$LectureDoc^2$ Tutorial

LectureDoc is an authoring system for creating lecture slides/notes/exercises. LectureDoc enables you to write a single (HTML or) reStructuredText document that contains the slides, additional annotations and also exercises.

This tutorial is written in reStructuredText and can be used as a template for creating your own lecture slides.

Dr. Michael Eichberg

DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

Basics

A basic slide consists of a (section) header and some reStructuredText content.

Example

```
Basics
-----------

A basic slide consists of a (section) header
and some reStructuredText content.
```

Embedding Formulae

Embed math equations using reStructuredText's default directive (`.. math::`) and role (`:math:`...``).

Example

The following code:

```
Computation in :math:`GF(2)`:

.. math::

  \begin{matrix}
  1 + 1 & = 1 - 1 & = 0 \\
  1 + 0 & = 1 - 0 & = 1 \\
  0 + 1 & = 0 - 1 & = 1
  \end{matrix}
```

Will render like this:

Computation in $GF(2)$:

$$\begin{matrix}
1 + 1 & = 1 - 1 & = 0 \\
1 + 0 & = 1 - 0 & = 1 \\
0 + 1 & = 0 - 1 & = 1
\end{matrix}$$

A slide without an explicit title can be created by explicitly creating an empty title:

Example

```
\
--
```

Note

You have to add a space after the backslash (\)!

Animation

Basic *appear* animations can be created using the (CSS) class `incremental`. You can also define a corresponding custom role (`.. role:: incremental`) to animate parts of a text.

Example

```
Animation
----------

Basic *appear* animations can be created using the (CSS) class
``incremental``. You can also define a corresponding custom role
(``.. role:: incremental``) :incremental:`to animate parts of a text.`

.. admonition:: Example
    :class: incremental

    ...
```

Animation of Lists

In case of lists (*ol* or *ul*) it is sufficient to specify *incremental* in the class attribute of *ol* or *ul*; it is also possible, to only specify the class attribute for the required list elements.

Example

```
..class:: incremental

- this
- is
- a test
```

Slide Dimensions

The slide dimensions can be controlled by specifying the corresponding meta information. If not specified, the default dimension is set to 1920×1200; i.e., a ratio of 16:10.

Example

In HTML documents add at the following meta tag:

```
<meta name="slide-dimensions" content="1600x1200">
```

In reStructuredText documents add at the beginning:

```
.. meta::
    :slide-dimensions: 1600x1200
```

Adding Supplemental Information

Adding information that should not be on the slides, but provide additional information, can be added using a container at the root level in combination with the class `supplemental`.

Example

```
.. container:: supplemental

    **Formatting Slides**

    Creating heavily formatted slides is easily possible
    using rst directives and roles which are mapped to
    CSS classes.
```

Formatting Slides

Creating heavily formatted slides is easily possible using rst directives and roles which are mapped to CSS classes.

Creating Section Marker Slides

Creating a slide which marks the beginning of a new section can be done using the "new-section" class.

Example

```
.. class:: new-section

<Title of Section>
------------------


...

<Title of next Slide>
---------------------
```

Adding Code

Adding code can be done using reStructuredText's code directive.

Example

The following code:

```
.. code:: python

    for i in range(0,10):
        print(i)
```

Will render like this:

```python
for i in range(0,10):
  print(i)
```

Advanced Formatting

LectureDoc comes with a set of predefined CSS classes that can be used to format the slides. Some of these classes have explicit support by LectureDoc and will be rendered differently in the different views (continuous view vs. slide view) (e.g., stacked layouts or supplemental information). See the *LectureDoc2* Cheat Sheet for a comprehensive list of predefined CSS classes.