need to serve your presentation through a local html server:

```
$ python -m http.server
```

If the filename starts with http:// or https://, it will be considered to be a URL and your presentation will link to it.

`\image` This macro will create an image box on the current slide. Of course an image box can cover the entire slide if desired.

Note that using the `\includegraphics` command of the `graphicx` package is still preferred to include the standard image formats, such as PDF, PNG, TIFF and JPG. However, the `image` command also allows to include (animated) GIFs, webp and SVG files.

Its syntax is:

- overlay mode:
`\image<overlay-spec>[options] (nodename) \at (x,y) { filename }`

- insert mode:
`\image<overlay-spec>[options] (nodename) { filename }`

The arguments are:

overlay-spec a standard beamer overlay specification that allows you to determine on which overlays the image is to appear; this argument will end up in a traditional `\only<>{}` clause that beamer provides.

options the following keys are available. In general they require a value: `key=value`.

Size options:

<code>width</code>	the width of the image box (a fraction relative to the width of the slide)
<code>height</code>	the height of the image box (a fraction relative to the height of the slide)
<code>aspectratio</code>	the aspectratio of the image box

Remember that you never specify both `width` and `height`, only one of those two in combination with the `aspectratio`.

Placement options:

<code>anchor</code>	value is one of center, north, west, south, east, north east, north west, south east, south west; this specifies where the anchor of the image is positioned; the anchor will be positioned at (x,y)
<code>above[†]</code>	synonym to <code>anchor=south</code> ;
<code>below[†]</code>	synonym to <code>anchor=north</code> ;
<code>left[†]</code>	synonym to <code>anchor=west</code> ;
<code>right[†]</code>	synonym to <code>anchor=east</code> ;
<code>above left[†]</code>	synonym to <code>anchor=south east</code> ;
<code>above right[†]</code>	synonym to <code>anchor=south west</code> ;
<code>below left[†]</code>	synonym to <code>anchor=north east</code> ;
<code>below right[†]</code>	synonym to <code>anchor=north west</code> ;

Appearance options:

<code>fit</code>	the way the image should occupy the box: fill, cover or fit
<code>background</code>	the color of the background of the box
<code>draw</code>	generates an outline around the box that allows you to inspect where the image will end up in your PDF-file; ⁰

Various options (only one for now):

<code>embed[†]</code>	causes the content of the image-file to be embedded as a Base64-encoded ASCII string in the main html file the script produces. I don't recommend this for image files.
--------------------------------	---

nodename (optional) name to assign to the rectangular node corresponding to the bounding box of the image, such that you can later refer to that node (and its derived anchors (e.g. as `'(nodename.south)'`) to position other material.

x, y the *x*- and *y*-coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

filename a filename that leads to the image file (e.g. a GIF file). Any file displayable by your browser will work. If the filename starts with `http://` or `https://`, it will be considered to be a URL and your presentation will link to it.

`\animation` This macro will create an animation box on the current slide. Different from the other boxes, this box will determine its own width and height, based on the dimensions of the LaTeX content embedded in it. In fact, it is illegal to specify `aspectratio`, `width` or `height`.

The animation is generated as follows: the content of macro will be written by the `beamer-reveal.pl` script to a separate LaTeX file using the `standalone` class. The preamble that it uses will be the part of the preamble in your beamer source file, in between the loading of the `beamer-reveal` package and the line containing `\begin{document}`. The animation block will be embedded in a loop that will be executed `duration \cdot framerate` times, providing a macro `\progress` that contains a fraction that goes up from 0 (first iteration) to 1 last iteration. The PDF-file that is generated with this standalone file, is converted to an mp4-file that is included as a video on your slide.

The tools used for this are the LaTeX-compiler you used for your beamer sourcefile, `pdfcrop` [7], `pdftoppm` [8] and `ffmpeg` [9].

This animation generation takes advantage of the multicore nature of your computer, by simple, but smart parallelization (on non-MS-Windows operating systems).

Its syntax is:

- overlay mode:
`\animation<overlay-spec>[options] (nodename) \at (x,y) { content }`
- insert mode:
`\animation<overlay-spec>[options] (nodename) { content }`

The arguments are:

`overlay-spec` a standard beamer overlay specification that allows you to determine on which overlays the animation is to appear; this argument will end up in a traditional `\only<>{}` clause that beamer provides.

`options` the following keys are available. In general they require a value: `key=value`.

Size options: no size options; size is determined from the LaTeX code itself!

Generation options:

<code>framerate</code>	number of frames per second that the animation should contain.
<code>duration</code>	duration (in seconds) of the animation
<code>progress</code>	tokenlist shaped as <code>{a handout:n1/b1,n2/b2,...}</code> with a the progress value for the PDF, and <code>b1, b2, ...</code> the progress values for handouts <code>n1, n2, ...</code> . This assumes that the <code>overlay-spec</code> of the <code>\animation</code> macro also specifies to generate the required handout slide numbers. Take a look at the example in this manual for more clarity on this. If the <code>handout:</code> section is missing, the <code>pdf progress</code> value will be used for the/all handout(s).
<code>expand once</code>	will cause the content to be expanded once; use this if you described your animation in a macro and want to expand that macro.

The number of frames that will be generated for the animation is:

$$\text{number-of-frames} = \text{framerate} \cdot \text{duration} \quad (1)$$

Your PDF file will contain a single shot of the animation (as if it were a preview shot). For this frame, the value of the `\progress` macro will be set to `pdf progress`.

Placement options:

<code>anchor</code>	value is one of <code>center</code> , <code>north</code> , <code>west</code> , <code>south</code> , <code>east</code> , <code>north east</code> , <code>north west</code> , <code>south east</code> , <code>south west</code> ; this specifies where the anchor of the animation is positioned; the anchor will be positioned at (x,y)
<code>above[†]</code>	synonym to <code>anchor=south</code>

<code>below[†]</code>	synonym to <code>anchor=north</code>
<code>left[†]</code>	synonym to <code>anchor=east</code>
<code>right[†]</code>	synonym to <code>anchor=west</code>
<code>above left[†]</code>	synonym to <code>anchor=south east</code>
<code>above right[†]</code>	synonym to <code>anchor=south west</code>
<code>below left[†]</code>	synonym to <code>anchor=north east</code>
<code>below right[†]</code>	synonym to <code>anchor=north west</code>

Appearance options:

<code>background</code>	the color of the background of the box
<code>draw[†]</code>	generates an outline around the box that allows you to inspect where the video will end up in your PDF-file
<code>autoplay[†]</code>	causes the video to start playing as soon as it appears on the slide
<code>controls[†]</code>	causes player controls to appear below your video
<code>loop[†]</code>	causes the animation to run in loop mode

`nodename` (optional) name to assign to the rectangular node corresponding to the bounding box of the animation, such that you can later refer to that node (and its derived anchors (e.g. as `'(nodename.south)'`) to position other material.

`x, y` the x - and y -coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

`content` the LaTeX code that generates every frame based on the `\progress` 'time variable'. Any macros used in this content need to be defined in the content itself or in the preamble of your slide deck in between the `\usepackage{beamer-reveal}` and the `\begin{document}` as this portion is also used when generating the animation. If that is not the case, you can specify an extra option `expand` once to perform a single expansion, such that the macros and environments on the top level are expanded once. The example contained in this manual shows you an example of this.

Note that you can pause and (re)start your animation fragment by pressing 'v' (as it is in fact a video) when presenting your slide deck.

`\still` This macro will create the still (of an animation) box on the current slide. The idea is that you can split your animation in parts and separate these parts with a slide (or a subframe, if you are using multiple overlays) containing a still corresponding to the point between the animation parts. Just as the animation box, this box will determine its own width and height, based on the dimensions of the LaTeX content embedded in it. In fact, it is illegal to specify `aspectratio`, `width` or `height`.

The still is generated as follows: the content of macro will be written by the `beamer-reveal.pl` script to a separate LaTeX file using the `standalone` class. The preamble that it uses will be the part of the preamble in your beamer source file, in between the loading of the `beamer-reveal` package and the line containing `\begin{document}`. The animation block will be embedded in an environment that sets the macro `\progress` to a value in between 0 and 1. The PDF-file that is generated with this standalone file, is converted to a `jpg`-file that is included as a still image on your slide.

The tools used for this are the \LaTeX -compiler you used for your beamer sourcefile, `pdfcrop` [7] and `pdftoppm` [8].

Its syntax is:

- overlay mode:
`\still<overlay-spec>[options] (nodename) \at (x,y) { content }`
- insert mode:
`\still<overlay-spec>[options] (nodename) { content }`

The arguments are:

`overlay-spec` a standard beamer overlay specification that allows you to determine on which

overlays the still is to appear; this argument will end up in a traditional `\only<>{}` clause that beamer provides.

options the following keys are available. In general they require a value: `key=value`.

Size options: no size options; size is determined from the \LaTeX code itself!

Generation options:

`progress` tokenlist shaped as `{a|handout:n1/b1,n2/b2,...}` with `a` the progress value for the PDF, and `b1`, `b2`, ... the progress values for handouts `n1`, `n2`, This assumes that the overlay-spec of the `\animation` macro also specifies to generate the required handout slide numbers. Take a look at the example in this manual for more clarity on this. If the `handout:` section is missing, the `pdf progress` value will be used for the/all handout(s).

`expand once` will cause the content to be expanded once; use this if you described your still in a macro and want to expand that macro.

Your PDF file will contain the image typeset by your \LaTeX run. In the reveal version of the presentation, the image will be overlaid by the generated still, such that it aligns perfectly with a corresponding video on an adjacent slide.

Placement options:

`anchor` value is one of center, north, west, south, east, north east, north west, south east, south west; this specifies where the anchor of the animation is positioned; the anchor will be positioned at (x,y)

`above`[†] synonym to `anchor=south`

`below`[†] synonym to `anchor=north`

`left`[†] synonym to `anchor=east`

`right`[†] synonym to `anchor=west`

`above left`[†] synonym to `anchor=south east`

`above right`[†] synonym to `anchor=south west`

`below left`[†] synonym to `anchor=north east`

`below right`[†] synonym to `anchor=north west`

Appearance options:

`background` the color of the background of the box

`draw`[†] generates an outline around the box that allows you to inspect where the video will end up in your PDF-file

`nodename` (optional) name to assign to the rectangular node corresponding to the bounding box of the animation, such that you can later refer to that node (and its derived anchors (e.g. as `'(nodename.south)'`) to position other material.

`x,y` the x - and y -coordinate of the box, specified as a fraction of the slide width and slide height (dimensionless), fixing the anchor of the box, w.r.t. the bottom left of your slide.

`animation-LaTeX-content` the \LaTeX code that generates every frame based on the `\progress` 'time variable'. Any macros used in this content need to be defined in the content itself or in the preamble of your slide deck in between the `\usepackage{beamer-reveal}` and the `\begin{document}` as this portion is also used when generation the still. If that is not the case, you can specify an extra option `expand once` to perform a single expansion, such that the macros and environments on the top level are expanded once. The example contained in this manual shows you an example of this.

`\voiceover` This macro makes the `beamer-reveal.pl` script send the text of the main argument to a text-to-speech engine. The macro itself inserts an autoplay audio block on the slide that will autoplay this audio file generated by the text-to-speech engine. Using `voiceover` commands and the autoplay feature for the slides (for every slide) enables you to make a kiosk-style video-like presentation that can run autonomously.

Its syntax is:

```
\voiceover<overlay-spec>[options] { text-to-read-out-loud }
```

Currently no specific options are available, but that may change in the future. For now, the option list is just passed on to the audio block that is generated.

In order to make use of one of the text-to-speech engines, you need to set it up before your first invocation of the `\voiceover` command, using all of the three macros below. You may call these macros at any time in your slide deck to change the settings.

`\voiceovertts` Use this macro to select the script that will invoke text-to-speech service. You can choose:

- `awspolly.pl`: for the Amazon engine [10]
- `azuretts.py`: for the Microsoft engine [11]
- `balabolka.pl`: for the free MS-windows engine [12]
- `festival.pl`: for the free UNIX/Linux engine by the University of Edinburgh [13]

Scripts to access these services are part of the Perl package. Consult the documentation of these services to set them up. There is no default, as I don't want to push anyone towards using a particular service.

`\setvoiceover` Use this macro to select the voice to use. Invoke the `tts` script to see how the voice should be formatted. There is no default, as I don't want to push anyone towards using a particular service, or voice.

`\setvoiceoverext` This macro sets the audio file format that will be used. Examples are:

- `.ogg`
- `.wav`
- `.mp3`

There is no default, as I don't want to push anyone towards using a particular audio format.

12.4 Short overview of the options for the different media macros

Most of the macros have very similar options, but some are not common to all of them. Therefore, I made a small overview. I admit, mostly to help myself to keep the overview while developing the package.

	Size		Placement		Appearance and sound					Various			Generation			
	w	h / ar	anchor	fit	background	draw	autoplay	controls	loop	muted	embed	framerate	duration	progress	expand	once
Macro																
<code>\video</code>	• ¹		•	•	•	•	•	•	•	•	•	×	×	×	×	×
<code>\audio</code>	• ¹		•	•	•	•	•	•	•	•	•	×	×	×	×	×
<code>\iframe</code>	• ¹		•	•	•	•	×	×	×	×	•	×	×	×	×	×
<code>\image</code>	• ¹		•	•	•	•	×	×	×	×	•	×	×	×	×	×
<code>\animation</code>	×		•	×	•	•	•	•	•	×	•	•	•	•	•	•
<code>\still</code>	×		•	×	•	•	×	×	×	×	•	×	×	•	•	•

¹ aspectratio and width, or aspectratio and height, never width and height together

13 Example

13.1 Using the example

An example will allow you to get an idea of the convenience of the package. The following steps will allow you to observe it in your browser:

```
$ lualatex beamer-reveal-example.tex
$ beamer-reveal.pl beamer-reveal-example
$ python -m http.server
```

Then open in your browser: **localhost:8000** and enjoy!

13.2 The source code of the example

```
<*example>
\documentclass[11pt,aspectratio=169,t]{beamer}%
\setbeamertemplate{navigation symbols}{}

\usepackage[width=1920,autoslide=4000]{beamer-reveal}
\usepackage{tikz}
\usepackage{siunitx}

\newcommand\eu{\mathrm{e}}
\newcommand\ju{\mathrm{j}}

\title{Test slide deck}
\subtitle{\textsc{beamer-reveal}}
\author{Walter Daems}

\setbeamertemplate{note page}{\insertnote}

\setvoiceovertts{azuretts.py}
\setvoiceover{en-GB:neural:Ryan}
\setvoiceoverext{.ogg}

\begin{document}

\begin{frame}[titleslide]
  \titlepage
\end{frame}

\AtBeginSection{
  \begin{frame}[sectionslide]{Overview}
    \tableofcontents[currentsection]
  \end{frame}
}
\AtBeginSubsection{
  \begin{frame}[subsectionslide]{Overview}
    \tableofcontents[currentsection,currentsubsection]
  \end{frame}
}

\section{Introduction}
\subsection{Slide making}

\begin{frame}
  {Good news}
\end{frame}
```

```

\voiceover {Slide making is fun}
You can keep on making your slides the way you are used to!
\begin{itemize}
\item all the nice \LaTeX{} stuff at your fingertips
\item no temptation to use too much unnecessary animation
\end{itemize}
\bigskip

Indeed, there are no tools that can typeset equations like the tools form the \TeX-ecosystem:
\begin{equation}
\eu^{-\ju\pi}+1=0
\end{equation}

\note[item]{Make joke about fingertips}
\note[item]{Don't speak about your personal temptations!}
\end{frame}

\begin{frame}[transition=concave,embed,autoslide=2000]
{Pr\^et-\`a-porter}
{A dummy slide}
\vfill
Showing off the 'concave' slide transition animation. Not recommended!
\vfill
\end{frame}

\begin{frame}[transition=convex]
{Très chique}
{A dummy slide}
\vfill
Showing off the 'convex' slide transition animation. Not recommended!
\vfill
\end{frame}

\subsection{Pimping your slides}

\begin{frame}
{And even more good news}
{\ldots almost seems to good to be true\ldots}
\small
However, now you can pimp your slides like never before. You can incorporate:
\begin{itemize}
\item videos and audio fragments
\item animated GIFs and LaTeX animations
\item iframe content
\end{itemize}
without being tied to Acrobat reader.
In addition, there are some extra features
\begin{itemize}
\item press '?' for keyboard help, amongst which you will find:
\item press 'm' to open the slide menu on the left
\item press 'o' to get an overview of the slides
\item press 's' to start a speaker view
\item press 'g' to go to a specific slide by typing its slide number
\end{itemize}
The pancake menu on the bottom left also opens the menu.
\end{frame}

\note{
Don't spend too much time on this slide
}

\audiostop{}

```

```

\begin{frame}[transition=zoom]
  {A dymmy slide}
  {number three}
  \vfill
  Showing off the 'zoom slide transition animation. Not recommended!
  \vfill
\end{frame}

\section{In detail}

\subsection{Candy for the eye}

\begin{frame}
  {Placing videos}
  {}
  \only<1>{On this first slide there is nothing to see. On the next animation frame, a video will appear.}
  \only<2>{Here it is!}
  \video<2>[above,draw,autoplay,height=0.7,aspectratio=16/9,
    background=yellow,fit=contain]
  (myvid) \at (0.5,0.1) {Media/beamer-reveal-testvideo.mp4}
  \tikz[remember picture,overlay] \node<2>[anchor=north,font=\tiny] at (myvid.south)
    {An example video (C) Walter Daems};
\end{frame}

\begin{frame}
  {Placing images (possibly animated)}
  {}
  \begin{columns}
    \column[T]{0.45\textwidth}
    Below you will find a png (for which you don't need reveal, BTW).
    \vspace*{1cm}

    Of course, you can exploit the transparency of the background layer in the PNG!
    \vspace*{2cm}

    And on the top right you will find a swinging pendulum (an animated GIF).
    \column[T]{0.45\textwidth}
    % this is a image whose box consumes area on your slide
    \image[width=0.33,aspectratio=1,fit=contain,draw] (myimage) {Media/beamer-reveal-AnimatedPendulum.gif}
  \end{columns}
  % this is an overlayed image (not consuming any space)
  \image[width=0.25,aspectratio=1,fit=contain] \at (0.1,0.6) {Media/beamer-reveal-WiresTp.png}
  \image[width=0.1,aspectratio=1,fit=contain] \at (0.9,0.1) {https://www.digmanwaves.net/About/Pictures/WalterDaems.}
\end{frame}

\begin{frame}
  {Placing iframe material (possibly animated)}
  {e.g. generated with asymptote}

  Click and drag on the iframe below. You can manipulate it! Use your mouse
  scroll-wheel to zoom in or out.
  \iframe[draw,anchor=north,height=0.6,aspectratio=16/9,embed,
    fit=cover] \at (0.5,0.7) {Media/beamer-reveal-PCB.html}
\end{frame}

\note{
  Don't mention the amount of hours that went into this iframe!
}

\subsection{Resonance for the ear}

```

```

\begin{frame}
  {Adding audio to your slides}
  {}
  Below, there is an audio block
  that automatically starts playing.\\
  \audio[autoplay,controls,width=0.1,aspectratio=16/9,
    background=blue,fit=cover] {Media/beamer-reveal-AudioSample.ogg}
\end{frame}

\subsection{Make (video) animations with LaTeX}

\newcommand\myanimation{%
  \begin{tikzpicture}[font=\footnotesize,transform shape,scale=0.75]
    \pgfmathsetmacro\angle{\progress*540}%
    \clip (-2,-5.25) rectangle (8,2);
    \node[below left,inner sep=1pt] at (0,0) {\tiny 0};
    \node[below left,inner sep=1pt] at (2.5,0) {\tiny 0};
    \node[above right,inner sep=1pt] at (0,-2) {\tiny 0};

    \begin{scope}[every node/.style={right}]
      \node[thick,draw,rectangle] at (2.5,-2)
        {\large  $x(t) = A \cdot e^{j\omega t}$ };
      \node at (3.5,-3)
        {\large  $e^{j\alpha} = \cos\alpha + j\sin\alpha$ };
      \node at (2.5,-4)
        {\large  $x(t) = \underbrace{A \cos \omega t}_{\text{\textcolor{orange}{real}}}
          + \underbrace{j A \sin \omega t}_{\text{\textcolor{olive}{imaginary}}}$ };
    \end{scope}
    \draw[->,thick] (3,-2.4) -- (3,-3.4);
    \draw[blue,thick] (0,0) circle (1);

    \draw[->] (-1.25,0) -- (1.25,0) node[below] {Re};
    \draw[->] (0,-1.25) -- (0,1.25) node[left] {Im};

    % circle
    \draw[olive,very thick] (0,0) -- (0,{sin(\angle)});
    \draw[orange,very thick] (0,0) -- ({cos(\angle)},0);
    \draw[blue,thick,->] (0,0) -- node[left,font=\tiny] {A} +(\angle:1);
    \draw[->] (0.4,0) arc (0:\angle:0.4);
    \node at (0.5*\angle:0.7) {\scriptsize  $\omega \tilde{t}$ };

    % right graph
    \draw[very thick,olive] ({2.5+\angle/180},0) -- +(0,{sin(\angle)});
    \draw[densely dotted] ({min(0,cos(\angle))},{sin(\angle)})
      -- ({2.5+\angle/180},{sin(\angle)});
    \draw[thick] ({2.5+\angle/180},0) +(0,1pt) -- +(0,-1pt) node[below] { $\tilde{t}$ };

    % bottom graph
    \draw[very thick,orange] (0,{ -2-\angle/180}) -- +({cos(\angle)},0);
    \draw[densely dotted] ({cos(\angle)},{max(0,sin(\angle))})
      -- ({cos(\angle)},{ -2-\angle/180});
    \draw[thick] (0,{ -2-\angle/180}) +(1pt,0) -- +(-1pt,0) node[left] { $\tilde{t}$ };

    % right graph
    \foreach \y/\l in {-1/-A,1/A} {
      \draw[gray,densely dotted] (2.5,\y) -- (6.25,\y);
      \draw (2.5,\y) +(1pt,0) -- +(-1pt,0) node[left] { $\l$ };
    }
    \draw[->] (2.0,0) -- (6.5,0) node[below] { $t$ };
    \draw[->] (2.5,-1.25) -- (2.5,1.25) node[left] { $\text{Im}(x(t))$ };
    \draw[olive,thick,domain=-0.25:3.5,samples=30,smooth] plot
      ({x+2.5},{sin(pi*x r)});
  \end{tikzpicture}
}

```

```

% bottom graph
\foreach \y/\l in {-1/-A,1/A} {
  \draw[gray,densely dotted] (\y,-2) -- (\y,-4.5);
  \draw (2.5,\y) +(1pt,0) -- +(-1pt,0) node[left] {$\l$};
}
\draw[>-] (-1.25,-2) -- (1.25,-2) node[above] {$\operatorname{Re}(x(t))$};
\draw[>-] (0,-1.5) -- (0,-5) node[left] {$t$};

\draw[orange,thick,domain=-0.25:2.6,samples=30,smooth] plot
  ({\cos(\pi*\x r)},{-2-\x});
\end{tikzpicture}%
}

\begin{frame}
  {Making animations with LaTeX (using TikZ as example)}
  {It is easier than ever before}
  \small The animation content is exported to a standalone LaTeX-document that creates a
  loop over it, for a \texttt{duration} seconds at
  \texttt{framerate} frames per second providing a \texttt{\textbackslash progress}
  variable that goes gradually from 0 to 1 in \texttt{duration} $\times$
  \texttt{framerate} frames. The beamer-reveal.pl script transforms it
  to mp4 maximally exploiting your multi-core CPU.

  \begin{center}
    \animation<1|handout:1-3>[framerate=25,duration=7.5,autoplay,
      progress={0.1|handout:1/0,2/0.2,3/0.5},expand once] {\myanimation}
  \end{center}
\end{frame}

\begin{frame}
  {Making animations with LaTeX (using TikZ as example)}
  {Another slide with just a 'still' - but with some music!}
  \small The animation content is exported to a standalone LaTeX-document that creates a
  loop over it, for a \texttt{duration} seconds at
  \texttt{framerate} frames per second providing a \texttt{\textbackslash progress}
  variable that goes gradually from 0 to 1 in \texttt{duration} $\times$
  \texttt{framerate} frames. The beamer-reveal.pl script transforms it
  to mp4 maximally exploiting your multi-core CPU.

  \begin{center}
    \still[progress=0.75,expand once] {\myanimation}
  \end{center}
  \audio[width=0.1,aspectratio=16/9,fit=cover,autoplay] \at (0,0)
    {Media/beamer-reveal-AudioSample.ogg}
\end{frame}

\end{document}
</example>

```

14 Implementation

14.1 The preamble of the package

```
1 <*reveal>
2 \ExplSyntaxOn
3 \tl_new:N \g_@@_fileversion_tl
4 \tl_set:Nn \g_@@_fileversion_tl {v1.10}
5 \tl_new:N \g_@@_filedate_tl
6 \tl_set:Nn \g_@@_filedate_tl {2026-04-08}
7 \RequirePackage{l3keys2e}
8 </reveal>
```

14.2 Error/warning messages

```
9 <*reveal>
10 \msg_new:nnn{ beamer-reveal } { inconsistent-dimensions } {
11   aspect-ratio-of-beamer~(#1)~and~reveal~(#2:#3)~are~not~consistent.\\
12   You~must~specify~consistent~values~for~width/height~and~aspectratio~
13   otherwise~your~reveal~items~(videos/images/animations)~will~not~appear~
14   on~the~right~locations~on~your~reveal~slidedeck.
15 }
16 \msg_new:nnn{ beamer-reveal } { missing-aspectratio } {
17   missing-aspect-ratio.\\
18   You~need~to~specify~at~least~an~aspect-ratio~for~a~beamer-reveal-item~
19   you~want~to~put~on~the~reveal~slide.
20 }
21 \msg_new:nnn{ beamer-reveal } { missing-width-or-height } {
22   missing-width~or~height.\\
23   You~need~to~specify~at~least~a~width~or~a~height~for~a~beamer-reveal-item~
24   you~want~to~put~on~the~reveal~slide.
25 }
26 \msg_new:nnn{ beamer-reveal } { overconstrained-box } {
27   overconstrained-box.\\
28   You~cannot~both~specify~the~width~and~the~height~of~a~beamer-reveal-item~
29   you~want~to~put~on~the~slide.
30   Specify~width~and~aspectratio~or~height~and~aspectratio.
31 }
32 \msg_new:nnn{ beamer-reveal } { dynamic-option-for-staticcontent } {
33   dynamic-option~given~(autoplay,~controls,~loop,~muted)~for~static~
34   content~(\image,~\iframe).\\
35   These~options~make~no~sense~for~the~\image~command.~Remove~them.
36 }
37 \msg_new:nnn{ beamer-reveal } { animation-option-for-nonanimation } {
38   duration~and~framerate~are~options~that~can~only~be~given~for~the~
39   \animation~command.\\
40   Remove~them~from~the~\video,~\audio,~\image~and~\iframe~commands.
41 }
42 \msg_new:nnn { beamerreveal / Syntax } { missing-coordinate }
43   { you~specified~'\c_backslash_str at'~but~gave~no~coordinate. }
44 \msg_new:nnn { beamerreveal / Syntax } { old-at-syntax }
45   { I~choked~on~'a';~note~that~the~syntax~has~changed:~replace~'at'~with~'\c_backslash_str at'. }
46 \msg_new:nnn { beamerreveal / Syntax } { missing-at }
47   { you~specified~a~coordinate~but~not~an~'\c_backslash_str at'~token;~did~you~forget~that? }
48 </reveal>
```

14.3 Some pdflatex backwards compatibility

```
49 <*reveal>
50 \ifundefined{pagewidth}{%
51   \let\pagewidth\pdfpagewidth
52   \let\pageheight\pdfpageheight
```

```
53 }{}
54 </reveal>
```

14.4 Package options

First some global variables to store the global width and height of the presentation, that can be specified as package options:

```
55 <*reveal>
56 \tl_new:N \g_@@_beameraspectratio_tl
57 \tl_set:Nn \g_@@_beameraspectratio_tl {43}
58 \fp_new:N \g_@@_canvaswidth_fp
59 \fp_set:Nn \g_@@_canvaswidth_fp { \dim_to_fp:n { \paperwidth} }
60 \fp_new:N \g_@@_canvasheight_fp
61 \fp_set:Nn \g_@@_canvasheight_fp { \dim_to_fp:n { \paperheight} }
62 \fp_new:N \g_@@_canvasaspectratio_fp
63 \fp_set:Nn \g_@@_canvasaspectratio_fp { \g_@@_canvaswidth_fp / \g_@@_canvasheight_fp }
64 \int_new:N \g_@@_canvaswidth_int
65 \int_set:Nn \g_@@_canvaswidth_int {0}
66 \int_new:N \g_@@_canvasheight_int
67 \int_set:Nn \g_@@_canvasheight_int {0}
68 \bool_new:N \g_@@_embed_bool
69 \bool_set:Nn \g_@@_embed_bool { \c_false_bool }
70 \bool_new:N \g_@@_loop_bool
71 \bool_set:Nn \g_@@_loop_bool { \c_false_bool }
72 \int_new:N \g_@@_autoslide_int
73 \int_set:Nn \g_@@_autoslide_int { -1 }
74 \keys_define:nn { beamerreveal } {
75   width      .int_set:N      = \g_@@_canvaswidth_int,
76   width      .value_required:n = true,
77   height     .int_set:N      = \g_@@_canvasheight_int,
78   height     .value_required:n = true,
79   embed      .bool_set:N     = \g_@@_embed_bool,
80   loop       .bool_set:N     = \g_@@_loop_bool,
81   autoslide  .int_set:N      = \g_@@_autoslide_int,
82   autoslide  .value_required:n = true,
83 }
84 \ProcessKeyOptions[beamerreveal]
85 \int_compare:nNnTF { \g_@@_canvaswidth_int } = {0}
86   {
87     \int_compare:nNnTF { \g_@@_canvasheight_int } = {0}
88       {
89         % we assume 4x3 on an HD screen
90         \int_set:Nn \g_@@_canvaswidth_int { 1920 }
91         \int_set:Nn \g_@@_canvasheight_int
92           { \fp_eval:n { round( \g_@@_canvaswidth_int / \g_@@_canvasaspectratio_fp ) } }
93       }
94       {
95         % we assume 4x3
96         \int_set:Nn \g_@@_canvaswidth_int
97           { \fp_eval:n { round( \g_@@_canvasheight_int * \g_@@_canvasaspectratio_fp ) } }
98       }
99   }
100   {
101     \int_compare:nNnTF { \g_@@_canvasheight_int } = {0}
102       {
103         % we assume 4x3 on an HD screen
104         \int_set:Nn \g_@@_canvasheight_int { \fp_eval:n
105           { round( \g_@@_canvaswidth_int / \g_@@_canvasaspectratio_fp ) } }
106       }
107       {
108         % both are set, we're good to go
```



```

109      \fp_new:N \l_@@_canvasaspectratioresidue_fp
110      \fp_set:Nn \l_@@_canvasaspectratioresidue_fp
111        { \fp_abs:n { \g_@@_canvaswidth_int / \g_@@_canvasheight_int - \g_@@_canvasaspectratio_fp } }
112      \fp_compare:nNnTF { \l_@@_canvasaspectratioresidue_fp } > {0.001}
113      {
114        \msg_warning:nneee { beamer-reveal } { inconsistent-dimensions }
115        { \tl_use:N \g_@@_beameraspectratio_tl }
116        { \int_use:N \g_@@_canvaswidth_int }
117        { \int_use:N \g_@@_canvasheight_int }
118      }{}
119    }
120  }
121 </reveal>

```

14.5 Extra options for the frame environment of BEAMER

```

122 <*reveal>
123 \bool_new:N \g_@@_titlepage_bool
124 \define@key{beamerframe}{titleslide}[true]{%
125   \ExplSyntaxOn
126   \bool_gset_true:N \g_@@_titlepage_bool
127   \ExplSyntaxOff
128 }
129 \bool_new:N \g_@@_sectionslide_bool
130 \define@key{beamerframe}{sectionslide}[true]{%
131   \ExplSyntaxOn
132   \bool_gset_true:N \g_@@_sectionslide_bool
133   \ExplSyntaxOff
134 }
135 \bool_new:N \g_@@_subsectionslide_bool
136 \define@key{beamerframe}{subsectionslide}[true]{%
137   \ExplSyntaxOn
138   \bool_gset_true:N \g_@@_subsectionslide_bool
139   \ExplSyntaxOff
140 }
141 \tl_new:N \g_@@_transition_tl
142 \define@key{beamerframe}{transition}[none]{%
143   \ExplSyntaxOn
144   \tl_gset:Nn \g_@@_transition_tl { #1 }
145   \ExplSyntaxOff
146 }
147 \define@key{beamerframe}{embed}[true]{%
148   \ExplSyntaxOn
149   \bool_gset_true:N \g_@@_embed_bool
150   \ExplSyntaxOff
151 }
152 \define@key{beamerframe}{autoslide}[0]{%
153   \ExplSyntaxOn
154   \int_set:Nn \g_@@_autoslide_int {#1}
155   \ExplSyntaxOff
156 }
157 </reveal>

```

14.6 File writing

File handles and auxiliary functions to write data to the .rvl file.

```

158 <*reveal>
159 \iow_new:N \g_@@_rvlfile
160 \iow_open:Nn \g_@@_rvlfile {\jobname.rvl}
161 </reveal>

```

`\writecomment_@@:nn`

```
162 <*reveal>
163 \cs_new:Npn \writecomment_@@:nn #1
164   { \iow_now:Ne \g_@@_rvlfile {\c_percent_str\c_percent_str\c_space_tl #1} }
165 </reveal>
```

`\writecontrol_@@:nn`

```
166 <*reveal>
167 \cs_new:Npn \writecontrol_@@:nn #1 #2
168   { \iow_now:Ne \g_@@_rvlfile {@ @#1:~#2} }
```

`\writeliteral_@@:nn`

```
169 \cs_new:Npn \writeliteral_@@:nn #1
170   { \iow_now:Nx \g_@@_rvlfile {#1} }
```

`\writeraw_@@:nn`

```
171 \cs_new:Npn \writeraw_@@:nn #1
172   { \iow_now:Nn \g_@@_rvlfile { #1 } }
```

`\writeraw_@@:nn`

```
173 \cs_generate_variant:Nn \iow_now:Nn {No }
174 \cs_new:Npn \writeraw_@@:nn #1
175   { \iow_now:No \g_@@_rvlfile { #1 } }
```

Now initialize our .rvl file.

```
176 \writecomment_@@:nn {Beamer-reveal-driver~file~
177   (\tl_use:N \g_@@_fileversion_tl - \tl_use:N \g_@@_filedate_tl)}
178 \writecontrol_@@:nn {Presentation} {}
179 \tl_new:N \l_@@_my_compiler_tl
180 \tl_set:Nn \l_@@_my_compiler_tl { unknown }
181 \sys_if_engine_pdftex:T { \tl_set:Nn \l_@@_my_compiler_tl { pdflatex } }
182 \sys_if_engine_xetex:T { \tl_set:Nn \l_@@_my_compiler_tl { xelatex } }
183 \sys_if_engine luatex:T { \tl_set:Nn \l_@@_my_compiler_tl { luatex } }
184 \AtBeginDocument{
185   \writeliteral_@@:nn {-parameters:
186     latexversion={\tl_use:N \g_@@_fileversion_tl},
187     latexdate={\tl_use:N \g_@@_filedate_tl},
188     sourcefilename={\currfilename},
189     title={\inserttitle},
190     author={\beamer@shortauthor},
191     compiler={\tl_use:N \l_@@_my_compiler_tl },
192     \bool_if:NT \g_@@_embed_bool {embed={}},
193     \bool_if:NTF \g_@@_loop_bool {loop={true}}{loop={false}},
194     \int_compare:nNnTF \g_@@_autoslide_int < {0} {}{autoslide={\int_use:N \g_@@_autoslide_int}},
195     canvaswidth={\int_use:N \g_@@_canvaswidth_int},
196     canvasheight={\int_use:N \g_@@_canvasheight_int}
197   }
198 }
199 </reveal>
```

14.7 Frame generation

```

200 <*reveal>
201 \AddToHook{ env / frame / begin} {
202   \bool_gset_false:N \g_@@_titlepage_bool
203   \bool_gset_false:N \g_@@_sectionslide_bool
204   \bool_gset_false:N \g_@@_subsectionslide_bool
205   \bool_gset_false:N \g_@@_embed_bool
206   \int_gset:Nn \g_@@_autoslide_int {-1}
207   \tl_gset:Nn \g_@@_transition_tl {none}
208 }
209 \bool_new:N \g_@@_animonslide_bool
210 \bool_new:N \g_@@_audioonslide_bool
211 \bool_new:N \g_@@_iframeonslide_bool
212 \AddToHook{ env / beamer@frameslide / before} {
213   \bool_gset_false:N \g_@@_animonslide_bool
214   \bool_gset_false:N \g_@@_audioonslide_bool
215   \bool_gset_false:N \g_@@_iframeonslide_bool
216   \writecontrol_@@:nn {BeamerFrame} {}
217 }
218 \AddToHook{ env / beamer@frameslide / end } {
219   \mode<presentation>{
220     \dim_zero_new:N \l_@@_markercount_dim
221     \bool_if:NTF \g_@@_animonslide_bool {
222       \dim_sub:Nn \l_@@_markercount_dim { 1ex }
223       \tikz[overlay,remember-picture] \node[anchor=south-east,shape=
224         circle,draw,gray,minimum-size=1mm,inner-sep=0pt,outer-sep=0pt,font=\TINY]
225       at ($(current~page.south-east)+(\dim_use:N \l_@@_markercount_dim,+1ex)$) {A};
226     }{}
227     \bool_if:NTF \g_@@_iframeonslide_bool {
228       \dim_sub:Nn \l_@@_markercount_dim { 1ex }
229       \tikz[overlay,remember-picture] \node[anchor=south-east,shape=
230       circle,draw,gray,minimum-size=1mm,inner-sep=0pt,outer-sep=0pt,font=\TINY]
231       at ($(current~page.south-east)+(\dim_use:N \l_@@_markercount_dim,+1ex)$) {I};
232     }{}
233     \bool_if:NTF \g_@@_audioonslide_bool {
234       \dim_sub:Nn \l_@@_markercount_dim { 1ex }
235       \tikz[overlay,remember-picture] \node[anchor=south-east,shape=
236       circle,draw,gray,minimum-size=1mm,inner-sep=0pt,outer-sep=0pt,font=\TINY]
237       at ($(current~page.south-east)+(\dim_use:N \l_@@_markercount_dim,+1ex)$) {S};
238     }{}
239   }
240 }
241 \AddToHook{ env / beamer@frameslide / after} {
242   \writeliteral_@@:n {-parameters:
243     rawpage={\insertpagenumber},
244     truepage={\insertframenummer},
245     overlay={\insertoverlaynumber},
246     transition={\tl_use:N \g_@@_transition_tl},
247     \bool_if:NT \g_@@_embed_bool {embed={},},
248     \int_compare:nNnTF \g_@@_autoslide_int < {0} {} {autoslide={\int_use:N \g_@@_autoslide_int},},
249     \bool_if:NT \g_@@_embed_bool {embed={},},
250     \bool_if:NTF \g_@@_sectionslide_bool
251     {
252       title={\secname},toc={section}
253     }
254     {
255       \bool_if:NTF \g_@@_subsectionslide_bool
256       {
257         title={\subsecname},toc={subsection}
258       }
259       {

```

```

260         \bool_if:NTF \g_@@_titlepage_bool
261         {
262             title={\beamer@shorttitle},toc={titlepage}
263         }
264         {
265             \sys_if_engine_pdftex:TF % avoid pdflatex screwing up old accents
266             {title={\unexpanded\expandafter{\beamer@shortframetitle}}}
267             {title={\beamer@shortframetitle}}
268         }
269     }
270 }
271 }
272 }
273 \AtBeginDocument {
274     \AddToHook { cmd / note / before } {
275         \writeliteral_@@:n
276         {-hasnote:true}
277     }
278 }
279 </reveal>

```

14.8 Common keys for the macros

```

280 <*reveal>
281 \fp_new:N \l_@@_mediawidth_fp
282 \fp_new:N \l_@@_mediaheight_fp
283 \fp_new:N \l_@@_mediaframerate_fp
284 \fp_new:N \l_@@_mediaduration_fp
285 \fp_new:N \l_@@_mediapdfprogress_fp
286 \tl_new:N \l_@@_mediafit_tl
287 \tl_new:N \l_@@_mediabackground_tl
288 \tl_new:N \l_@@_mediahandoutprogress_tl
289 \tl_new:N \l_@@_mediaprogress_tl
290 \fp_new:N \l_@@_xposdelta_fp
291 \fp_new:N \l_@@_yposdelta_fp
292 \bool_new:N \l_@@_mediaembed_bool
293 \bool_new:N \l_@@_mediaautoplay_bool
294 \bool_new:N \l_@@_medialoop_bool
295 \bool_new:N \l_@@_mediaboxdraw_bool
296 \bool_new:N \l_@@_mediamuted_bool
297 \bool_new:N \l_@@_mediacontrols_bool
298 \tl_new:N \l_@@_mediaanchor_tl
299 \bool_new:N \l_@@_mediaexpandonce_bool
300
301 \msg_new:nnn { beamerreveal / Syntax } { unknown-key }
302 { Unknown~key~'#1'~for~media~(video,~animated,~...)~command. }
303 \msg_new:nnn { beamerreveal / Syntax } { illegal-keys-video }
304 { Illegal~key(s)~'#1'~for~a~\video. }
305 \msg_new:nnn { beamerreveal / Syntax } { illegal-keys-image }
306 { Illegal~key(s)~'#1'~for~an~\image. }
307 \msg_new:nnn { beamerreveal / Syntax } { illegal-keys-iframe }
308 { Illegal~key(s)~'#1'~for~an~\iframe. }
309 \msg_new:nnn { beamerreveal / Syntax } { illegal-keys-animation }
310 { Illegal~key(s)~'#1'~for~an~\animation. }
311 \msg_new:nnn { beamerreveal / Syntax } { key-syntax-changed }
312 { The~syntax~of~the~key~of~'#1'~changed;~change~this~into~'#2'. }
313 \keys_define:nn { beamerreveal / media } {
314     width .fp_set:N          = \l_@@_mediawidth_fp,
315     width .value_required:n  = true,
316     width .initial:n         = 0,
317     width .groups:n          = { size },
318     height .fp_set:N         = \l_@@_mediaheight_fp,

```

```

319 height .value_required:n      = true,
320 height .initial:n             = 0,
321 height .groups:n              = { size },
322 aspectratio .fp_set:N         = \l_@@_mediaaspectratio_fp,
323 aspectratio .value_required:n = true,
324 aspectratio .groups:n         = { size },
325 fit .tl_set:N                 = \l_@@_mediafit_tl,
326 fit .value_required:n         = true,
327 fit .initial:n                = fill,
328 fit .groups:n                 = { fit },
329 background .tl_set:N          = \l_@@_mediabackground_tl,
330 background .value_required:n  = true,
331 background .initial:n         = white,
332 background .groups:n          = { draw },
333 draw .bool_set:N              = \l_@@_mediaboxdraw_bool,
334 draw .initial:n               = false,
335 draw .groups:n                = { draw },
336 embed .bool_set:N             = \l_@@_mediaembed_bool,
337 embed .initial:n              = false,
338 autoplay .bool_set:N          = \l_@@_mediaautoplay_bool,
339 autoplay .initial:n           = false,
340 autoplay .groups:n             = { dynamic },
341 loop .bool_set:N              = \l_@@_medialoop_bool,
342 loop .initial:n               = false,
343 loop .groups:n                = { dynamic },
344 controls .bool_set:N          = \l_@@_mediacontrols_bool,
345 controls .initial:n           = false,
346 controls .groups:n            = { dynamic },
347 muted .bool_set:N             = \l_@@_mediamuted_bool,
348 muted .initial:n              = false,
349 muted .groups:n               = { sound },
350 duration .fp_set:N            = \l_@@_mediaduration_fp,
351 duration .initial:n           = 0,
352 duration .value_required:n    = true,
353 duration .groups:n            = { animation },
354 pdfprogress .code:n           = { \msg_fatal:nnnn { beamerreveal / Syntax } { key-syntax-changed }
355   { pdfprogress=#1 } { progress=#1 } },
356 pdfprogress .groups:n         = { animation },
357 pdf-progress .code:n          = { \msg_fatal:nnnn { beamerreveal / Syntax } { key-syntax-changed }
358   { pdf-progress=#1 } { progress=#1 } },
359 pdf-progress .groups:n        = { animation },
360 handout-progress .code:n      = { \msg_fatal:nnnn { beamerreveal / Syntax } { key-syntax-changed }
361   { handout-progress=#1 } { progress={0.1|handout:1/0.2,2/0.5,3/0.75} } },
362 handout-progress .groups:n    = { animation },
363 progress .tl_set:N            = \l_@@_mediaprogress_tl,
364 progress .value_required:n    = true,
365 progress .groups:n            = { animation },
366 framerate .fp_set:N          = \l_@@_mediaframerate_fp,
367 framerate .initial:n         = 0,
368 framerate .value_required:n   = true,
369 framerate .groups:n           = { animation },
370 expand-once .bool_set:N       = \l_@@_mediaexpandonce_bool,
371 expand-once .initial:n        = false,
372 expand-once .groups:n         = { animation },
373 anchor .choice:,
374 anchor / center .code:n      = { \fp_set:Nn \l_@@_xposdelta_fp { -0.5 }
375   \fp_set:Nn \l_@@_yposdelta_fp { -0.5 }
376   \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
377 anchor / west .code:n        = { \fp_set:Nn \l_@@_xposdelta_fp { 0 }
378   \fp_set:Nn \l_@@_yposdelta_fp { -0.5 }
379   \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
380 anchor / north-west .code:n  = { \fp_set:Nn \l_@@_xposdelta_fp { 0 }

```

```

381             \fp_set:Nn \l_@@_yposdelta_fp { 0 }
382             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
383 anchor / north .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { -0.5 }
384             \fp_set:Nn \l_@@_yposdelta_fp { 0 }
385             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
386 anchor / north~east .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { -1 }
387             \fp_set:Nn \l_@@_yposdelta_fp { 0 }
388             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
389 anchor / east .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { -1 }
390             \fp_set:Nn \l_@@_yposdelta_fp { -0.5 }
391             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
392 anchor / south~east .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { -1 }
393             \fp_set:Nn \l_@@_yposdelta_fp { -1 }
394             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
395 anchor / south .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { -0.5 }
396             \fp_set:Nn \l_@@_yposdelta_fp { -1 }
397             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
398 anchor / south~west .code:n = { \fp_set:Nn \l_@@_xposdelta_fp { 0 }
399             \fp_set:Nn \l_@@_yposdelta_fp { -1 }
400             \tl_set:Nn \l_@@_mediaanchor_tl {#1} },
401 anchor .value_required:n = true,
402 anchor .initial:n = center,
403 anchor .groups:n = { position },
404 above .meta:n = { anchor = south },
405 above .groups:n = { position },
406 below .code:n = { anchor = north },
407 below .groups:n = { position },
408 left .code:n = { anchor = east },
409 left .groups:n = { position },
410 right .code:n = { anchor = west },
411 right .groups:n = { position },
412 above~left .code:n = { anchor = south east },
413 above~left .groups:n = { position },
414 above~right .code:n = { anchor = south west },
415 above~right .groups:n = { position },
416 below~left .code:n = { anchor = north east },
417 below~left .groups:n = { position },
418 below~right .code:n = { anchor = north west },
419 below~right .groups:n = { position },
420 unknown .code:n =
421 {
422   \msg_fatal:nne { beamerreveal / Syntax } { unknown-key } {\l_keys_key_str}
423 },
424 }
425 </reveal>

```

14.9 Fiddling with relative widths/heights

As mentioned in the section ‘quirks’ the package only allows specifying box dimensions as height and an aspect ratio, or width and an aspect ratio. BEAMER-REVEAL will recalculate these into fractional values relative to the slide width and height.

In order not to get lost in the calculations, let’s find the equations we need to implement. Let’s call the absolute width and height W and L and the relative width and height w and h . Let W_P and H_P be the absolute page width and height. Obviously:

$$w = \frac{W}{W_P} \qquad h = \frac{H}{H_P} \qquad (2)$$

Now let’s denote the aspect ratio of an object as A and the aspect ratio of the page as A_P .

$$A = \frac{W}{H} \qquad A_P = \frac{W_P}{H_P} \qquad (3)$$

This allows to calculate w from h and vice versa:

$$w = \frac{W}{W_p} = \frac{AH}{A_p H_p} = \frac{A}{A_p} h \qquad h = \frac{H}{H_p} = \frac{W/A}{W_p/A_p} = \frac{A_p}{A} w \quad (4)$$

The following macro calculates verifies if at least the aspectratio A has been given otherwise a fatal error is generated. It then calculates w from h or h from w unless both w and h are missing, or unless both have been specified.

`\process_a_w_h_@@:NNN`

```

426 <*reveal>
427 \cs_new:Npn \process_a_w_h_@@:NNN #1#2#3 {
428   \fp_compare:nNnTF { #1 } = {0} {
429     \msg_fatal:nn { beamer-reveal } { missing-aspectratio }
430   } {
431     \fp_compare:nNnTF { #2 } = {0} {
432       \fp_compare:nNnTF { #3 } = {0} {
433         \msg_fatal:nn { beamer-reveal } { missing-width-or-height }
434       } {
435         % calculating w
436         \fp_gset:Nn \l_tmpa_fp { round( #1 / \g_@@_canvasaspectratio_fp * #3, 6 ) }
437         \fp_gset:Nn \l_tmpb_fp { #3 }
438       }
439     } {
440       \fp_compare:nNnTF { #3 } = {0} {
441         % calculating h
442         \fp_gset:Nn \l_tmpa_fp { #2 }
443         \fp_gset:Nn \l_tmpb_fp { round( \g_@@_canvasaspectratio_fp / #1 * #2, 6 ) }
444       } {
445         \msg_fatal:nn { beamer-reveal } { overconstrained-box }
446       }
447     }
448   }
449   \fp_set_eq:NN #2 \l_tmpa_fp
450   \fp_set_eq:NN #3 \l_tmpb_fp
451 }
452 </reveal>

```

14.10 Auxiliary functions

`\@@extractlleft`

```

453 <*reveal>
454 \newdimen\xlleft
455 \newdimen\ylleft
456 \newcommand*\@@extractlleft[1]{\path (#1);\pgfgetlastxy{\xlleft}{\ylleft};}
457 </reveal>

```

`\@@extractupright`

```

458 <*reveal>
459 \newdimen\xupright
460 \newdimen\yupright
461 \newcommand*\@@extractupright[1]{\path (#1);\pgfgetlastxy{\xupright}{\yupright};}
462 </reveal>

```

Finally, some native l3exp fp variables to hold x and y position.

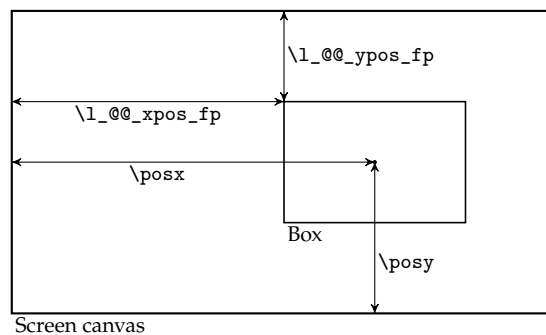
```
463 <*reveal>
464 \fp_new:N \l_@@_xpos_fp
465 \fp_new:N \l_@@_ypos_fp
466 </reveal>
```

14.11 Macros

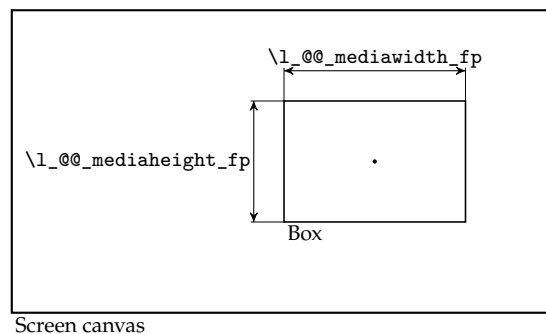
14.11.1 Main macros

The macro's fiddle with positions and sizes of boxes. Therefore it helps to have the following pictures in mind when reading the code.

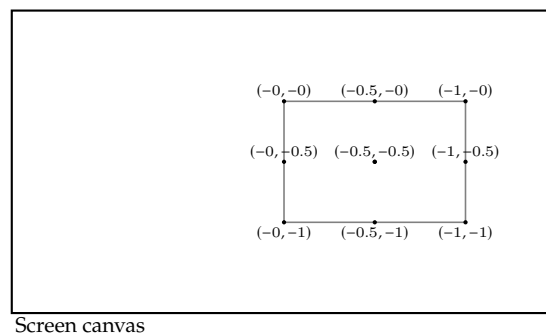
The parameters `\posx` and `\posy` are the relative BEAMER coordinates of the box. The parameters `\l_@@_xpos_fp` and `\l_@@_ypos_fp` are the HTML top and left distances of the top left corner of the box. All parameters are relative to the screen canvas dimensions (and therefore have values in between 0 and 1).



The width and height of the box have dedicated parameters.



The anchor of a box can be any of the main 8 wind directions (north, east, south, west, and the ones in between those). This anchor location in combination with the width and height of the box, allow for a correction on the HTML coordinates. The correction coefficients (relative to width and height) are:



The flow of the macros is as follows. We use the `\video` macro as example:

- `|\video|`:
- knows it is a `|Dynamic|` element (= XXX below)
- parses the appropriate keys from the option parameter list
- raises a fatal error if unknown keys have been detected
- hands over to the `|\contentDispatcher|`:
- checks if this is overlay content (`|\at (x,y)|`); hands over to the `|\overlayXXXContent|`: generates overlay content
- checks if this is insert content (no `|\at (x,y)|`); hands over to the `|\insertXXXContent|`: generates insert content
- otherwise: raises correct fatal syntax errors

`\video`

```

467 <*reveal>
468 \tl_new:N \l_@@_keyoverflow_tl
469 \NewDocumentCommand\video{D<>{1-} 0{} d() t\at d() m }{
470   \only<#1> {
471     % begin group to keep the key setting local
472     \group_begin:
473     \keys_set_exclude_groups:nnnN { beamerreveal / media } { animation } { #2 } \l_@@_keyoverflow_tl
474     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
475       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-video } { \l_@@_keyoverflow_tl }
476     }
477     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {video}
478                                   \overlayDynamicContent_@@:nnnn
479                                   \insertDynamicContent_@@:nnn
480   \group_end:
481 }
482 }
483 </reveal>

```

`\audio`

```

484 <*reveal>
485 \NewDocumentCommand\audio{ D<>{1-} 0{} d() t\at d() m }{
486   \only<#1> {
487     \bool_gset_true:N \g_@@_audioonslide_bool
488     % begin group to keep the key setting local
489     \group_begin:
490     \keys_set_exclude_groups:nnnN { beamerreveal / media } { animation } { #2 } \l_@@_keyoverflow_tl
491     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
492       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-audio } { \l_@@_keyoverflow_tl }
493     }
494     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {audio}
495                                   \overlayDynamicContent_@@:nnnn
496                                   \insertDynamicContent_@@:nnn
497   \group_end:
498 }
499 }
500 </reveal>

```

`\iframe`

```

501 <*reveal>

```

```

502 \NewDocumentCommand\iframe{D<>{1-} 0{} d() t\at d() m }{
503   \only<#1> {
504     \bool_gset_true:N \g_@@_iframeonslide_bool
505     % begin group to keep the key setting local
506     \group_begin:
507     \keys_set_exclude_groups:nnnN
508       { beamerreveal / media } { dynamic, sound, animation } { #2 } \l_@@_keyoverflow_tl
509     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
510       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-iframe } { \l_@@_keyoverflow_tl }
511     }
512     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {iframe}
513                                   \overlayDynamicContent_@@:nnnn
514                                   \insertDynamicContent_@@:nnn
515   \group_end:
516 }
517 }
518 </reveal>

```

\image

```

519 <*reveal>
520 \NewDocumentCommand\image{D<>{1-} 0{} d() t\at d() m}{
521   \only<#1> {
522     % begin group to keep the key setting local
523     \group_begin:
524     \keys_set_exclude_groups:nnnN
525       { beamerreveal / media } { dynamic, sound, animation } { #2 } \l_@@_keyoverflow_tl
526     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
527       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-image } { \l_@@_keyoverflow_tl }
528     }
529     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {image}
530                                   \overlayStaticContent_@@:nnnn
531                                   \insertStaticContent_@@:nnn
532   \group_end:
533 }
534 }
535 </reveal>

```

\animation

```

536 <*reveal>
537 \NewDocumentCommand\animation{D<>{1-} 0{} d() t\at d() +m}{
538   \only<#1> {
539     \bool_gset_true:N \g_@@_animonslide_bool
540     % begin group to keep the key setting local
541     \group_begin:
542     \keys_set_exclude_groups:nnnN
543       { beamerreveal / media } { size, sound, fit } { #2 } \l_@@_keyoverflow_tl
544     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
545       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-animation } { \l_@@_keyoverflow_tl }
546     }
547     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {animation}
548                                   \overlayFixedContent_@@:nnnn
549                                   \insertFixedContent_@@:nnn
550   \group_end:
551 }
552 }
553 </reveal>

```

`\still`

```
554 <*reveal>
555 \NewDocumentCommand\still{D<>{1-} 0{} d() t\at d() +m}{
556   \only<#1> {
557     % begin group to keep the key setting local
558     \group_begin:
559     \keys_set_exclude_groups:nnnN
560       { beamerreveal / media } { size, sound, fit, dynamic } { #2 } \l_@@_keyoverflow_tl
561     \tl_if_empty:NTF \l_@@_keyoverflow_tl {} {
562       \msg_fatal:nne { beamerreveal / Syntax } { illegal-keys-animation } { \l_@@_keyoverflow_tl }
563     }
564     \contentDispatcher_@@:nnnnnNN {#3} {#4} {#5} {#6} {still}
565                                   \overlayFixedContent_@@:nnnn
566                                   \insertFixedContent_@@:nnn
567   \group_end:
568 }
569 }
570 </reveal>
```

`\voiceover`

```
571 <*reveal>
572 \int_zero_new:N \g_@@_voiceovercounter_int
573 \str_new:N \g_@@_voiceoverext_str
574 \str_gset:Nn \g_@@_voiceoverext_str { ERROR }
575 \str_new:N \g_@@_voiceover_str
576 \str_gset:Nn \g_@@_voiceover_str { ERROR }
577 \str_new:N \g_@@_voiceovertts_str
578 \str_gset:Nn \g_@@_voiceovertts_str { ERROR }
579 \NewDocumentCommand\voiceover{ D<>{1-} 0{} +m }{
580   \only<#1> {
581     \int_gincr:N \g_@@_voiceovercounter_int
582     \writeliteral_@@:n {
583       -voiceover:
584       file={voiceover-\int_use:N \g_@@_voiceovercounter_int \str_use:N \g_@@_voiceoverext_str},
585       ext={\str_use:N \g_@@_voiceoverext_str},
586       voice={\str_use:N \g_@@_voiceover_str},
587       tool={\str_use:N \g_@@_voiceovertts_str},
588       text={#3}
589     }
590     \audio<#1> [autoplay,aspectratio=16/9,width=1cm,#2] \at (0,0)
591       {voiceover-\int_use:N \g_@@_voiceovercounter_int \str_use:N \g_@@_voiceoverext_str}
592   }
593 }
594 </reveal>
```

`\setvoiceovertts`

```
595 <*reveal>
596 \NewDocumentCommand\setvoiceovertts{ m }{
597   \str_gset:Nn \g_@@_voiceovertts_str {#1}
598 }
599 </reveal>
```

`\setvoiceoverext`

```
600 <*reveal>
601 \NewDocumentCommand\setvoiceoverext{ m }{
602   \str_gset:Nn \g_@@_voiceoverext_str {#1}
603 }
604 </reveal>
```

`\setvoiceover`

```
605 <*reveal>
606 \NewDocumentCommand\setvoiceover{ m }{
607   \str_gset:Nn \g_@@_voiceover_str {#1}
608 }
609 </reveal>
```

14.11.2 Auxiliary macros

`\at`

This is actually no macro, it even won't clash with a macro called `\at` that the user may declare. It is just a parsing token as documented in `usrguide.pdf` under 'control sequence tokens'.

`\contentDispatcher_@@:nnnnn`

```
610 <*reveal>
611 \cs_new:Npn \contentDispatcher_@@:nnnnnNN #1 #2 #3 #4 #5 #6 #7 {
612   \IfBooleanTF{#2}
613     {% \at
614       \IfNoValueTF{#3}
615         { \msg_fatal:nn { beamerreveal / Syntax } { missing-coordinate } }
616         { #6 {#1} {#3} {#4} {#5} }
617       }
618     {
619       %no \at
620       \IfNoValueTF{#3}
621         {
622           \tl_if_eq:nnTF {#4} a
623             { \msg_fatal:nn { beamerreveal / Syntax } { old-at-syntax } }
624             { #7 {#1} {#4} {#5} }
625           }
626         { \msg_fatal:nn { beamerreveal / Syntax } { missing-at } }
627       }
628 }
629 </reveal>
```

`\overlayDynamicContent_@@:nnnn`

```
630 <*reveal>
631 \cs_new:Npn \overlayDynamicContent_@@:nnnn #1#2#3#4 {
632   % convert combination width/aspectratio to height, or
633   % height/aspectratio to width
634   \process_a_w_h_@@:NNN \l_@@_mediaaspectratio_fp \l_@@_mediawidth_fp \l_@@_mediaheight_fp
635 }
```

```

636 % extract relative position from bottom left of page (0,0) to top right (1,1)
637 \seq_set_split:Nnn \l_tmpa_seq { , } { #2 }
638 \pgfmathsetmacro{\posx}{\seq_item:Nn \l_tmpa_seq {1}}
639 \pgfmathsetmacro{\posy}{\seq_item:Nn \l_tmpa_seq {2}}
640 % convert to html coordinates from top left corner, and correct for anchor location
641 \fp_set:Nn \l_@@_xpos_fp { \posx + \l_@@_xposdelta_fp * \l_@@_mediawidth_fp }
642 \fp_set:Nn \l_@@_ypos_fp { (1-\posy) + \l_@@_yposdelta_fp * \l_@@_mediaheight_fp }
643
644 % write info to .rvl file
645 \writeliteral_@@:n {
646   -#4:
647   width={\fp_use:N \l_@@_mediawidth_fp},
648   height={\fp_use:N \l_@@_mediaheight_fp},
649   fit={\tl_use:N \l_@@_mediafit_tl},
650   background={\tl_use:N \l_@@_mediabackground_tl},
651   \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={},} {}
652   \bool_if:NTF \l_@@_mediacontrols_bool {controls={},} {}
653   \bool_if:NTF \l_@@_medialoop_bool {loop={},} {}
654   \bool_if:NTF \l_@@_mediamuted_bool {muted={},} {}
655   \bool_if:NTF \l_@@_mediaembed_bool {embed={},} {}
656   x={\fp_use:N \l_@@_xpos_fp},
657   y={\fp_use:N \l_@@_ypos_fp},
658   file={#3}
659 }
660
661 % write node to PDF file
662 \bool_if:NTF \l_@@_mediaboxdraw_bool {
663   \begin{tikzpicture}[overlay,remember~picture,font=\tiny]
664     \@@extractlloleft{$(current~page.south~west)$}
665     \@@extractupright{$(current~page.north~east)$}
666     \node[
667       anchor = \tl_use:N \l_@@_mediaanchor_tl,
668       minimum~width={\fp_use:N \l_@@_mediawidth_fp * (\xupright - \xlloleft)},
669       minimum~height={\fp_use:N \l_@@_mediaheight_fp * (\yupright - \ylloleft)},
670       draw,
671       fill=\tl_use:N \l_@@_mediabackground_tl,
672     ] (#1) at ({\xlloleft*(1-\posx)+\xupright*\posx},{\ylloleft*(1-\posy)+\yupright*\posy})
673       {\textcolor{gray}{#3}};
674   \end{tikzpicture}
675 }{}
676 }
677 </reveal>

```

`\insertDynamicContent_@@:nnn`

```

678 <*reveal>
679 \cs_new:Npn \insertDynamicContent_@@:nnn #1#2#3 {
680   % parameters: nodename filename content-type
681   % convert combination width/aspectratio to height, or
682   % height/aspectratio to width
683   \process_a_w_h_@@:NNN \l_@@_mediaaspectratio_fp \l_@@_mediawidth_fp \l_@@_mediaheight_fp
684
685   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \pagewidth } }
686   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pageheight } }
687
688   % write node to PDF file
689   \begin{tikzpicture}[remember~picture,font=\tiny]
690     \node[
691       outer~sep=0pt,inner~sep=0pt,rectangle,
692       anchor = \tl_use:N \l_@@_mediaanchor_tl,

```

```

693     minimum-width={\fp_use:N \l_@@_mediawidth_fp * \fp_use:N \l_tmpa_fp },
694     minimum-height={\fp_use:N \l_@@_mediaheight_fp * \fp_use:N \l_tmpb_fp },
695     \bool_if:NTF \l_@@_mediaboxdraw_bool {draw} {},
696     fill=\tl_use:N \l_@@_mediabackground_tl,
697 ] (#1) {\bool_if:NTF \l_@@_mediaboxdraw_bool {\textcolor{gray}{#2}}{}};
698 \path let \p1=($(#1.north-west) - (current-page.north-west))$)
699 in \pgfextra{
700     \fp_set:Nn \l_@@_xpos_fp { \x1 / \fp_use:N \l_tmpa_fp }
701     \fp_set:Nn \l_@@_ypos_fp { - \y1 / \fp_use:N \l_tmpb_fp }
702     % write info to .rvl file
703     \writeliteral_@@:n {
704         -#3:
705         width={\fp_use:N \l_@@_mediawidth_fp},
706         height={\fp_use:N \l_@@_mediaheight_fp},
707         fit={\tl_use:N \l_@@_mediafit_tl},
708         background={\tl_use:N \l_@@_mediabackground_tl},
709         \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={},} {}
710         \bool_if:NTF \l_@@_mediacontrols_bool {controls={},} {}
711         \bool_if:NTF \l_@@_medialoop_bool {loop={},} {}
712         \bool_if:NTF \l_@@_mediamuted_bool {muted={},} {}
713         x={\fp_use:N \l_@@_xpos_fp},
714         y={\fp_use:N \l_@@_ypos_fp},
715         file={#2}
716     }
717 };
718 \end{tikzpicture}
719 }
720 </reveal>

```

\overlayStaticContent_@@:nnnn

```

721 <*reveal>
722 \cs_new:Npn \overlayStaticContent_@@:nnnn #1#2#3#4 {
723     % convert combination width/aspectratio to height, or
724     %             height/aspectratio to width
725     \process_a_w_h_@@:NNN \l_@@_mediaaspectratio_fp \l_@@_mediawidth_fp \l_@@_mediaheight_fp
726
727     % extract relative position from bottom left of page (0,0) to top right (1,1)
728     \seq_set_split:Nnn \l_tmpa_seq { , } { #2 }
729     \pgfmathsetmacro{\posx}{\seq_item:Nn \l_tmpa_seq {1}}
730     \pgfmathsetmacro{\posy}{\seq_item:Nn \l_tmpa_seq {2}}
731     % convert to html coordinates from top left corner, and correct for anchor location
732     \fp_set:Nn \l_@@_xpos_fp { \posx + \l_@@_xposdelta_fp * \l_@@_mediawidth_fp }
733     \fp_set:Nn \l_@@_ypos_fp { (1-\posy) + \l_@@_yposdelta_fp * \l_@@_mediaheight_fp }
734
735     % write info to .rvl file
736     \writeliteral_@@:n {
737         -#4:
738         width={\fp_use:N \l_@@_mediawidth_fp},
739         height={\fp_use:N \l_@@_mediaheight_fp},
740         fit={\tl_use:N \l_@@_mediafit_tl},
741         background={\tl_use:N \l_@@_mediabackground_tl},
742         \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={true},} {}
743         \bool_if:NTF \l_@@_mediaembed_bool {embed={},} {}
744         x={\fp_use:N \l_@@_xpos_fp},
745         y={\fp_use:N \l_@@_ypos_fp},
746         file={#3}
747     }
748
749     % write node to PDF file

```

```

750 \bool_if:NTF \l_@@_mediaboxdraw_bool {
751   \begin{tikzpicture}[overlay,remember~picture,font=\tiny]
752     \@@extractloleft{$(current~page.south~west)$}
753     \@@extractupright{$(current~page.north~east)$}
754     \node[
755       anchor = \tl_use:N \l_@@_mediaanchor_tl,
756       minimum~width={\fp_use:N \l_@@_mediawidth_fp * (\xupright - \xlleft)},
757       minimum~height={\fp_use:N \l_@@_mediaheight_fp * (\yupright - \ylleft)},
758       draw,
759       fill=\tl_use:N \l_@@_mediabackground_tl,
760     ] (#1) at ({\xlleft*(1-\posx)+\xupright*\posx},{\ylleft*(1-\posy)+\yupright*\posy})
761       {\textcolor{gray}{#3}};
762   \end{tikzpicture}
763 }{}
764 }
765 \</reveal>

```

\insertStaticContent_@@:nnn

```

766 \<*reveal>
767 \cs_new:Npn \insertStaticContent_@@:nnn #1#2#3 {
768   % convert combination width/aspectratio to height, or
769   % height/aspectratio to width
770   \process_a_w_h_@@:NNN \l_@@_mediaaspectratio_fp \l_@@_mediawidth_fp \l_@@_mediaheight_fp
771
772   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \pagewidth } }
773   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pageheight } }
774
775   % write node to PDF file
776   \begin{tikzpicture}[remember~picture,font=\tiny]%
777     \node[
778       outer~sep=0pt,inner~sep=0pt,rectangle,
779       anchor = \tl_use:N \l_@@_mediaanchor_tl,
780       minimum~width={\fp_use:N \l_@@_mediawidth_fp * \fp_use:N \l_tmpa_fp },
781       minimum~height={\fp_use:N \l_@@_mediaheight_fp * \fp_use:N \l_tmpb_fp },
782       \bool_if:NTF \l_@@_mediaboxdraw_bool {draw}{},
783       \bool_if:NTF \l_@@_mediaembed_bool {embed={},}{},
784       fill=\tl_use:N \l_@@_mediabackground_tl,
785     ] (#1) {\bool_if:NTF \l_@@_mediaboxdraw_bool {\textcolor{gray}{#2}}{}};
786     \path let \p1=($(#1.north~west) - (current~page.north~west)$)
787     in \pgfextra{
788       \fp_set:Nn \l_@@_xpos_fp { \x1 / \fp_use:N \l_tmpa_fp }
789       \fp_set:Nn \l_@@_ypos_fp { - \y1 / \fp_use:N \l_tmpb_fp }
790       % write info to .rvl file
791       \writeliteral_@@:n {
792         -#3:
793         width={\fp_use:N \l_@@_mediawidth_fp},
794         height={\fp_use:N \l_@@_mediaheight_fp},
795         fit={\tl_use:N \l_@@_mediafit_tl},
796         background={\tl_use:N \l_@@_mediabackground_tl},
797         \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={true},}{},
798         x={\fp_use:N \l_@@_xpos_fp},
799         y={\fp_use:N \l_@@_ypos_fp},
800         file={#2}
801       }
802     };
803   \end{tikzpicture}
804 }
805 \</reveal>

```

```
806 <*reveal>
807 \cs_new:Npn \overlayFixedContent_@@:nnnn #1#2#3#4 {
808   % set up the progress macro
809   \regex_split:nVN {\handout:} { \l_@@_mediaprogress_tl } \l_tmpa_seq
810   \fp_set:Nn \l_@@_mediapdfprogress_fp { \seq_item:Nn \l_tmpa_seq {1} }
811   \tl_set:Nx \l_@@_mediahandoutprogress_tl { \seq_item:Nn \l_tmpa_seq {2} }
812   \str_if_eq:eeTF { \use:c { beamer@currentmode } } { handout }
813   {
814     \tl_if_empty:NTF \l_@@_mediahandoutprogress_tl {
815       \pgfmathsetmacro \progress { \fp_use:N \l_@@_mediapdfprogress_fp }
816     }{
817       \regex_split:nVN { , } { \l_@@_mediahandoutprogress_tl } \l_tmpa_seq
818       \seq_map_inline:Nn \l_tmpa_seq {
819         \regex_split:nnN { / } { ##1 } \l_tmpb_seq
820         \int_compare:nT { \use:c {beamer@overlaynumber} == \seq_item:Nn \l_tmpb_seq {1} }
821         {
822           \pgfmathsetmacro\progress{ \seq_item:Nn \l_tmpb_seq {2} }
823         }
824       }
825     }
826   }
827   { \pgfmathsetmacro \progress { \fp_use:N \l_@@_mediapdfprogress_fp } }
828
829   % now set the content in a box and measure it
830   \hbox_set:Nn \l_tmpa_box {\tl_trim_spaces:n{#2}}
831   \pgfmathsetmacro\progress{ \fp_use:N \l_@@_mediapdfprogress_fp }%
832   \hbox_set:Nn \l_tmpa_box {\tl_trim_spaces:n{#3}}
833   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \box_wd:N \l_tmpa_box } }
834   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pagewidth } }
835   \fp_set:Nn \l_@@_mediawidth_fp { \fp_eval:n { \l_tmpa_fp / \l_tmpb_fp } }
836   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \box_ht:N \l_tmpa_box } }
837   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pageheight } }
838   \fp_set:Nn \l_@@_mediaheight_fp { \fp_eval:n { \l_tmpa_fp / \l_tmpb_fp } }
839
840   % extract relative position from bottom left of page (0,0) to top right (1,1)
841   \seq_set_split:Nnn \l_tmpa_seq { , } { #2 }
842   \pgfmathsetmacro{\posx}{\seq_item:Nn \l_tmpa_seq {1}}
843   \pgfmathsetmacro{\posy}{\seq_item:Nn \l_tmpa_seq {2}}
844
845   % convert to html coordinates from top left corner, and correct for anchor location
846   \fp_set:Nn \l_@@_xpos_fp { \posx + \l_@@_xposdelta_fp * \l_@@_mediawidth_fp }
847   \fp_set:Nn \l_@@_ypos_fp { (1-\posy) + \l_@@_yposdelta_fp * \l_@@_mediaheight_fp }
848
849   % write .rvl information
850   \writeliteral_@@:n {
851     -#4:
852     width={ \fp_use:N \l_@@_mediawidth_fp },
853     height={ \fp_use:N \l_@@_mediaheight_fp },
854     framerate={ \fp_use:N \l_@@_mediaframerate_fp },
855     duration={ \fp_use:N \l_@@_mediaduration_fp },
856     progress={ \fp_use:N \l_@@_mediapdfprogress_fp },
857     \bool_if:NTF \l_@@_mediaembed_bool {embed={},} {}
858     x={\fp_use:N \l_@@_xpos_fp},
859     y={\fp_use:N \l_@@_ypos_fp},
860     fit={fit},
861     background={\tl_use:N \l_@@_mediabackground_tl},
862     \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={},} {}
863     \bool_if:NTF \l_@@_medialoop_bool {loop={},} {}
864     \bool_if:NTF \l_@@_mediacontrols_bool {controls={},} {}
```



```

865 }
866
867 % put node in PDF files
868 \begin{tikzpicture}[overlay,remember~picture,font=\tiny]%
869   \@@extractlloleft{$(current~page.south-west)$}
870   \@@extractupright{$(current~page.north-east)$}
871   \node[
872     outer~sep=0pt,inner~sep=0pt,rectangle,
873     \bool_if:NTF \l_@@_mediaboxdraw_bool {draw}{},
874   ]
875     (#1) at ({\xloleft*(1-\posx)+\xupright*\posx},{\yloleft*(1-\posy)+\yupright*\posy})
876     {\box_use:N \l_tmpa_box};
877 \end{tikzpicture}%
878
879
880 %
881 % write code to .rvl file
882 \bool_if:NTF \l_@@_mediaexpandonce_bool
883   { \writeraw_@@:o { #3 } } { \writeraw_@@:n { #3 } }
884 }
885 </reveal>

```

`\insertFixedContent_@@:nnn`

```

886 <*reveal>
887 \cs_new:Npn \insertFixedContent_@@:nnn #1#2#3 {
888   %% % first create the box with the content and measure it
889   % check if we are in handout mode
890   \regex_split:nVN {\handout:} { \l_@@_mediaprogress_tl } \l_tmpa_seq
891   \fp_set:Nn \l_@@_mediapdfprogress_fp { \seq_item:Nn \l_tmpa_seq {1} }
892   \tl_set:Nx \l_@@_mediahandoutprogress_tl { \seq_item:Nn \l_tmpa_seq {2} }
893   \str_if_eq:eeTF { \use:c { beamer@currentmode } } { handout }
894   {
895     \tl_if_empty:NTF \l_@@_mediahandoutprogress_tl {
896       \pgfmathsetmacro \progress { \fp_use:N \l_@@_mediapdfprogress_fp }
897     }{
898       \regex_split:nVN { , } { \l_@@_mediahandoutprogress_tl } \l_tmpa_seq
899       \seq_map_inline:Nn \l_tmpa_seq {
900         \regex_split:nnN { / } { ##1 } \l_tmpb_seq
901         \int_compare:nT { \use:c {beamer@overlaynumber} == \seq_item:Nn \l_tmpb_seq {1} }
902         {
903           \pgfmathsetmacro\progress{ \seq_item:Nn \l_tmpb_seq {2} }
904         }
905       }
906     }
907   }
908   { \pgfmathsetmacro \progress { \fp_use:N \l_@@_mediapdfprogress_fp } }
909
910   % now set the content in a box and measure it
911   \hbox_set:Nn \l_tmpa_box {\tl_trim_spaces:n{#2}}
912   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \box_wd:N \l_tmpa_box } }
913   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pagewidth } }
914   \fp_set:Nn \l_@@_mediawidth_fp { \fp_eval:n { \l_tmpa_fp / \l_tmpb_fp } }
915   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \box_ht:N \l_tmpa_box } }
916   \fp_set:Nn \l_tmpb_fp { \dim_eval:n { \pageheight } }
917   \fp_set:Nn \l_@@_mediaheight_fp { \fp_eval:n { \l_tmpa_fp / \l_tmpb_fp } }
918   % restore the pagewidth in tmpa
919   \fp_set:Nn \l_tmpa_fp { \dim_eval:n { \pagewidth } }
920
921   % write node to PDF file

```

```

922 \begin{tikzpicture}[remember~picture,font=\tiny]
923   \node[
924     outer~sep=0pt,inner~sep=0pt,rectangle,
925     \bool_if:NTF \l_@@_mediaboxdraw_bool {draw}{},
926   ] (#1) {\box_use:N \l_tmpa_box};
927   \path let \p1=(\l_@@_media_north_west) - (current~page.north~west)$)
928   in \pgfextra{
929     \fp_set:Nn \l_@@_xpos_fp { \x1 / \fp_use:N \l_tmpa_fp }
930     \fp_set:Nn \l_@@_ypos_fp { - \y1 / \fp_use:N \l_tmpb_fp }
931     % write info to .rvl file
932     \writeliteral_@@:n {
933       -#3:
934       width={ \fp_use:N \l_@@_mediawidth_fp },
935       height={ \fp_use:N \l_@@_mediaheight_fp },
936       framerate={ \fp_use:N \l_@@_mediaframerate_fp },
937       duration={ \fp_use:N \l_@@_mediaduration_fp },
938       progress={ \fp_use:N \l_@@_mediapdfprogress_fp },
939       \bool_if:NTF \l_@@_mediaembed_bool {embed={},} {}
940       x={\fp_use:N \l_@@_xpos_fp},
941       y={\fp_use:N \l_@@_ypos_fp},
942       fit={fit},
943       background={\tl_use:N \l_@@_mediabackground_tl},
944       \bool_if:NTF \l_@@_mediaautoplay_bool {autoplay={},} {}
945       \bool_if:NTF \l_@@_medialoop_bool {loop={},} {}
946       \bool_if:NTF \l_@@_mediacontrols_bool {controls={},} {}
947     }
948   };
949 \end{tikzpicture}
950 \bool_if:NTF \l_@@_mediaexpandonce_bool
951   { \writeraw_@@:o { #2 } } { \writeraw_@@:n { #2 } }
952 }
953 \</reveal>

```

14.12 Postamble

```

954 \<reveal>
955 \AtEndDocument{
956   \ExplSyntaxOn
957   \iow_close:N \g_@@_rvlfile
958   \ExplSyntaxOff
959 }
960 \ExplSyntaxOff
961 \</reveal>

```

References

[1] Till Tantau et al., "beamer - A L^AT_EX class for producing presentations and slides", <https://ctan.org/pkg/beamer>, online, accessed in December 2025.

[2] Alexander Grahn, "movie15 - Multimedia inclusion package", <https://ctan.org/pkg/movie15>, online, accessed in December 2025.

[3] Alexander Grahn, "media9 - Multimedia inclusion package with Adobe Reader-9/X compatibility", <https://ctan.org/pkg/media9>, online, accessed in December 2025.

[4] Hakim El Hattab et al., "reveal.js - The HTML Presentation framework", <https://revealjs.com>, online, accessed in January 2026.

[5] Posit Software - PBC, "Quarto - An open-source scientific and technical publishing system", <https://quarto.org>, online, accessed in January 2026.

[6] Grant Sanderson, "manim - A community maintained Python library for creating mathematical animations", <https://www.manim.community>, online, accessed in December 2025.

[7] Heiko Oberdiek, "pdfcrop - Crop PDF graphics", <https://ctan.org/pkg/pdfcrop>, online, accessed in December 2025.

[8] Poppler developers, "Poppler Utilities (pdftoppm)", <https://poppler.freedesktop.org>, oline, accessed in December 2025.

[9] FFmpeg developers, "ffmpeg tool", <https://ffmpeg.org>, oline, accessed in December 2025.

[10] Amazon Polly — An online text-to-speech engine. <https://aws.amazon.com/polly> online, accessed in May 2024.

[11] Microsoft Azure TTS — An online text-to-speech engine. <https://azure.microsoft.com/en-us/products/ai-foundry/tools/speech/> online, accessed in April 2026.

[12] The Balabolka TTS-program. <http://www.cross-plus-a.com/balabolka.htm>. online, accessed in May 2024.

[13] The Festival TTS-program. <https://www.cstr.ed.ac.uk/projects/festival>. online, accessed in April 2026.

Change History

v0.80		v1.03	
General: alpha 1 embryonic demo version	1	General: new feature release	1
v0.85		- introduced new command-line options to	
General: alpha 2 tested by Walter, functional on Linux	1	accomodate users of latexmk	1
v0.90		v1.04	
General: beta 1 tested by Paul, not functional on		General: bug fix release	1
MS-Windows	1	- assigned mssing node name to rectangle	
v0.95		boundingbox of video/animation/image/audio	
General: beta 2 tested by Paul, functional on		such that positioning relative to the bounding box	
MS-Windows	1	(via its anchors) is possible	1
v1.00		v1.05	
General: first appearance on CTAN	1	General: new feature release	1
v1.01		- breaking syntax change: 'at' must be replaced by	
General: bug fix release	1	'\at'	1
- restored pdflatex compatibility (including		- implemented both overlay-mode and insert-mode	
macro-based accented characters)	1	media types	1
v1.02		- removed x-y distortion from animations	1
General: no changes - needed to correct faulty upload	1		

v1.06		- implemented stills as a complementing feature to animations	1
General: new feature release	1		
- implemented embedding of media files	1		
- improved animation generation space trimming and error logging	1		
- regular bugfixes	1		
- renamed extractloleft and extractupright to avoid nameclashes	1		
- reverted to short versions of frametitle for menu entries	1		
v1.07			
General: new feature release	1		
- allow for multiple handout slides per animation	1		
- enabled (full graphical) speaker notes	1		
v1.08			
General: mixed release	1		
- added pause key 'v' for video player and 'a' for audio player, 'i' to initialize/reset all videos/audios	1		
- extended embedding mode to also work for slide backgrounds and notes (not yet for animations and stills)	1		
- fixed notes being not only notes bug	1		
- implemented autoslide mode, together with loop mode, that allows for kiosk presentations	1		
		v1.09	
		General: mixed release	1
		- added --debug option to perl script	1
		- completed embedding mode, now everything except the quarto/reveal.js boilerplate can be embedded	1
		- fixed handout generation for fixed (size) content (animations and stills)	1
		- fixed problem of presentations without autoslide	1
		- removed requirement for stills to have a handout spec	1
		- scrutinized invalid option detection for media elements	1
		- simplified progress key for animations/stills	1
		- solved problem with non-uniqueness of reused animations and stills	1
		v1.10	
		General: new feature release	1
		- Autoplay voiceover using various tts services supported	1
		- Slides containing animations or audio are visibly marked (bottom right)	1
		- URL-style images/videos/audios/iframes now also work	1

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	<code>\insertFixedContent_{@:nnn}</code> . 885	T
<code>\@@extractloleft</code> 452, 456, 664, 752, 869	<code>\insertStaticContent</code> 531, 767	<code>\tikz</code> 223, 229, 235
<code>\@@extractupright</code> 457, 461, 665, 753, 870	<code>\insertStaticContent_{@:nnn}</code> . 765	<code>\TINY</code> 224, 230, 236
A	M	V
<code>\animation</code> 39, 310, 535, 537	<code>\mode</code> 219	<code>\video</code> 40, 304, 466, 469
<code>\at</code> 469, 485, 502, 520, 537, 555, 590, 609, 613, 619	O	<code>\voiceover</code> 570, 579
<code>\audio</code> 40, 483, 485, 590	<code>\overlayDynamicContent</code> 478, 495, 513, 631	W
B	<code>\overlayDynamicContent_{@:nnnn}</code> 629	<code>\writecomment</code> 163, 176
<code>\beamer@shortauthor</code> 190	<code>\overlayFixedContent</code> . 548, 565, 807	<code>\writecomment_{@:n}</code> 161
<code>\beamer@shortframetitle</code> . . 266, 267	<code>\overlayFixedContent_{@:nnnn}</code> 805	<code>\writecontrol</code> 167, 178, 216
<code>\beamer@shorttitle</code> 262	<code>\overlayStaticContent</code> 530, 722	<code>\writecontrol_{@:n}</code> 165
C	<code>\overlayStaticContent_{@:nnnn}</code> 720	<code>\writeliteral</code> . 169, 185, 242, 275, 582, 645, 703, 736, 791, 850, 932
<code>\contentDispatcher</code> 477, 494, 512, 529, 547, 564, 611	P	<code>\writeliteral_{@:n}</code> 168
<code>\contentDispatcher_{@:nnnnn}</code> . 609	<code>\process</code> 427, 634, 683, 725, 770	<code>\writeraw</code> 171, 174, 883, 951
I	<code>\process_a_w_h_{@:NNN}</code> . . . 425	<code>\writeraw_{@:n}</code> 170, 172
<code>\iframe</code> 34, 40, 308, 500, 502	<code>\progress</code> 815, 822, 827, 831, 896, 903, 908	X
<code>\image</code> 34, 35, 40, 306, 518, 520	S	<code>\xloleft</code> 454, 456, 668, 672, 756, 760, 875
<code>\insertDynamicContent</code> 479, 496, 514, 679	<code>\setvoiceover</code> 604, 606	<code>\xupright</code> 459, 461, 668, 672, 756, 760, 875
<code>\insertDynamicContent_{@:nnn}</code> 677	<code>\setvoiceoverext</code> 599, 601	Y
<code>\insertFixedContent</code> . 549, 566, 887	<code>\setvoiceovertts</code> 594, 596	<code>\yloleft</code> 455, 456, 669, 672, 757, 760, 875
	<code>\still</code> 553, 555	<code>\yupright</code> 460, 461, 669, 672, 757, 760, 875