Einführung in das Textsatzsystem

LaTeX

09 – Bildschirmpräsentationen

Lion Lecturer

8. Januar 2014

Inhalt

1 Die beamer-Klasse

2 Multimedia

3 Hintergrundinformation: pgf

4 Präsentationssoftware

Vorbemerkungen

- LaTeX ist *nicht* für Präsentationen geschaffen
- spezielle Programme oft besser geeignet
- Wahl des Programms vom Inhalt abhängig:
- bei strukturierter, klarer Darstellungsform: LaTeX mit `beamer`-Klasse

Beamer

- Dokumentklasse `beamer` ermöglich Satz von Präsentationen
- erstellt bildschirmfüllende „Folien“ (pdfs)
- ansprechende Farbgebung
- strukturierte Darstellung des Inhaltes
- „dynamische“ Effekte (Ein / Ausblenden)
- multimediale Unterstützung
- wirklich dynamische Übergangseffekte mittels pdf-Features möglich!

Aufbau einer Präsentation

- `\documentclass{beamer}`
- alle Pakete, Befehle, Umgebungen (fast) wie normal zu verwenden:
- `\tableofcontents` erzeugt Inhaltsverzeichnis (nur `\part`-weise!)
- `\begin{tabular}` setzt Tabelle etc.
- wichtigste Umgebung:
  `\begin{frame}`
  (setzt jeweils einzelne Folien, auf mehreren pdf-Seiten)
  `\end{frame}`
- Abkürzung: `\frame{}`

Aufbau einer Präsentation

- `\documentclass{beamer}`
- alle Pakete, Befehle, Umgebungen (fast) wie normal zu verwenden:
- `\tableofcontents` erzeugt Inhaltsverzeichnis (nur `\part`-weise!)
- `\begin{tabular}` setzt Tabelle etc.
- wichtigste Umgebung:
  `\begin{frame}`
  (setzt jeweils einzelne Folien, auf mehreren pdf-Seiten)
  `\end{frame}`
- Abkürzung: `\frame{}`
- Eliminieren der (eher störunden) Navigationsleiste mittels
  `\setbeamertemplate{navigation symbols}{}`
  im Header

Besonderheiten
frames

- Umgebung `frame` erzeugt eine „Folie“
- erstes Argument: Titel, zweites: Untertitel
  (beide mit `{}`, aber optional!)
- optionales Argument `[fragile]` nötig, falls `\verb` u. ä. verwendet wird
- eine einzelne pdf-Seite ist ein statisches Objekt (ohne Scripte)

$\Longrightarrow$ Überblendeffekte benötigen mehrere pdf-Seiten!

Beispielframe

Vertikale Ausrichtung mittels optionalem Argument `[t,b,c]`, auch als Klassenoption

```
\begin{frame}[fragile,t]{Thema}{Unterthema}
    Folieninhalt mit \verb|\LaTeX-Befehlen|
\end{frame}
```

Überblendeffekte

- für dynamische Effekte: <kürzel> in spitzen Klammern
- <+-> lässt Objekt erscheinen und bleibt

Überblendeffekte

- für dynamische Effekte: <kürzel> in spitzen Klammern
- <+-> lässt Objekt erscheinen und bleibt
- <+> lässt Objekt einmalig erscheinen

Überblendeffekte

- für dynamische Effekte: <kürzel> in spitzen Klammern
- <+-> lässt Objekt erscheinen und bleibt

Überblendeffekte

- für dynamische Effekte: <kürzel> in spitzen Klammern
- <+-> lässt Objekt erscheinen und bleibt

- <4> Objekt erscheint auf Version 4 der Folie

Überblendeffekte

- z. B. bei `itemize`:

```
\begin{itemize}[<+->]  % Angabe gilt für alle \items
\item Punkt 1
\item Punkt 2
\item Punkt 3
\item<-2> Punkt 4
\end{itemize}
```

Auch bei `\includegraphics<+->` und anderen Elementen

Überblendeffekte

Pause

- `\pause` stoppt den Inhalt an beliebiger Stelle
- erste Seite wird bis zu `\pause` gesetzt
- zweite Seite enthält den gesamten Inhalt (bis zum nächsten `\pause`)

$$a =$$

Überblendeffekte

Pause

- `\pause` stoppt den Inhalt an beliebiger Stelle
- erste Seite wird bis zu `\pause` gesetzt
- zweite Seite enthält den gesamten Inhalt (bis zum nächsten `\pause`)

$$a = b_c$$

Überblendeffekte

Pause

- `\pause` stoppt den Inhalt an beliebiger Stelle
- erste Seite wird bis zu `\pause` gesetzt
- zweite Seite enthält den gesamten Inhalt (bis zum nächsten `\pause`)

$$a = b_{c \cdot d}$$

Überblendeffekte

only

- `\only<kürzel>{inhalt}` setzt den `inhalt` nur in den angegeben Seiten
- Platz für den `inhalt` wird *nicht* freigehalten
- `\only<4>{inhalt}` setzt nur in der vierten Seite
- `\only<3->{inhalt}` setzt ab der dritten Seite

Überblendeffekte

un(der)cover

- `\uncover<kürzel>{inhalt}` setzt den `inhalt` nur in den angegeben Seiten
- Platz für den `inhalt` *wird* freigehalten
- `\uncover<4>{inhalt}` setzt nur in der vierten Seite
- `\uncover<3->{inhalt}` setzt ab der dritten Seite

Überblendeffekte
dynamisch

- pdf-Spezifikation definiert standardasierte dynamische Übergänge
- nur mit pdfTeX bzw. LuaTeX möglich!
- nicht von allen Viewern unterstützt! (möglich: Acrobat Reader, okular)
- siehe `beamer`-Dokumentation, `14.3 Slide Transitions`

themes
allgemeine

- themes sind Stilvorlagen, die das gesamte Layout beeinflussen
- Einbinden mittels `\usetheme` im Header
- benannt nach Tagungsorten
- siehe `beamer`-Dokumentation, `15 Themes`

themes
inner

- beeinflussen das Aussehen von Elementen in der Folie
- Aufzählungen, Abbildungsbeschriftung, Boxen etc.
- `\useinnertheme`

themes
outer

- beeinflussen das Aussehen der äußeren Element
- Kopfzeile, Fußzeile, Navigation etc.
- `\useoutertheme`

themes
color

- wie der Name sagt ...
- je nach Theme werden verschiedene Elemente coloriert
- Farben für jedes Element anpassbar:
- `\setbeamercolor{footnote}{fg=red}`
- `fg` für `foreground`, `bg` für `background`

Gliederung

- normale Gliederungselemente vorhanden
- `\section, \subsection, \chapter, ...`
- Angabe von `\section` bewirkt zunächst nichts!
  (Absatzüberschriften werden *nicht* ausgegeben)
- Einfluss nur in Inhaltsverzeichnissen und Headern

Strukturelemente
block

```
\begin{block}{Titel}
Inhalt eines schön
gefärbten Blockes.
\end{block}
\begin{block}{Zwei}
Und noch einer.
\end{block}
```

Titel

Inhalt eines schön gefärbten Blockes.

Zwei

Und noch einer.

Strukturelemente
theorem

```
\begin{theorem}[Trautmann
et al. 2010]
1 + 2 = 3
\end{theorem}
\begin{proof}
2 = 1+1\\
1+1+1 = 3
\end{proof}
\begin{example}
2+1 = 3
\end{example}
```

Satz (Trautmann et al. 2010)

1 + 2 = 3

Beweis.

2 = 1+1

1+1+1 = 3 □

Beispiel

2+1 = 3

Konflikt mit `theorem` aus `amsmath`!

Können nummeriert werden mit Dokumentenoption `[envcountsec]`

Gleitumgebungen

- Einfügen von Abbildungen, Tabellen u. ä. wie gewohnt
- Gleitumgebungen werden *nicht* nummeriert
- Positionsangaben (h,t,b) werden ignoriert
- `\logo` fügt ein Logo global in die Präsentation ein (z. B. oben links)
- Bilder einfügen mittels `\includegraphics` oder mittels `\pgfdeclareimage` und -`useimage`

```
\pgfdeclareimage[height=0.5cm]{logo}{tu-logo}
\logo{\pgfuseimage{logo}}
\logo{\includegraphics[height=0.5cm]{logo}{tu-logo}}
```

Farben

- LaTeX bietet keinen nativen Farbmechanismus an
- Paket `xcolor` erlaubt einfachen Zugriff auf verschiedene Farbmuster
- einige Paket verwenden implizit Farbangaben (z. B. `hyperref`, natürlich `beamer`)

Farben

- LaTeX bietet keinen nativen Farbmechanismus an
- Paket `xcolor` erlaubt einfachen Zugriff auf verschiedene Farbmuster
- einige Paket verwenden implizit Farbangaben (z. B. `hyperref`, natürlich `beamer`)
- Umstellen der Farbel mittels: `\color{red}`
- Hervorheben einzelner Textstellen: `\textcolor{red}{Test}`
- Farbige Boxen: `\colorbox`, `\fcolorbox`
- Verschiedene Farbmodellen möglich: rgb, cmyk, hsb, wave, ...

$\Longrightarrow$ siehe `xcolor`-Dokumentation

Filme

- Paket `multimedia` (gehört zu beamer) laden
- unter Verwendung von pdfLaTeX und geeignetem Viewer: Einbinden von Videos möglich

Animationen

- Paket `animations` bietet Möglichkeit, Animationen ins pdf einzubinden
- Viewer muss dieses feature unterstüzen!

pgf

- `beamer` baut auf `pgf` auf
- `pgf`: portable graphics format (oder „pretty, good, functional“)
- `pgf` besteht aus drei verschiedenen Ebenen:

1. Systemebene
2. Basisebene
3. Frontend (Nutzerebene)

pgf
Systemebene

- Abstraktion von Treibern
- Unabhängigkeit von genauen Abläufen der Treiber
- Portabilität, Stabilität, leichte Erweiterung auf neue Treiber
- unterschiedliche `\special`-Befehle je nach Treiber
- so minimalistisch wie möglich (jeder Befehl muss im Treiber umgesetzt werden)
- kann z. B. keine Kreise, nur Bézier-Kurven
- Nutzer muss sich nicht um Treiberabhängigkeiten kümmern

pgf
Basisebene

- bietet Basisbefehle (z. B. Befehl für Kreis)
- besteht aus verschiedenen Modulen:
- *core*, bietet die Grundfunktionalität (mehrere Module, die zusammen benötigt werden)
- weitere optionale Module (node management, plotting …)

pgf
Frontend (Nutzerebene)

- Vereinfacht die Benutzung der Basisebene (vgl. Makropaket für TeX)
- Ti*k*Z ist die normale Nutzerebene von `pgf`
- `pgfpict2e` ist eine Reimplementierung von LaTeXs `{picture}`-Umgebung
- `beamer` ist eine spezialisierte Nutzerebene

Präsentationssoftware

Kriterien für eine gute Präsentationssoftware

- fullscreen-Modus
- Bedienung mit Tastatur und Maus möglich
- schwärzen / weißen des Schirms
- schnelle Navigation zwischen Folien
- Implementierung aller pdf-Features
- Kennzeichnungen / Hervorhebungen während der Präsentation
- eigene Überblendmechanismen
- kein Blockieren des pdfs!

TeXworks

- frei verfügbar (= offener Quellcode)
- hervorragender Editor mit eingebautem Viewer
- nötige Änderungen in der Präsentation können on-the-fly vorgenommen werden
- syncTeX bereitet mit beamer große Probleme!
- nicht alle pdf-features vorhanden

Adobe Acrobat Reader

- kostenlose Software
- nicht *frei* (im Sinne von offenem Quellcode)
- für Windows, Mac, Linux verfügbar
- implementiert sämtliche pdf-Features (z. B. Videos möglich)
- bietet einige Präsentationsfeatures (Bildschirm schwarz / weiß etc.)
- blockiert das pdf!

okular

- vielfältiger Viewer
- implementiert (scheinbar) alle pdf-features
  (kann Videos abspielen, Transitions etc)
- zuverlässiger als Acrobat Reader (persönlicher Eindruck!)

impressive!

- speziell für Präsentationen erstellt
- freie Software ($\Longrightarrow$ für alle Platformen verfügbar)
- Start aus Kommandozeile
- Effekte nur über Kommandozeilenargumente steuerbar!
- ermöglicht nützliche Präsentationseffekte: Schirm schwärzen, Spotlight, helle Rahmen ziehen, schnelle Navigation …

impressive!

- speziell für Präsentationen erstellt
- freie Software ($\Longrightarrow$ für alle Platformen verfügbar)
- Start aus Kommandozeile
- Effekte nur über Kommandozeilenargumente steuerbar!
- ermöglicht nützliche Präsentationseffekte: Schirm schwärzen, Spotlight, helle Rahmen ziehen, schnelle Navigation …
- Bereitet eventuell Probleme bei Dual-Screen!

Alternativen

- `lecturer` erfordert nur plainTeX, funktioniert auch mit LaTeX (`\usepackage{lecturer}`) und ConTeXt

- `powerdot` ist eine LaTeX-Klasse, die auf PSTricks basiert, daher nur im dvi-Mode verwendbar!

$\Rightarrow$ nur mittels `latex` oder `dvilualatex` verwendbar

- Umwandlung in pdf dann mittels `dvipdf` oder `dvips` und `ps2pdf` möglich.