LectureDoc2 Cheat Sheet for Slides using the DHBW Corporate Design

Information

Author: Michael Eichberg
Version: 2024-02-12

Vertical Titles

Add the class `vertical-title`. This will change the layout of the slide to a column-based layout. To get back to a row based layout add a container with the class `width-100`.

Example

```
.. class:: vertical-title

<Slide Title>
----------------

.. container:: width-100

   ...
```

Slide Tweaks

Slide without Title

For a slide without a title set the title to a single space character using a backslash and an explicit space `\␣` and assign the class `no-title` to the slide.

Smaller Slide Titles

Add the class *smaller-slide-title*` to the slide.

Centered Content on Slide

Use the class `center-child-elements`.

Example

```
.. class:: center-child-elements
           no-title

\
___

   Text
```

Sections and Subsections

Create a slide that marks the beginning of a new section or subsection by adding the class `new-section` or `new-subsection` to the slide.

Exercises and Solutions

Create a slide with exercises by adding the class `integrated-exercise` to the slide. Solutions can be added by using the custom directive `protected-exercise-solution`

Example (Solution in supplemental information)

```
.. admonition:: Solution
   :class: supplemental
           exercise-solution
```

Example (Solution requires password.)

```
.. class:: integrated-exercise

Exercise: XYZ
--------------

Calculate uvw...

.. protected-exercise-solution:: S

   The result is ...
```

Table of Contents

A navigable table of contents (e.g. of the section slides) can be manually created by referencing the titles.

Example

```
Table of Contents
------------------

- `Section 1 Title`_
- `Subsection 1.1 Title`_
```

Footnotes

`.. [#]` and `[#]_` create footnotes.

```
Test\ [#]_
-----------------

...

.. [#] `test.org`
```

Explicit Footers

Use a container with the class `footer-left` or `footer-right`.

References

Use standard rst references.

Example

```
...
Like described in [Eic24]_ ...
...

References
-----------

.. [Eic24] LectureDoc2; 2024
```

Copy to Clipboard

To make it easily possible to copy code the clipboard add the class `copy-to-clipboard` to the code block.

Example

```
.. code:: java
   :class: copy-to-clipboard

   public static void main(...)
```

Fade-out Content

Use the class `faded-to-white` for the container with the content that should be faded out.

Boxes with Supplemental Information on the Slide

Example

```
.. admonition:: TBD
   :class: note

   Some text in a box.
```

Supplemental Information

Add a container with the class `supplemental` to add respective information. How this information is rendered depends on the chosen view.

Example

```
.. container:: supplemental

   Text
```

Text Alignment

Control text alignment: `text-align-[left|center|right]`

Images

Adding a drop-shadow and rounded corners: `picture`.

Tables

The layout can be adapted using: `compact`, `compact-cells`, `no-table-borders`, `no-inner-borders`, `no-column-borders`, `fake-header[-2nd]-row` and `fake-header[-2nd]-column`.

Animation

`incremental` (and `wobble`).

`highlight-line-on-hover` (always usable), `highlight-on-hover` (explicit column or row headers are not supported) or `highlight-identical-cells`

Lists

- Use `li-margin-top-0-75em` to have more space between the list items.
- `list-with-explanations` renders text paragraphs of list items less pronounced.

  (As shown here.)
- Use `impressive` to make the list more impressive:

Add `negative-list` to use " ❗ " for bullet points.

Add `positive-list` to use "✓" for bullet points.

Example

```
- Point 1

.. class:: negative-list list-with
- Point 2
  Some on-slide explanation.

.. class:: positive-list
- Point 3
- Point 4
```

Decorations

`line-above` draws a horizontal lines.

`box-shadow` adds a shadow.

`rounded-corners` the corners will be rounded.

Example

```
.. container:: margin-top-1em
               line-above
               padding-top-1em
               box-shadow

   Text
```

Font Styling

"rem" based relative sizes: `xxl`, `huge`, `large`, `small`, `footnotesize`, `scriptsize`, `tiny`, `x-tiny`, `xx-tiny`

"em" based relative sizes: `larger`, `smaller`, `much-smaller`

Font weight: `bold`, `light`, `thin`

Font family: `monospaced`, `serif`

Font style: `italic`

Slide Transitions

Available slide transitions: `transition-move-left`, `transition-scale`, `transition-fade`, `transition-move-to-top`

Example

```
.. class:: transition-move-left

<Slide Title>
----------------
```

Revealing Slide Content

All elements with the class `incremental` are revealed incrementally.

Example

```
.. class:: incremental

- Item 1 - Part 1
  :incremental:`Item 1 - Part 2`
- Item 2

  - Item 2.1

    .. class:: incremental
  - Item 2.2
```

Column-based Layouts

We support 2- (`two-columns`) and 3-column (`three-columns`) layouts based on nested rst `container`s for each column.

Example

```
.. container:: two-columns

   .. container:: column

      Column 1

   .. container:: column

      Column 2
```

To enable unbalanced column widths add the class `no-default-width` to the root container. To remove the separator between two columns use the class `no-separator` on the left column.

Stacked Layouts

Stacked layouts are based on nested rst `container`s for each layer. In general. each layer - except the first one - needs to have the class `incremental`. If a new layer should be transparent; e.g., to incrementally build up an image, add the class `overlay` to the layer. (Currently, up to 10 layers are supported (CSS Limitation).)

Images in Stacked Layouts

To avoid that a parent element of a floating element is collapsed, add the class `clearfix` to the parent element. This is in particular necessary when you use a stacked layout where an element of a layer is a floating image.

Example

```
.. container:: stack

  .. container:: layer clearfix

     .. image:: <p1.svg>
        :align: left

  .. container:: layer overlay

     .. image:: <p2.svg>
        :align: left

   .. container:: layer
                  incremental

      Important!
```

Semantic-based Text Markup

`minor`: for less important text. `obsolete`: for obsolete statements. `ger`: to markup German Words. `eng`: t markup English words. `ger-quote`: Uses German quotation marks.

Box sizes

Use `width-100%` and `width-75%` t control the width of a container.

Colors (`roles`)

Font Colors

DHBW Colors: `dhbw-red`, `dhbw-gray` `dhbw-light-gray`

DHBW Compatible Colors: `the-blue`, `the-green`, `the-orange`

Other: `black`, `shiny-green`, `shiny-red`, `dark-red`

Background Colors

DHBW Colors: `dhbw-red-background`, `dhbw-gray-background`, `dhbw-light-gray-background`

DHBW Compatible Colors: `the-blue-background`, `the-green-background`, `the-yellow-background`

Other: `light-green-background`, `white-background`

Example

```
:dhbw-red:`Red Text.`
```

Controlling Whitespace

Adding space around an element (in particular images): `border-transparent-1em`

Fine-grained Control (Try to avoid!)

`margin-none`, `margin-0-5em`, `margin-1em`, `margin-top-1em`, `margin-top-2em`, `margin-bottom-1em`, `margin-bottom-2em`, `margin-right-1em`, `margin-left-1em`, `padding-none`, `padding-0-5em`, `padding-1em`, `padding-top-1em`, `padding-top-2em`