LaTeX für Musiker

Michael Enzenhofer

Dezember 2016

Inhaltsverzeichnis

I. Der erste Einstieg 3

1. LaTeX 4
- 1.1. Geschichte von LaTeX . . . 4
- 1.2. Grundprinzip von LaTeX . . . 4
- 1.3. Finale versus LilyPond . . . 5
- 1.4. Word versus LaTeX . . . 7
- 1.5. Vorteile von LaTeX . . . 9

2. LaTeX-Editoren 11
- 2.1. Die Qual der Wahl . . . 11
- 2.2. TeXShop die Wahl für Mac OSX . . . 11
- 2.3. Editor – Viewer . . . 12

3. Erster Umgang mit LaTeX 13
- 3.1. Die Computertastatur . . . 13
- 3.2. LaTeX-Befehle . . . 13
- 3.3. LaTeX-Packages . . . 16
- 3.4. Der LaTeX-Quellcode . . . 17
- 3.5. Dokumentenklasse . . . 19
- 3.6. Gliederung des Dokumententeils . . . 20

4. Die musikalische Notation 21

5. Zusätzliche Pakete 26

6. Weitere Befehle 28

7. Bilder einfügen 31
- 7.1. \includegraphics . . . 31
- 7.2. Bild-Positionierung . . . 36
- 7.3. figure-Umgebung . . . 37
- 7.4. sidecap . . . 38
- 7.5. picinpar . . . 39

8. Linkempfehlungen 42

Teil I.

Der erste Einstieg

1. LaTeX

sprich: „Latech“

1.1. Geschichte von LaTeX

Das Basis-Programm von LaTeX ist TeX.

Dieses wurde von Donald Ervin Knuth während seiner Zeit als Informatik-Professor an der Stanford University von 1977 bis 1986 entwickelt.

Auf TeX aufbauend entwickelte Leslie Lamport Anfang der 1980er Jahre LaTeX.

Der Name LaTeX ist eine Abkürzung für Lamport TeX.

Lamports Entwicklung von LaTeX endete gegen 1990 mit der Version 2.09.

Die aktuelle Version LaTeX2 wurde ab 1989 von einer größeren Zahl von Autoren um Frank Mittelbach, Chris Rowley und Rainer Schöpf entwickelt.

1.2. Grundprinzip von LaTeX

Im Gegensatz zu anderen herkömmlichenTextverarbeitungsprogrammen, die nach dem What-You-See-is-What-You-Get-Prinzip (WYSIWYG) funktionieren, arbeitet man in LaTeX mit CODE. Mit Hilfe von Befehls-CODE werden zu formatierende Passagen oder Überschriften – beziehungsweise sämtliche Feinheiten – genau definiert.

Das Verfahren von LaTeXwird auch mit WYSIWYAF (What-You-See-Is-What-You-Asked-For) umschrieben – manchmal auch mit WYSIWYM (What-You-See-Is-What-You-Mean).

LaTeX ist eine sogenannte Auszeichnungssprache (englisch: markup language).

Diese ist eine maschinenlesbare Sprache für die Gliederung und Formatierung von Texten.

Die bekannteste Auszeichnungssprache ist die Hypertext Markup Language (HTML), die Kernsprache des World Wide Web.

Der Gedanke der Trennung von Inhalt und Form spielt hierbei eine wesentliche Rolle.

1.3. Finale versus LilyPond

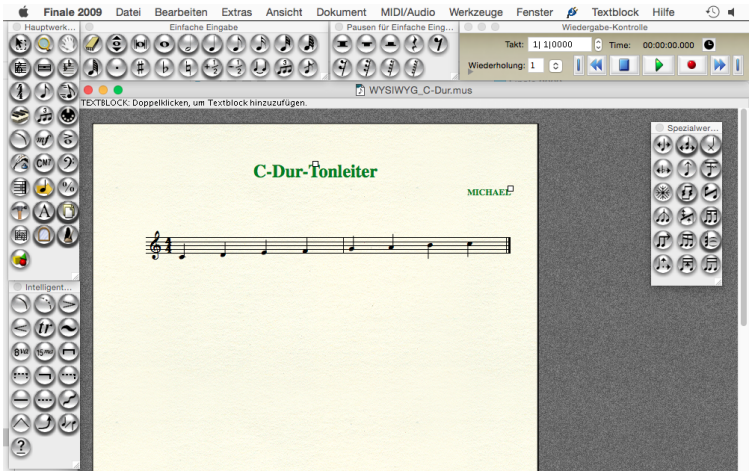


Abbildung 1.1.: WYSIWYG in Finale

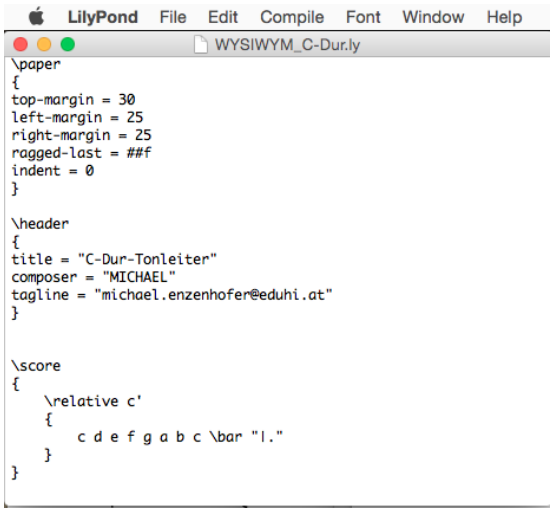
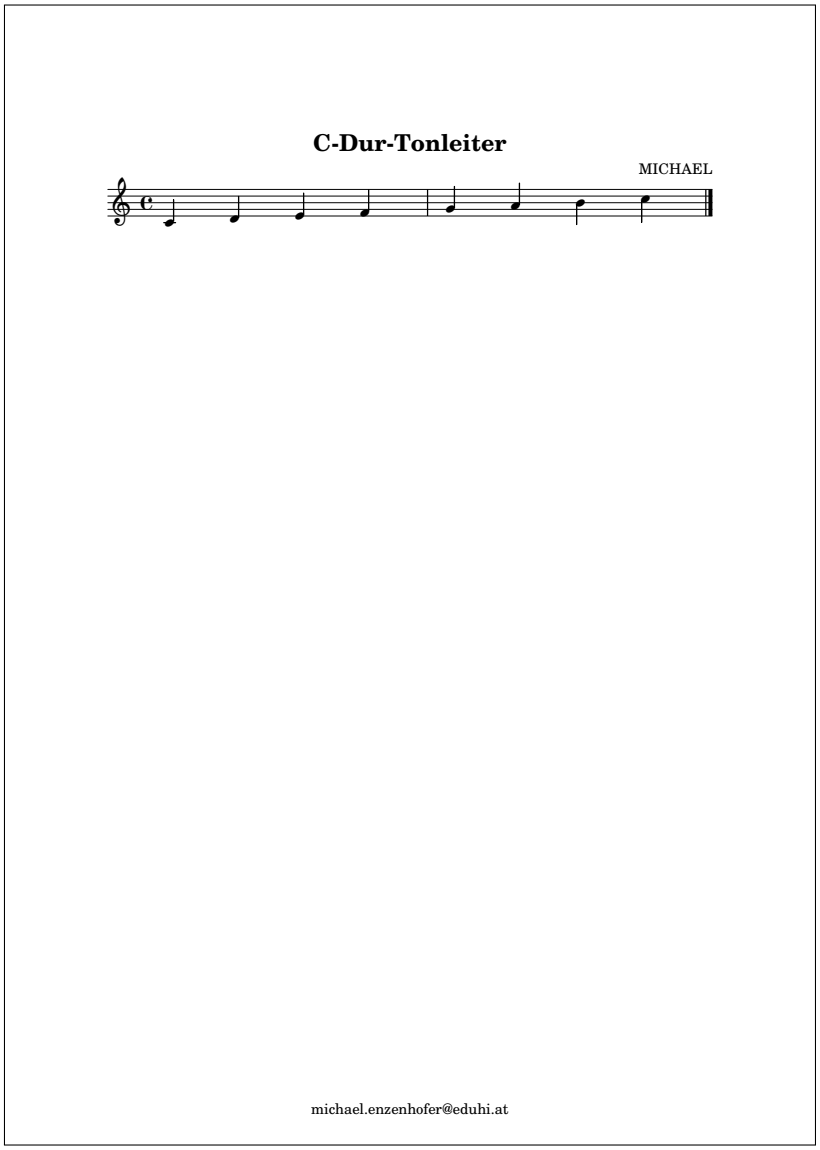


Abbildung 1.2.: WYSIWYM in LilyPond

Das Ergebnis aus LilyPond ist ein *Pdf*-File:



1.4. Word versus LaTeX

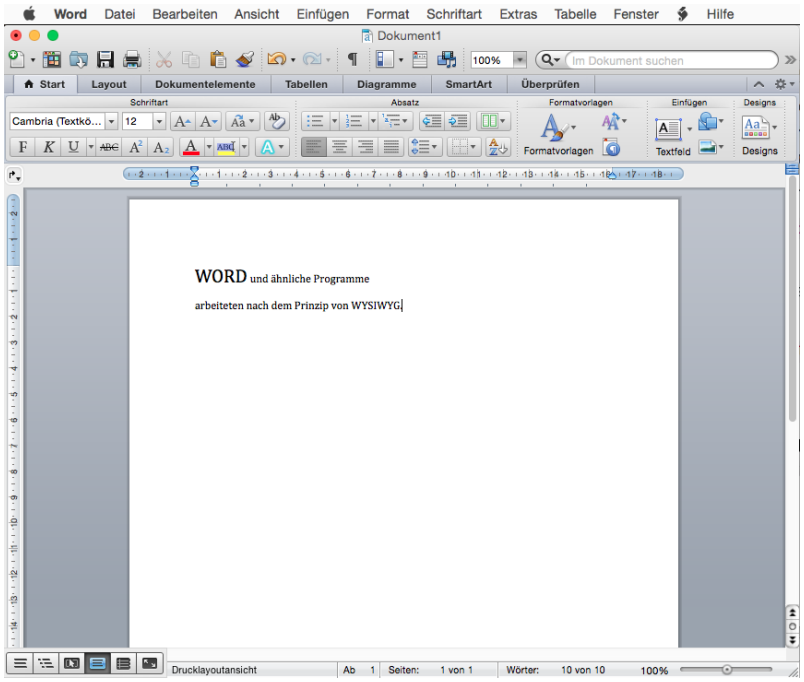


Abbildung 1.3.: WYSIWYG in Word

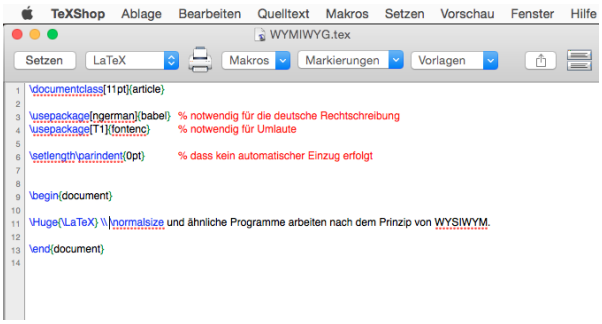
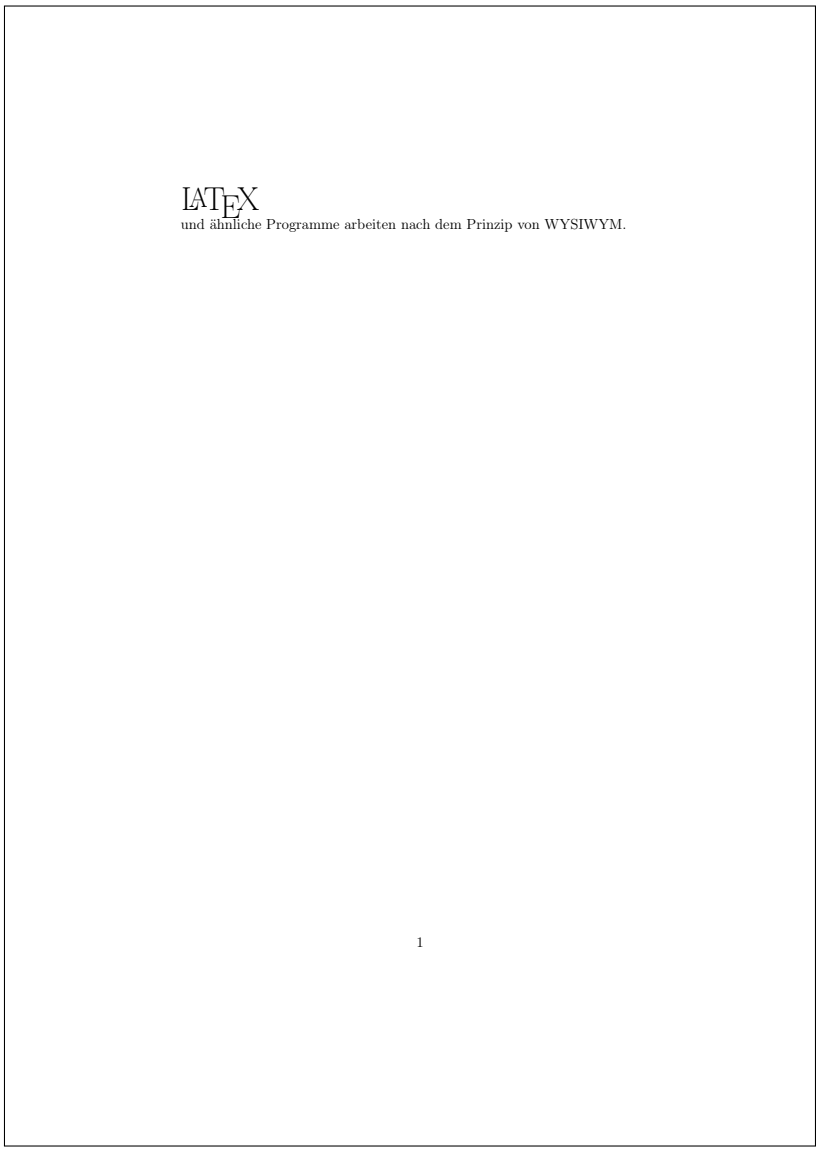


Abbildung 1.4.: WYSIWYM in LaTeX

Das Ergebnis aus LaTeXist ein *Pdf*-File:



1.5. Vorteile von LaTeX

Beim ersten Kennenlernen von LaTeX kann man wahrscheinlich nicht glauben, dass es Vorteile haben kann, sich auf diese andere Art Texte zu schreiben, einzulassen. Dafür sind Programme, wie Microsoft-Word geschaffen, wozu LaTeX?

Der erste Schein trügt hier.

Es wird nicht versucht mit optischem Aufputz das Programm unter die Leute zu bringen.

Jeder, der es nützt, wird bald seine Vorzüge zu schätzen wissen.

- LaTeX ist kostenfrei.
- LaTeX läuft auf allen gängigen Betriebssystem.
- LaTeX läuft auf alten und neuen Computersystemen.
- LaTeX bietet höchste typografische Qualität und optimales Seiten-Layout – auch für Textschreiber ohne Vorkenntinisse.
- Höchste Stabilität und Zuverlässigkeit des Programms ist gewährleistet.
- Sehr geringe Rechnerbelastung.
- Die Auf-und Abwärtskompatibilität der LaTeX-Dokumente ist über die Entwicklungs-Generationen bestens gewährleistet.
- Volle Anpassungsfähigkeit an die Bedürfnisse des Verfassers durch unzählige Erweiterungsmöglichkeiten.
- Ein eventuelles Software-Update beeinflusst nicht den gewohnten Workflow bei der Arbeit.
- Viele Anwender teilen ihr Wissen im WordWideWeb.
- In LaTeX sind Jahrhunderte von Erfahrung im Schriftsatz eingearbeitet.

Des Weiteren ist bemerkenswert:

- Es gibt keine Werkzeuge, die man suchen muss.
- Das Prinzip der Trennung von Inhalt und Form.
- Vom Text bis hin zu grafischen Darstellungen lässt sich alles auf der Tastatur erledigen, ohne auf die Computermaus zu wechseln.
- LilyPond-Code – also Notentext – lässt sich nahtlos in den LaTeX-CODE integrieren.

- Grafiken und Bilder lassen sich optimal in LaTeX-CODE integrieren.

Das Einbinden von musikalischer Notation oder Grafiken mittels CODE in LaTeX hat den großen Vorteil, dass bei eventuellen Fehlern in der Grafik oder im Notenbild nur der dazugehörige Code geändert werden muss. Es ist daher kein umständlicher Workaround notwendig, um Grafik oder Notation im Dokument zu ändern.

Wenn auch das Schreiben gewisser CODE-Schnipsel etwas gewöhnungsbedürftig ist, sind auch hier Vorteile zu erkennen:

- CODE kann kopiert werden – entweder
  - aus beliebigen Textdokumenten,
  - aus anderen Textabschnitten,
  - aus Seiten vom Internet oder
  - von eigenen CODE-Sammlungen.
- LaTeX-CODE ist White-Space-Insesitive. Das heißt, dass mehr als **ein** Leerzeichen im Quellcode ignoriert wird. Dieser Umstand führt auch zur individuellen Gestaltung des Quellcodes, weil auch Tabulatoren und Leerzeilen im Quellcode ignoriert werden.
- CODE lässt sich wiederverwenden.
- LaTeX-Quellcode lässt sich mit jedem Texteditor erzeugen.

2. LaTeX-Editoren

2.1. Die Qual der Wahl

Um in LaTeX den sogenannten Quelltext zu verfassen sind Programm üblich, die diese Aufgabe übernehmen.
Es werden sogenannte Editoren eingesetzt.
Die Frage nach dem richtigen Editor ist eine Frage des Geschmacks. Unterschiedliche Anwender haben unterschiedliche Voraussetzungen und unterschiedliche Vorlieben.
Hier eine kleine Auflistung von LaTeX-Editoren für die verschiedenen Betriebssysteme[1]:

TeXworks:	Freeware	Windows	OS X	Linux
Texmaker:	Freeware	Windows	OS X	Linux
TeXstudio:	Freeware	Windows	OS X	Linux
emacs:	Freeware	Windows	OS X	Linux
TeXnicCenter:	Freeware	Windows		
WinEdt:	Shareware	Windows		
TeXShop:	Freeware		OS X	

2.2. TeXShop die Wahl für Mac OSX

Um Lilypond-Code in LaTeX integrieren zu können, ist es vorteilhaft, wenn innerhalb des Editors LilyPond-Book als *Schriftsatzprogramm* ausgewählt werden kann. Damit steht dem Eingeben von LilyPond-CODE innerhalb von LaTeX nichts mehr im Wege.

[1] Hier ist eine vollständigere Auflistung zu finden:
https://en.wikipedia.org/wiki/Comparison_of_TeX_editors (Dezember 2016).

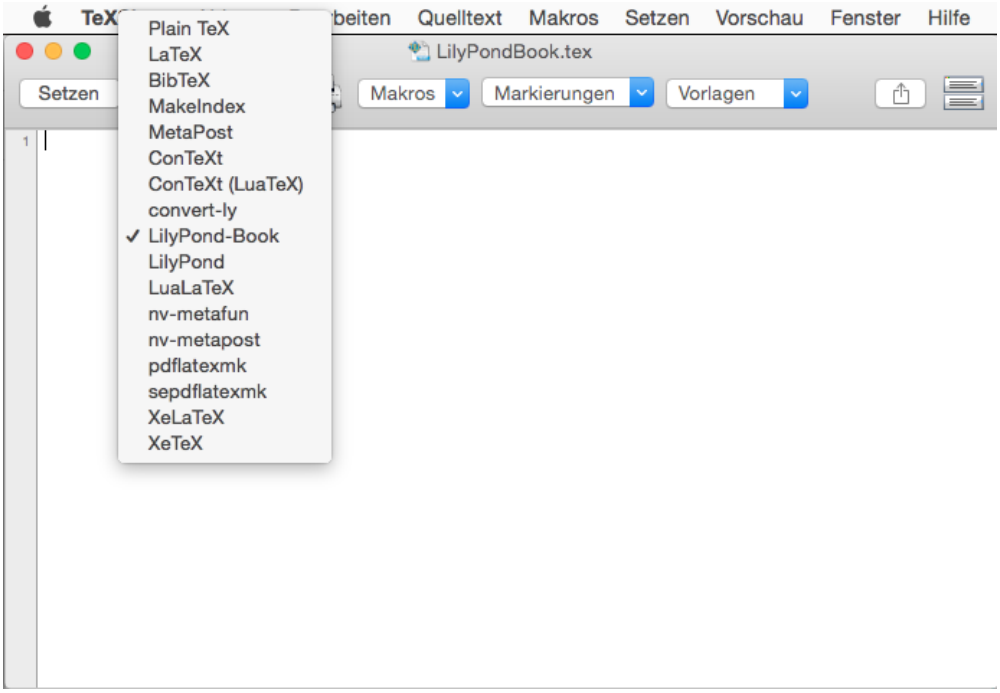


Abbildung 2.1.: Auswahl von LilyPond-Book in TeXshop

TeXShop ist in einer umfangreichen Distribution für Mac OSX – The MacTeX-2016 Distribution – hier[2] zu finden.

2.3. Editor – Viewer

Der Bereich, in dem der LaTeX-Quellcode geschrieben wird ist der Editor.

Beim drücken der Taste *Setzten* oder der Tastenkombination *cmd+T* wird der Quellcode in einem *Pdf*-Viewer angezeigt.

Um komfortabel bei einer Textstelle von einer Ansicht in die andere zu gelangen, klickt man am besten bei gedrückter *cmd*-Taste mit der Computermaus auf die gewünschte Textstelle.

[2] http://www.tug.org/mactex/(Dezember 2016).

3. Erster Umgang mit LaTeX

3.1. Die Computertastatur

Nachdem der LaTeX-Quellcode für das Dokument weitestgehend von der Computertastatur alleine ohne Zuhilfenahme der Maus geschrieben wird, ist der mühelose Umgang mit der Tastatur von großem Vorteil.

Einige Zeichen werden besonders oft gebraucht.

Zeichen	OSX	WIN
\ (Backslash)	SHIFT+alt+7	ALT GR + ß
/ (Slash)	SHIFT+7	SHIFT + 7
{ (Geschwungene Klammer auf)	alt+8	ALT GR + 7
} (Geschwungene Klammer zu)	alt+9	ALT GR + 0
[(Eckige Klammer auf)	alt+5	ALT GR + 8
] (Eckige Klammer zu)	alt+6	ALT GR + 9

Auch noch gut zu wissen sind:

~ (Tilde)	alt+n	ALT GR + +
| (Pipe)	alt+7	CTRL + ALT GR + ß

3.2. LaTeX-Befehle

Nachdem in LaTeX jede Kleinigkeit, die von der Text-Norm abweicht im Quell-Code festgelegt wird, müssen dafür Befehle eingesetzt werden.
Beispielsweise für:

Schriftart – Zeichensatz

Die voreingestellte Schrift – die sogenannte *Brotschrift*[1] – in LaTeX ist die von Donald E. Knuth in allen Serien und Formen entworfene Schriftart voreingestellt:
Computer Modern (CM Roman, CM Sansserif, CM Typewriter, usw.)

[1] Als Brotschrift bezeichneten Schriftsetzer die Schriftart, in welcher der Fließtext gesetzt wurde. Der Ursprung des Begriffs liegt darin, dass der Schriftsetzer mit der Brotschrift sein „tägliches Brot“ verdiente. Es wird auch der Begriff Werk- oder Grundschrift verwendet.

Andere Schriften müssen explizit ausgewählt werden ... darauf möchte ich hier nicht eingehen.

Schriftfamilie[2]

Befehl mit Beispiel:	Auswirkung
`\textbf{Fett}`	**Fett**
`\textit{Kursiv}`	*Kursiv*
`\emph{hervorgehoben}`	*hervorgehoben*
`\textsl{schief}`	*schief*
`\textsc{Kapitälchen}`	KAPITÄLCHEN
`\textsf{Sans Serif}`	Sans Serif
`\textrm{Roman}`	Roman
`\texttt{Schreibmaschine}`	`Schreibmaschine`
`\textnormal{Normale Schrift}`	Normale Schrift
`\underline{unterstrichen}`	unterstrichen

Der Schriftschnitt kann auch kombiniert werden:

`\textit{\textbf{Fett und kursiv}}` ***Fett und kursiv***

und so weiter ...

Schriftgröße

Mit folgenden Befehlen kann man innerhalb eines Dokuments die Schriftgröße ändern:

Befehl mit Beispiel:	Auswirkung
`\tiny{winzig}`	winzig
`\scriptsize{Skriptgröße}`	Skriptgröße
`\footnotesize{Fußnotengröße}`	Fußnotengröße
`\small{klein}`	klein
`\normalsize{normale Größe}`	normale Größe
`\large{groß}`	groß
`\Large{größer}`	größer
`\LARGE{noch größer}`	noch größer
`\huge{riesig}`	riesig
`\Huge{am riesigsten}`	am riesigsten

ACHTUNG!
Wenn die Schriftgröße in einem Dokument geändert wird und anschließend mit normaler Größe weitergeschrieben werden sollte, muss wieder auf `\normalsize` umgestellt werden!

[2] Schriftfamilie bedeutet die Gesamtheit der Schnitte einer Schriftart

Anführungszeichen

„ (Anführungszeichen unten) werden mit `\glqq{}` und
“ (Anführungszeichen oben) mit `\grqq{}` notiert.

Zeilenschaltung

Eine Zeilenschaltung wird mit zwei Backslashes `\\`, mit
mit dem Befehl `\newline` oder
mit einer Leerzeile vorgenommen.

Abstände

Ein größerer horizontaler Abstand wird zum Beispiel mit `\hspace{50mm}` und
ein vertikaler Abstand mit `\vspace{20mm}` vorgenommen.
Die horizontalen Abstände funktionieren nur innerhalb einer Zeile.

Wenn der Abstand am Anfang einer Zeile gebraucht wird, muss nach dem Befehl ein Asterix-Zeichen eingefügt werden: `\hspace*{50mm}`.

Innerhalb eines Textes kann es erforderlich sein, Abstände einzufügen. Dazu stehen folgende Befehle zur Verfügung:

Befehl mit Beispiel:	Auswirkung	Abstand
`m\!m`	mm	Ein negativer Abstand
`mm`	mm	normaler Abstand
`m\,m`	m m	kleinster Abstand
`m m`	m m	ein Leerzeichen
`m \enspace m`	m m	so breit wie ein Ziffer
`m \quad m`	m m	so breit, wie ein Buchstabe hoch ist
`m \qquad m`	m m	doppelt so breit wie ein `\quad`
`m \hspace{15mm} m`	m m	Ein 15mm breiter Abstand
`m \hfill m`	m m	Abstand bis zum Zeilenende

Maßeinheiten

Wenn für erforderliche Abstände im Dokument Werte eingegeben werden müssen – zum Beispiel für `\hspace{...}`, dann müssen die Werte mit den entsprechenden Maßeinheiten angegeben werden.

In LaTeX sind die wesentlichsten Maßeinheiten:

- `pt` das entspricht ca. 0.35 Millimeter
- `mm` Millimeter

- `cm` Zentimeter
- `in` Inch
- `ex` Die höhe eines kleinen „x“ vom momentan gewählten Zeichensatz.
- `em` Die breite eines großen „M“ vom momentan gewählten Zeichensatz.

Längenangaben

Abstände und Längen können auch auf andere Weise angegeben werden.
Beispielsweise mit:

- `\baselineskip` Vertikaler Zeilenabstand in einem Absatz.
- `\columnsep` Abstand zwischen Spalten.
- `\columnwidth` Die Breite einer Spalte.
- `\linewidth` Breite einer Zeile in er gewählten Umgebung.
- `\paperwidth` Breite einer Seite.
- `\paperheight` Höhe einer Seite.
- `\parindent` Einückung bei Absätzen.
- `\parskip` Vertikaler Abstand zwischen Absätzen.
- `\textheight` Höhe des Textbereiches der Seite.
- `\textwidth` Breite des Textbereiches der Seite.

3.3. LaTeX-Packages

Mit sogenannten *Packages* wird der Funktionsumfang an die Bedürfnisse des Autoren angepasst.
Der Befehl zum Import eines *Packages* lautet `\usepackage{ ..... }`.

Anschließend eine Auflistung der drei *Packages*, die in keinem deutschsprachiges LaTeX-Quelltext fehlen sollten.

`\usepackage[ngerman]{babel}`	notwendig für die deutsche Rechtschreibung
`\usepackage[T1]{fontenc}`	notwendig für deutsche Umlaute
`\usepackage[latin9]{inputenc}`	bei Mac OSX ideal.

Die Angaben in den eckigen Klammern sind für die *Optionen* des *Packages* zuständig.
Beim *Package* `inputenc` ist je nach Betriebssystem eine unterschiedliche Option anzugeben. Bei Windows-Computersystemen ist die Option `ansinew` gängig.
also:

```
\usepackage[ansinew]{inputenc}
```

3.4. Der LaTeX-Quellcode

Der Quellcode eines LaTeX-Dokuments ist im Wesentlichen in zwei Teile aufgeteilt:

Die Präambel ... dort wird in erster Linie das Dokumentenformat festgelegt, und es werden die notwendigen *Packages* eingefügt.

Das eigentliche Dokument beginnt mit `\begin {document}` und endet mit `\end {document}`.

Der Quellcode eines minimalen LaTeX-Dokuments sieht also beispielsweise so aus:

```
% Präambel
\documentclass[11pt, a4paper, oneside]{report}

\usepackage[ngerman]{babel}
\usepackage[T1]{fontenc}
\usepackage[latin9]{inputenc} % für OSX

% Das eigentliche Dokument
\begin{document}
Das ist mein vorläufig erstes \LaTeX{}-Dokument.
\end{document}
```

Alles was in einer Zeil nach einem `%` geschrieben wird gilt als Kommentar.

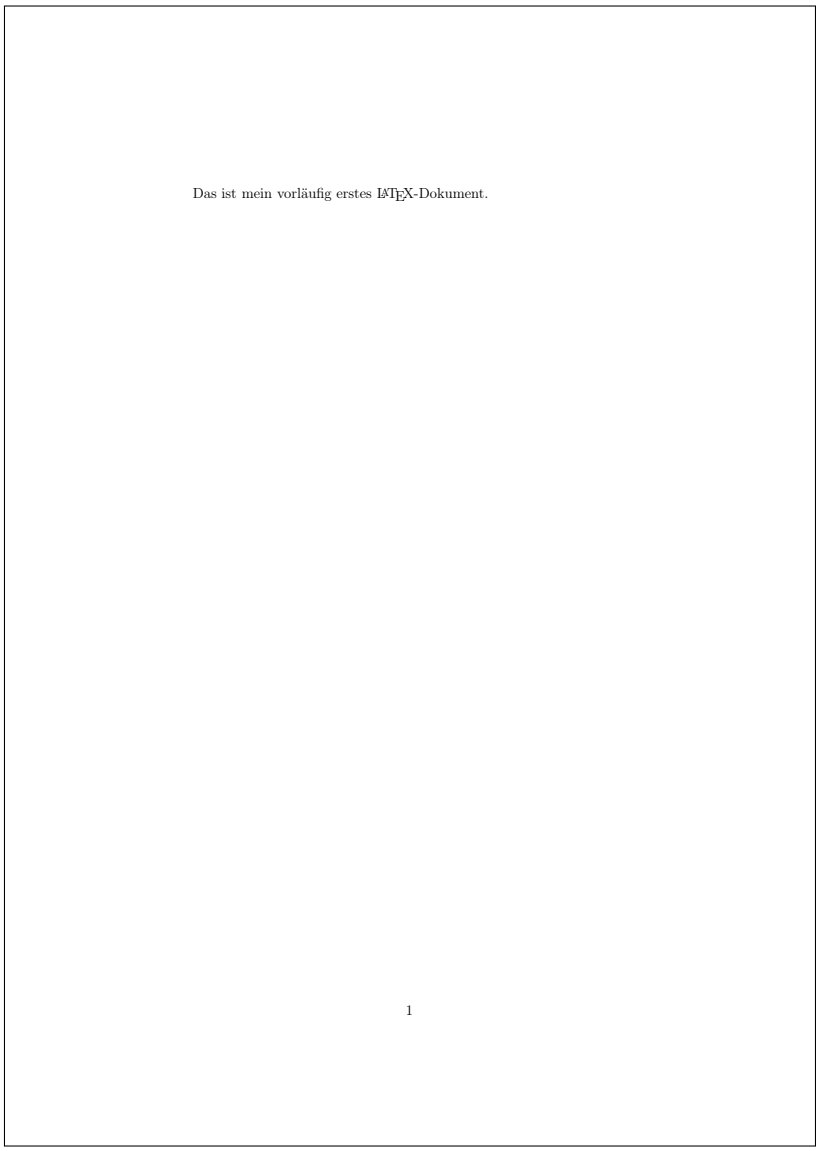
Mit der Möglichkeit im Quellcode zu Kommentieren, kann der Autor zusätzliche Gedanken und Beweggründe für seine Formulierungen einbinden.

Diese Kommentare erscheinen natürlich nicht im eigentlichen Dokument.

Dadurch, dass Leerräume, die über die Anzahl eines Leerzeichen oder einer Leerzeile hinausgehen von LaTeX ignoriert werden, kann der Quellcode beliebig gegliedert werden.

Am besten so, dass der Autor auch noch später einen guten Überblick über sein *Gesetztes* behält.

Das *Setzen* dieses Quellcodes liefert letztlich folgenden *Pdf*-File:



3.5. Dokumentenklasse

Die Auswahl der Dokumentenklasse muss in LaTeX im Quellcode an erster Stelle festgelegt werden.
Die Auswahl kann natürlich zu jedem Zeitpunkt wieder geändert werden.

Grundlegend existieren **zwei** Kategorien:

- Standardklassen
- KOMA-Script-Klassen[3]

Die Standardklassen sind eher auf amerikanische Typografie abgestimmt. Die KOMA-Script-Klassen unterstützen eher europäische Typografierichtlinien. Sie sind eine Weiterentwicklung der Standardklassen und bieten neue und komfortablere Schnittstellen zur Verwendung von LaTeX.

Durch die Auswahl des Klassen-Namens wird entschieden um welchen Klassentyp es sich handeln wird.

- Standardklassen

`article`	für kleinere Schriftwerke
`report`	für mittlere Schriftwerke
`book`	für große Schriftwerke
`letter`	für Briefe

- Koma-Script-Klassen

`scrartcl`	für kleinere Schriftwerke
`scrreprt`	für mittlere Schriftwerke
`scrbook`	für große Schriftwerke
`scrlttr2`	für Briefe

Das sind die gängigsten Auswahlmöglichkeiten zum Befehl `\documentclass[...]{...}`
In die geschwungene Klammer des `\documentclass`-Befehls wird der Klassenname geschrieben.
Innerhalb der eckigen Klammer stehen die eventuellen Optionen.
Mehrere Optionen werden durch einen Beistrich getrennt aufgelistet.

Optionen wären zum Beispiel:

- `12pt` für Schriftgröße (`10pt` ist die Voreinstellung).
- `a5paper` für das Seitenformat (`a4` ist die Voreinstellung).

[3] KOMA-Script existiert seit 1994 und wird von Markus Kohm entwickelt.

- `twocolumn` für das Seitenauteilung (`onecolumn` ist die Voreinstellung).
- `oneside` ist die Voreinstellung für `article` und `scrartcl` sowie `report` und `scrreprt`.
- `twoside` ist die Voreinstellung für `book` und `scrbook`.

Eine – zumindest vorläufige – Wahl einer geeigneten Dokumentklasse mit seinen Optionen steht jedenfalls zu Beginn eines Projektes.

3.6. Gliederung des Dokumententeils

Der Text zwischen `\begin{document}` und `\end{document}` wird in Bereiche eingeteilt. Hierbei ist in erster Linie die Einteilung in Kapitel und Überschriften gemeint.
Je nach Dokumentenklasse kann das LaTeX-Dokument in bis zu sieben Abschnitte unterteilt werden.

- `\part {Der erste Teil}`
- `\chapter {Das erste Kapitel}`
- `\section {erster Abschnitt}`
- `\subsection {erster Unterabschnitt }`
- `\subsubsection {erster Unterunterabschnitt}`
- `\paragraph {Absatz}`
- `\subparagraph {Unterabsatz}`

`\part` und `\chapter` sind nur in der `report`- und `book`- sowie in der `scrreprt`- und `scrbook`-Klasse möglich.
All diese Abschnitte werden in das Inhaltsverzeichnis aufgenommen.
Ein Inhaltsverzeichnis wird automatisch an der Stelle des Dokuments erstellt, wo der Befehl `\tableofcontents` geschrieben wird.

Falls ein Dokumentenabschnitt nicht in das Inhaltsverzeichnis aufgenommen werden sollte, muss ein Asterix (*) vor die öffnende geschwungene Klammer gesetzt werden. Also beispielsweise: `\chapter*{Das erste Kapitel}`. Damit bekommt dieser Dokumententeil auch keine Nummrierung.

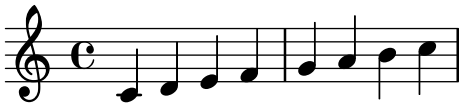
Um wiederum ein Kapitel ohne Nummerierung dennoch in das Inhaltsverzeichnis aufzunehmen ist beispielsweise der Befehl:
`\addcontentsline{toc}{section}{Titel dieses Abschnitts}` notwendig.

Damit das Inhaltsverzeichnis im *gesetzten* Dokument erscheint, muss der Befehl zum *Setzen* zweimal (!) erfolgen.

4. Die musikalische Notation

Für die musikalische Notation in LaTeX gibt es mehrere Möglichkeiten. Wobei ich mich für eine entschieden habe:

Die Integration von LilyPond in die LaTeX-Umgebung.



Diese Kombination bietet nach meinem Geschmack das beste Aussehen für kombinierte Text- und Noten-Dokumente.
Damit diese Notenzeile – und das könnte jede beliebige Notendarstellung sein – in das Dokument eingefügt wird, muss:

- in den LaTeX-Quelltext LilyPond-CODE eingefügt werden und
- im Editor **TeXShop** bei der Auswahl für *Schriftsatzsystem* „LilyPond-Book“ ausgewählt werden.

Es gibt vier Möglichkeiten LilyPond-CODE in LaTeX einzufügen:

- mit einem einfachen `\lilypond{...}`-Befehl, womit man direkt kurze LilyPond-Codeabschnitte schreiben kann
- mit der `\begin{lilypond}` ... `\end{lilypond}`-Umgebung, mit der man längere LilyPond-Codeabschnitte direkt schreiben kann. Bei dieser Möglichkeit, wird eine neue Zeile im Text begonnen.
- mit dem `\lilypondfile{...}`-Befehl um eine ganze LilyPond-Datei einzufügen.
- mit dem `\musicxmlfile{...}`-Befehl um eine MusicXML-Datei einzufügen, die dann LilyPond-Gerecht bearbeitet wird.

Das obige Notenbeispiel sieht im Quelltext genau so aus:

```
\begin{lilypond}
\relative c'
 { c d e f g a b c }
\end{lilypond}
```

Die Lilypond-Befehle können auch mit *Optionen* in eckigen Klammern fein abgestimmt werden.

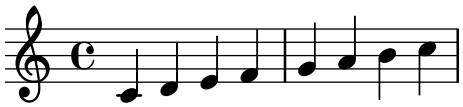
```
\lilypond[Optionen kommen hier her]{...}

\begin{lilypond}[Optionen kommen hier her]
LilyPond Quellcode
\end{lilypond}

\lilypondfile{...}[Optionen kommen hier her]

\musicxmlfile{...}[Optionen kommen hier her]
```

Zwischen den einzelnen Optionen muss immer ein Beistrich gesetzt werden.
Optionen wären zum Beispiel:

`indent=0mm`	Einzug der erste Notenzeile.
`noindent`	Kein Einzug der erste Notenzeile.
`quote`	reduziert die Notenzeilenlänge auf 2 * 0.4 inch es wird dabei eine neue Zeile begonnen das Notierte in einen Absatz gesetzt.
`exampleindent`	Setzt den Faktor für den Einzug bei `quote`
`staffsize=12`	gemessen in *points*
`ragged-right=##t`	`##t` oder `##f` für *true* oder *false*.
`noraggedright`	Notenzeilen werden über die Textbreite gestreckt.
`line-width=100mm`	Die Maßeinheiten in *pt*, *mm*, *cm*, *in* sind erlaubt.
`notime`	Versteckt die Taktanzeige in der Notation und setzt keine Taktstriche.
`fragment`	Einfacher LilyPond-CODE darf damit ohne die sonst obligaten geschwungenen Klammern eingegeben werden.
`relative=1`	implementiert automatisch die `fragment`-Option. Die Angabe nach dem Gleichheitszeichen setzt die Oktave. 1 = das eingestrichene c.

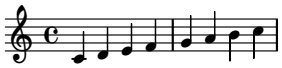
Beispiele mit LilyPond-Optionen

```
\begin{lilypond}
\relative c'
   { c d e f g a b c }
\end{lilypond}
```

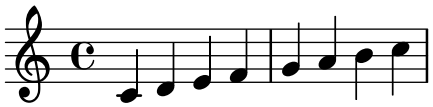


Mit den Voreinstellungen.

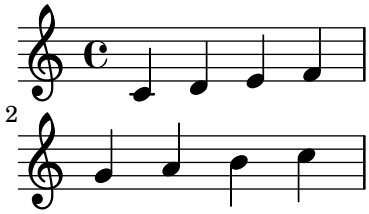
```
\begin{lilypond}[staffsize=12]
\relative c'
   { c d e f g a b c }
\end{lilypond}
```



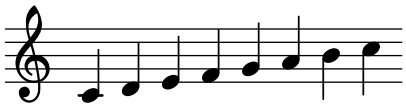
```
\begin{lilypond}[line-width=60mm]
\relative c'
   { c d e f g a b c }
\end{lilypond}
```



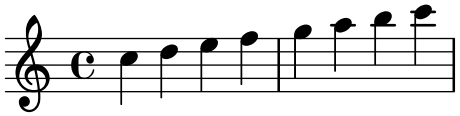
```
\begin{lilypond}[line-width=50mm]
\relative c'
   { c d e f g a b c }
\end{lilypond}
```



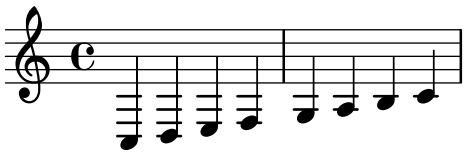

```
\begin{lilypond}[notime]
\relative c'
   { c d e f g a b c }
\end{lilypond}
```



```
\begin{lilypond}[relative=2]
   { c d e f g a b c }
\end{lilypond}
```

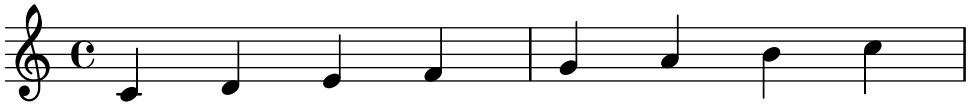


```
\begin{lilypond}[fragment, relative=0]
    c d e f g a b c
\end{lilypond}
```

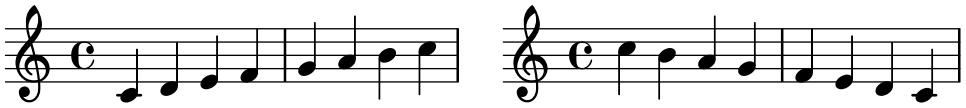


Man beachte, dass hier die ansonsten obligaten Klammern fehlen.

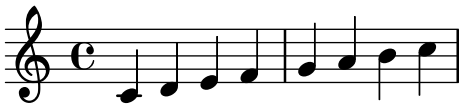
```
\begin{lilypond}[relative=1, noragged-right]
   { c d e f g a b c }
\end{lilypond}
```

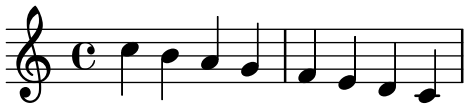


```
\lilypond[relative=1]{ c d e f g a b c }
\lilypond[relative=2]{ c b a g f e d c }
```

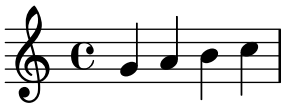


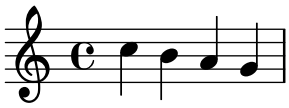
```
\lilypond[relative=1]{ c d e f g a b c }
\lilypond[relative=2, indent=30]{ c b a g f e d c }
```



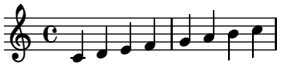


```
\lilypond[relative=2, indent=30]{ g a b c }
\lilypond[relative=2]{ c b a g }
```





```
Eine Notenzeile inmitten des Textflusses
\lilypond[staffsize=12, relative=1]{ c d e fg a b c }
vor und danach befindet sich Text \ldots
```

Eine Notenzeile inmitten des Textflusses ...  ...vor und danach befindet sich Text ...

Die von mir empfohlene Vorgangsweise, in der LilyPond-CODE in die LaTeX-Umgebung integriert wird, habe ich bisher nur auf Mac OSX mit dem Editor **TeXShop** getestet. Es funktioniert grundsätzlich auf allen Systemen ähnlich.

Im besten Fall wird auf einem Windows-LaTeX-Editor die Einstellung für „LilyPond-Book“ unterstützt – im schlimmsten Fall muss die Vorgangsweise, wie auf der LilyPond-Website beschrieben[1], eingehalten werden..

[1] http://lilypond.org/doc/v2.18/Documentation/usage/invoking-lilypond_002dbook(Dezember 2016)

5. Zusätzliche Pakete

Um LaTeX den Bedürfnissen des Autoren anzupassen und gegebenenfalls zu erweitern, ist das Einfügen von zusätzlichen *Packages* notwendig. Es existieren unzählige Erweiterungsmöglichkeiten, die in den Weiten des WorldWideWeb zu finden sind. Wenn Wünsche auftauchen, einfach das Internet befragen ;-)

Hier eine Auflistung von gängigen Erweiterungen, die gegebenenfalls in die *Präambel* geschrieben werden müssen. Es ist auch üblich, *Packages*, die in der *Präambel* stehen und nicht gebraucht werden, mit einem `%` auszukommentieren. Damit wird der Kommentar wie einen Schalter benützt.

- `\usepackage{lmodern}` für Feinheiten in der Typografie.
- `\usepackage{eurosym}` `\euro` erzeugt das Symbol €.
- `\usepackage{tabto}` `\tabto{50mm}` erzeugt einen Tabulator-Sprung.
- `\usepackage{microtype}` für feine typografische Optimierungen.
- `\usepackage{xcolor}` um überhaupt Farbe verwenden zu können.
- `\usepackage{graphicx}` zum Einbinden von Grafiken oder Bilder.
- `\usepackage{picins}` zum Umfließen von Bildern (muss „händisch“ geladen werden).
- `\usepackage{picinpar}` zum Umfließen von Bildern
- `\usepackage{wrapfig}` zum Umfließen von Bildern
- `\usepackage{sidecap}` um Bilder seitlich zu beschriften.
- `\usepackage{adjustbox}` um Bilder zu platzieren.
- `\usepackage{pdfpages}` zum Integrieren von *Pdf*-Seiten bzw. *Pdf*-Dokumente.
- `\usepackage{framed}` um Teile im Text einrahmen zu können.
- `\usepackage{verbatim}` um Textteile als CODE zu gestalten.
- `\usepackage{calc}` um Berechnungen ausführen zu können.
- `\usepackage{todonotes}` um in das Dokument Erinnerungen zu setzen.

- `\usepackage{amsmath, amssymb, amstext}`
  um mathematische Formeln schreiben zu können.
- `\usepackage{mathtools}` Mathematik-Module
- `\usepackage{cancel}` Mathematik-Module zum Kürzen von Brüchen
- `\usepackage{fancyhdr}` um Headers und Footers genau zu gestalten.
- `\usepackage{hyperref}` um eine *URL* setzen zu können.
- `\usepackage{soul}` um u. a. Textteile *gesperrt* zu schreiben oder zu unterstreichen.
- `\usepackage{geometry}` um die Seiteneinteilung selbst zu gestalten.
- `\usepackage{marginnote}` um Randnotizen einfügen zu können.
- `\usepackage{float}` für *Gleitumgebungen*.
- `\usepackage{booktabs}` zum Gestalten von Tabellen.
- `\usepackage{tabu}` zum besonderen Gestalten von Tabellen.
- `\usepackage{eso-pic}` zum Platzieren von Bildelementen.
- `\usepackage{rotating}` spezielle Maßnahmen um Elemente zu rotieren.
- `\usepackage{backgrounds}`
  Maßnahmen für den Hintergrund.
- `\usepackage{ifsym}` ein Zeichenfundus.
- `\usepackage{textpos}` absolute Positionierung von Text.
- `\usepackage{standalone}` für umfangreichere Dokumente.
- `\usepackage{musixtex}` wenn man LilyPond nicht verwenden will – eine andere Möglichkeit direkt in LaTeX zu notieren.

Große Grafikpakete mit jeder Menge zusätzlicher *Packages* sind noch:

- `\usepackage{tikz}` **T**ikz **i**st **k**ein **Z**eichenprogramm.
- `\usepackage{pdftricks}` für grafische Darstellungen.

6. Weitere Befehle

Zu den grundlegenden Befehlen, die bisher behandelt wurden, hier noch eine nützliche Auflistung von oftmals benötigten Anweisungen.

Einige Zeichen haben eine spezielle Bedeutung, wenn sie in den Quelltext geschrieben werden. Deshalb muss – um sie in den Textfluss einzubauen – für diese Zeichen eine spezielle Maßnahme getroffen werden:
Es handelt sich um:

- `\&` für ein &-Zeichen
- `\%` für ein %-Zeichen
- `\$` für ein $-Zeichen
- `\#` für ein #-Zeichen
- `\_` für ein _-Zeichen
- `\{` für ein {-Zeichen
- `\}` für ein }-Zeichen
- `\textasciitilde` für ein ~-Zeichen
- `\textasciicircum` für ein ^-Zeichen
- `\textbackslash` für ein \-Zeichen

Unter vielen möglichen, werden folgende Befehle des Öfteren benötigt:

- `--` für einen längeren Bindestrich –
- `---` für einen extralangen Bindestrich —
- `\glqq{}` und `\grqq{}` für normale Anführungszeichen „ und “
- `\glq{}` und `\grq{}` für halbe Anführungszeichen , und ‘
- `\flqq{}` und `\frqq{}` für französiche Anführungszeichen « und »
- `\flq{}` und `\frq{}` für halbe französiche Anführungszeichen ‹ und ›
- `\emph{hervorzuheben}` um Text *hervorzuheben*

- `\textsuperscript{hochzustellen}` um Text $^{\text{hochzustellen}}$
- `\textsubscript{tiefzustellen}` um Text $_{\text{tiefzustellen}}$
- `\footnote{...}` um Fußnoten zu erzeugen
- `\LaTeX{}` für den LaTeX-Schriftzug
- `\ldots` für drei Punkte ...
- `\vdots` für drei vertikale Punkte ⋮
- `\copyright` für das Copyright-Zeichen ©
- `\textcircled{x}` um Zeichen einzukreisen ⓧ
- `\newpage` um eine neue Seite zu beginnen
- `\centerline{...}` zentriert eine einzelne Textzeile
- `\fbox{...}` erzeugt einen rechteckigen Rahmen um den Inhalt
- `\rule{...}{...}` erzeugt eine Linie mit anzugebender Länge und Dicke.

Viele LaTeX-Befehle benötigen bestimmte *Packages* damit sie funktionieren.

- `\textcolor{...}{...}` um Text einzufärben – benötigt `\usepackage{xcolor}`
- `\includegraphics{...}` um Bilder einzufügen – benötigt `\usepackage{graphixs}`
- `\includepdf{...}` um *Pdf*-Files einzufügen – benötigt `\usepackage{pdfpages}`
- `\url{...}` um Links einzufügen – benötigt `\usepackage{hyperref}`
- `\tabto{...}` `\tabto{50mm}` für Tabulatoren (Wert und Einheit eingeben!) – z.B.: – benötigt `\usepackage{tabto}`

Manche Befehle werden mit einer *Umgebung*[1] eingeleitet:

- `\begin{itemize}` für Auflistungen mit Punkt
  `\item ...`
  ⋮
  `\end{itemize}`

[1] Umgebungen wirken auf einen begrenzten Textbereich. Sie werden mit den Befehlen `\begin{...}` und `\end{...}` eingerahmt.

- `\begin{enumerate}` für Aufzählungen mit Zahl
  `\item ...`
  ⋮
  `\end{enumerate}`
- `\begin{description}` für Auflistung mit Schlagwörtern
  `\item[Schlagwort] ...`
  ⋮
  `\end{description}`
- `\begin{minipage}{...}` für Seiten innerhalb von Seiten
  `{...}`
  `\end{minipage}` Die Breite der MiniPage muss angegeben werden!
- `\begin{quote}` formatiert den umschlossenen Text als Zitat
  `{...}`
  `\end{quote}`
- `\begin{center}` zentriert den umschlossenen Text
  `{...}`
  `\end{center}`
- `\begin{flushleft}` rückt den umschlossenen Text nach links
  `{...}`
  `\end{flushleft}`
- `\begin{flushright}` rückt den umschlossenen Text nach rechts
  `{...}`
  `\end{flushright}`
- `\begin{command}` Text innerhalb der Umgebung wird ignoriert
  `{...}`
  `\end{command}`

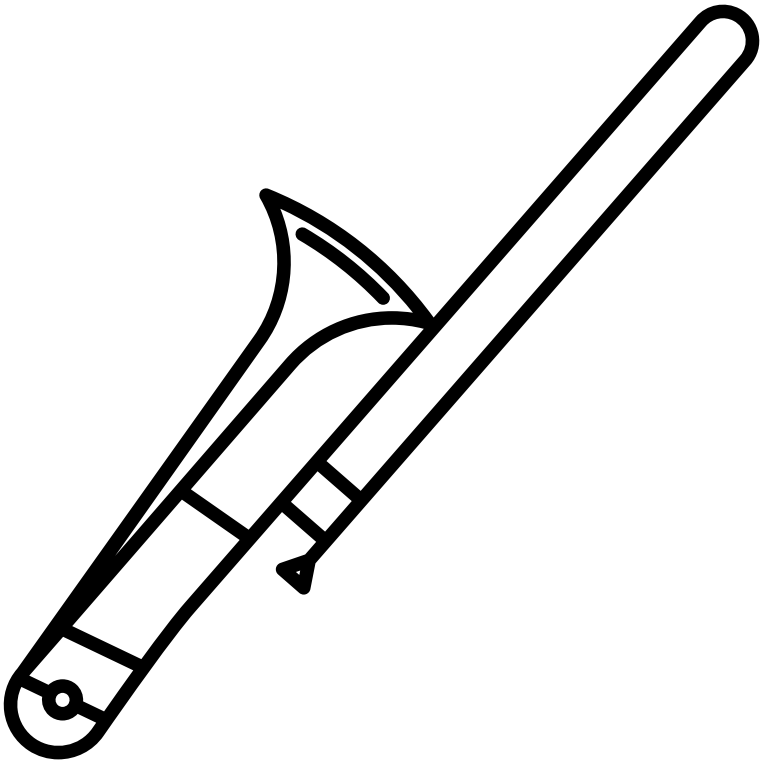
7. Bilder einfügen

7.1. \includegraphics

Wie erwähnt, muss – um Bilder in ein Dokument einzufügen zu können – das `\graphicx`-*Package* mit dem Befehl `\usepackage{graphicx}` in der Präambel eingebunden werden.

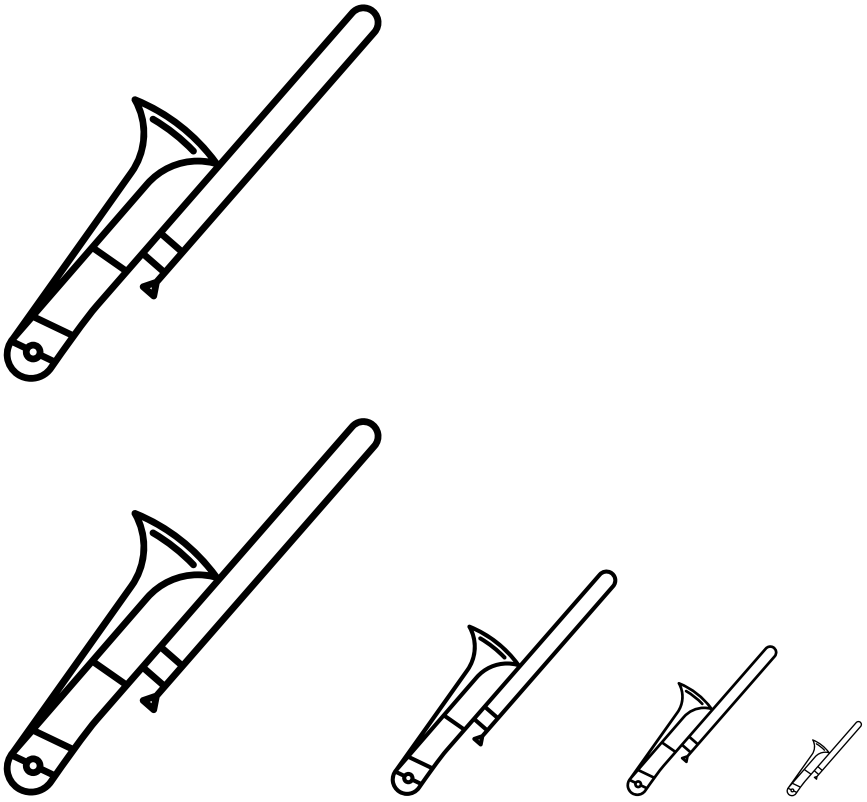
Bilder werden dann im Dokumententeil des LaTeX-Quelltextes mit `\includegraphics{...}` eingebunden.

Mit `\includegraphics{trombone.eps}` wird folgendes Bild[1] geladen und angezeigt:



[1] Das Bild wurde hier gefunden:
http://www.flaticon.com/free-icon/trombone_93001 (Dezember 2016).

Der Befehl `\includegraphics{...}` lässt einige Optionen zu.
Beispielsweise wird mit `\includegraphics[width=5cm]{trombone.eps}` das Bild in der Größe angepasst.



Nach einer neuen Zeile wurde die selbe Grafik hintereinander vier Mal mit `[with=5cm]`, `[with=3cm]`, `[with=2cm]` und `[with=1cm]` eingefügt.

Die Bildformate, welche sich in LaTeX einbinden lassen sind in der Regel .jpg, .png und .pdf oder .eps.

Sonderzeichen, Umlaute, das scharfe ß und Leerzeichen im Dateinamen sind nicht erlaubt!

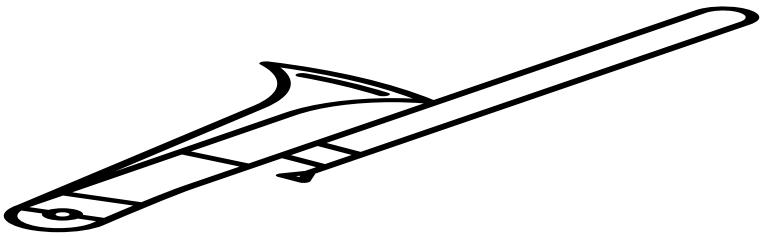
Wenn möglich sind den Bildformaten .eps oder .pdf der Vorzug zu geben, weil diese beliebig skalierbar sind.

Es wird in den folgenden Beispielen davon ausgegangen, dass die Bilddatei „trombone.eps“ im selben Ordner liegt, als die LaTeX.tex-Quelldatei. Das ist wichtig und es wird im nächsten Teil näher darauf eingegangen.

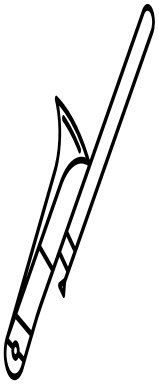
Einige gängige `\includegraphics`-Optionen sind:

- `width=...` die Grafik wird auf die angegebene Breite skaliert
- `height=...` die Grafik wird auf die angegebene Höhe skaliert
- `scale=...` die Grafik wird entsprechend einem Faktor skaliert
- `angle=...` die Grafik wird gegen den Uhrzeigersinn gedreht (Angabe in Winkelgrad)
- `trim=... ... ... ...` beschneidet die Grafik *links*, *unten*, *rechts* und *oben*

Die *Optionen* `width` und `height` skalieren das Bild proportional auf eine fixe Breite beziehungsweise eine fixe Höhe. Die Befehle können auch kombiniert werden. Beispielsweise führt
`\includegraphics[width=10cm, height=3cm]{trombone.eps}` zu:



und `\includegraphics[width=2cm, height=5cm]{trombone.eps}` zu:

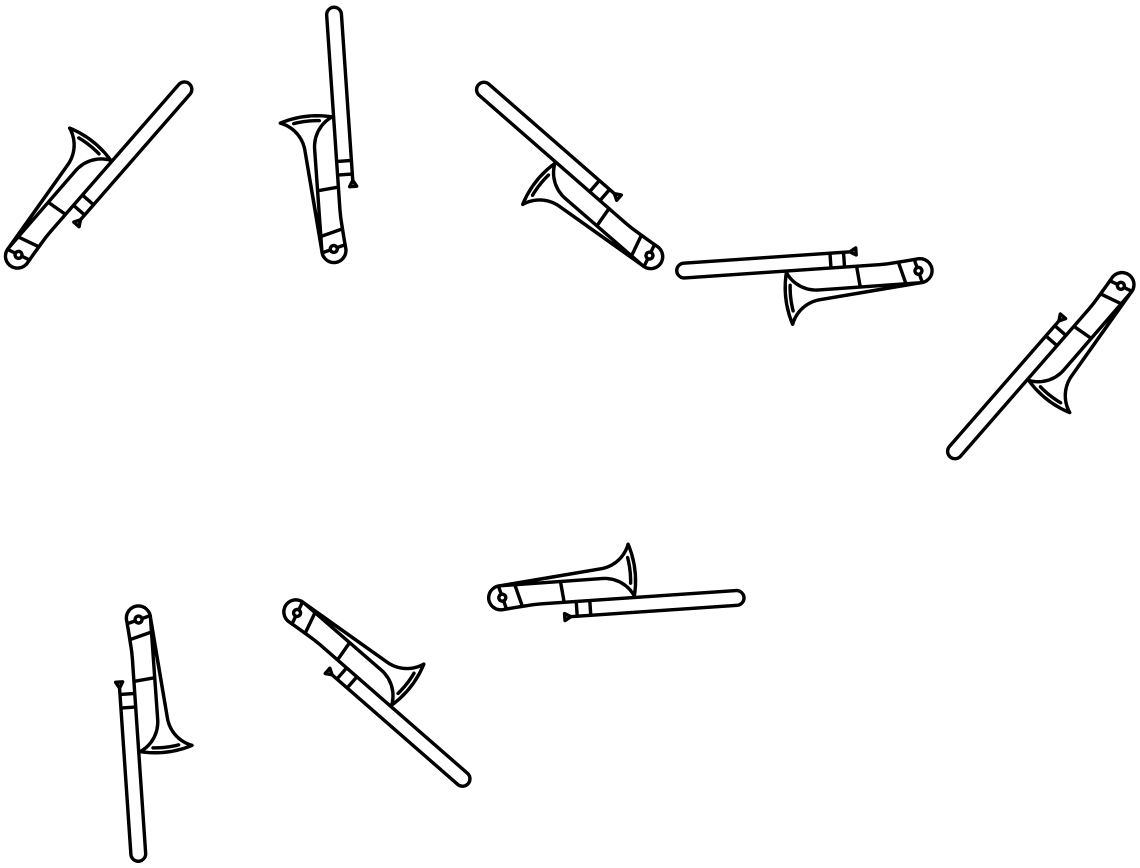


und

```
\includegraphics[scale=0.2, angle=0]{trombone.eps}
\includegraphics[scale=0.2 angle=45]{trombone.eps}
\includegraphics[scale=0.2, angle=90]{trombone.eps}
\includegraphics[scale=0.2, angle=135]{trombone.eps}
```

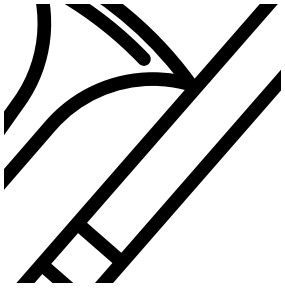
```
\includegraphics[scale=0.2, angle=180]{trombone.eps}
\includegraphics[scale=0.2, angle=225]{trombone.eps}
\includegraphics[scale=0.2, angle=270]{trombone.eps}
\includegraphics[scale=0.2, angle=316]{trombone.eps}
```

führt zu:

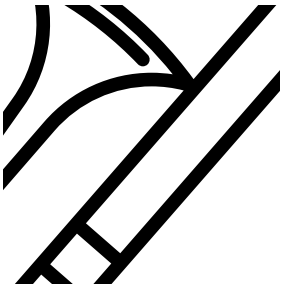


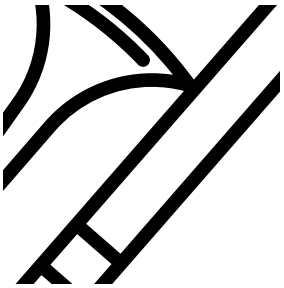
Zu beachten ist, dass die an `\includegraphics` übergebenen Optionen in der Reihenfolge ausgeführt werden, in der sie im Quellcode stehen. Es spielt also eine Rolle, ob ein Bild zuerst gedreht wird und danach auf eine Gesamtbreite skaliert, oder ob zuerst die Breite geändert und danach das Bild gedreht wird.

Ein Bild zu beschneiden, wäre eine Möglichkeit mit:
`\includegraphics[angle=0, scale=0.8, trim=4cm 4cm 4cm 4cm]{trombone.eps}`



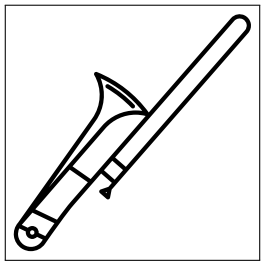
Text direkt nach der Grafik ohne `\newline`

Text direkt vor der Grafik ohne `\newline`: 

Text unmittelbar vor  und unmittelbar nach der Grafik.

Hier ist auch zu sehen, dass ein Bild direkt in den Textfluss eingebettet wird.

Ein einfacher Rahmen um eine Grafik lässt sich mit dem Befehl `\fbox{\includegraphics[scale=0.25]{trombone.eps}}` realisieren:



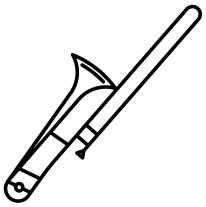
7.2. Bild-Positionierung

Wenn in die *Präambel* das *Package* `adjustbox` mit `\usepackage[export]{adjustbox}` eingebunden wird, sind zusätzliche Optionen zum Befehl `\includegraphics` möglich:

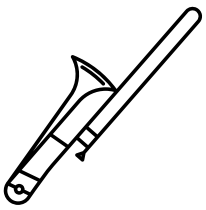
- `left`
- `right`
- `center`
- `outer`
- `inner`

Die letzten beiden `outer` und `inner` sind für beidseitige Dokumente gedacht.

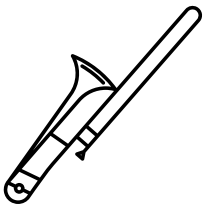
```
\includegraphics[width=0.2\textwidth, left]{trombone.eps}
```



```
\includegraphics[width=0.2\textwidth, center]{trombone.eps}
```



```
\includegraphics[width=0.2\textwidth, right]{trombone.eps}
```



7.3. figure-Umgebung

Eine übliche Art Bilder einzufügen, ist die beprochene `\includegraphics{...}` innerhalb einer `figure`-Umgebung mit `\begin{figure}` und `\end{figure}`.
`figure` ist eine sogenannte *Gleitumgebung*. Der Vorteil liegt darin, dass der Anteil von Text und Bild in einem Dokument für die Seitenaufteilung oft ungünstig ist, und LaTeX mit dieser *Gleitumgebung* die Platzierung vornimmt. Wobei folgende *Optionen* eingesetzt werden können:

- `h` *here*... wenn möglich setzt LaTeX das Bild an diese Stelle
- `t` *top*... das Bild kommt oben auf die Seite
- `b` *bottom*... das Bild kommt unten auf die Seite
- `p` *page*... das Bild kommt auf eine eigene Seite
- `!` *override*... erzwingt – wenn irgend möglich – die angegebene Position
- `H` *HERE*... das Bild kommt auf jeden Fall an diese Stelle
  damit die *Option* `H` funktioniert, muss zusätzlich das *Package* `float` mit `\usepackage{float}` in die *Präambel* eingefügt werden.

Innerhalb der `figure`-Umgebung kann durch den Befehl `\caption` eine Bildunterschrift vorgenommen werden. Des Weiteren ist mit dem Befehl `\reflectbox` möglich, das Bild zu spiegeln.

Folgender Code

```
\begin{figure}[H]
  \caption{Die Bildbezeichnung über dem Bild}
  \centering
  \includegraphics[scale=0.4]{trombone.eps}
\end{figure}

\begin{figure}[H]
  \centering
  \reflectbox
    {
      \includegraphics[scale=0.4]{trombone.eps}
    }
  \caption{Die Bildbezeichnung unter dem \textit{gespiegelten} Bild}
\end{figure}
```

führt zu:

Abbildung 7.1.: Die Bildbezeichnung über dem Bild

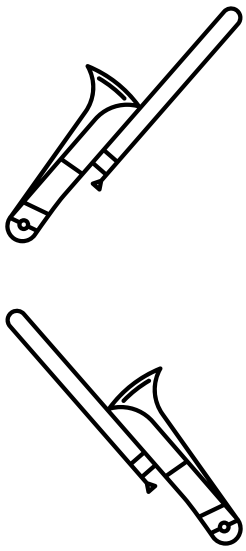


Abbildung 7.2.: Die Bildbezeichnung unter dem *gespiegelten* Bild

7.4. sidecap

Wenn das *Package* `sidecap` mit `\uspackage{sidecap}` in der *Präambel* eingefügt wird, ist eine seitliche Beschriftung der Bilder möglich:
Achtung: die eckige Klammer zwei Mal!

```
\begin{SCfigure}[][h!]
  \centering
  \caption{Die Posaune in all ihren Facetten}
  \includegraphics[scale=0.3]{trombone.eps}
\end{SCfigure}
```

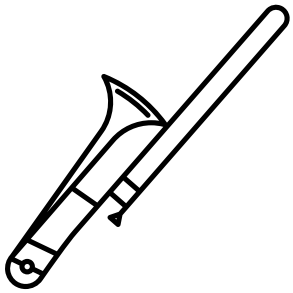


Abbildung 7.3: Die Posaune in all ihren Facetten

7.5. picinpar

Wenn das *Package* `picinpar` mit `\usepackage{picinpar}` in die *Präambel* eingesetzt wird, ist das Umfließen des Bildes mit Text möglich.

Der Befehl im Dokumententeil des LaTeX-Quelltexts lautet:

```
\begin{figwindow}[zeilenanzahl, position, {bild}, {capture}]
{
...beliebiger Text, der das Bild umschließt...
}
 \end{figwindow}
```

Die *Optionen* sind:

- `zeilenanzahl` die Anzahl der Zeilen über dem Bild
- `position` `r` für rechts oder `l` für links
- `bild` in geschwungene Klammern den Befehl mit `\includegraphics...`
- `capture` die Bildunterschrift

Vor dem `\end{figwindow}` kommt noch der Text, der das Bild umfließen soll in geschwungene Klammern.

Zum Beispiel:

```
\begin{figwindow}[1, l, {\includegraphics[width=3cm]{trombone.eps}}, {\\Die Posaune}]
{
Über die Entstehung der Posaune gibt es nur wenige Daten.
Zu den ältesten Existenz-Belegen des Instruments zählt ein englisches Dokument von 1495
und ein Gemälde von Matteo di Giovanni, der 1495 verstorben ist.
Die Posaune ist neben der Violine eines der ältesten voll chromatisch
spielbaren Orchesterinstrumente.
Weil eine Naturtrompete (Tromba) mit dem Grundton b eine unhandliche Länge
von etwa 2,80 Meter aufweist, wurden die Instrumente in S-Form gebogen,
gerollt oder in \glqq{}Brezelform\grqq{} hergestellt.
Der englische, französische und italienische Name des Instruments trombone
bedeutet wörtlich nichts anderes als \glqq{}große Trompete\grqq{}.
Der deutsche Name entwickelte sich aus der altfranzösischen
Bezeichnung \glqq{}buisine\grqq{}.
}
 \end{figwindow}
```

Über die Entstehung der Posaune gibt es nur wenige Daten. Zu den ältesten Existenz-Belegen des Instruments zählt ein englisches Dokument von 1495 und ein Gemälde von Matteo di Giovanni, der 1495 verstorben ist. Die Posaune ist neben der Violine eines der ältesten voll chromatisch spielbaren Orchesterinstrumente. Weil eine Naturtrompete (Tromba) mit dem Grundton b eine unhandliche Länge von etwa 2,80 Meter aufweist, wurden die Instrumente in S-Form gebogen, gerollt oder in „Brezelform“ hergestellt. Der englische, französische und italienische Name des Instruments trombone bedeutet wörtlich nichts anderes als „große Trompete“. Der deutsche Name entwickelte sich aus der altfranzösischen Bezeichnung „buisine“.

Abbildung 7.4.: Die Posaune

Der Text wurde hier[2] gefunden.
Bei den Beispielen wurde die Grafik etwas manipuliert,
und die `zeilenanzahl` von `1` bis `3` geändert (`0` wäre auch möglich).
Die `position` `r` und `l` für links und rechts hat bei mir keinen Unterschied bewirkt.

Über die Entstehung der Posaune gibt es nur wenige Daten. Zu den ältesten Existenz-Belegen des Instruments zählt ein englisches Dokument von 1495 und ein Gemälde von Matteo di Giovanni, der 1495 verstorben ist. Die Posaune ist neben der Violine eines der ältesten voll chromatisch spielbaren Orchesterinstrumente. Weil eine Naturtrompete (Tromba) mit dem Grundton b eine unhandliche Länge von etwa 2,80 Meter aufweist, wurden die Instrumente in S-Form gebogen, gerollt oder in „Brezelform“ hergestellt. Der englische, französische und italienische Name des Instruments trombone bedeutet wörtlich nichts anderes als „große Trompete“. Der deutsche Name entwickelte sich aus der altfranzösischen Bezeichnung „buisine“.

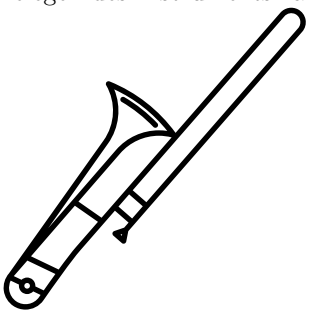


Abbildung 7.5.: Die Posaune

Über die Entstehung der Posaune gibt es nur wenige Daten. Zu den ältesten Existenz-Belegen des Instruments zählt ein englisches Dokument von 1495 und ein Gemälde von Matteo di Giovanni, der 1495 verstorben ist. Die Posaune ist neben der Violine eines der ältesten voll chromatisch spielbaren Orchesterinstrumente. Weil eine Naturtrompete (Tromba) mit dem Grundton b eine unhandliche Länge von etwa 2,80 Meter aufweist, wurden die Instrumente in S-Form gebogen, gerollt oder in „Brezelform“ hergestellt. Der englische, französische und italienische Name des Instruments trombone bedeutet wörtlich nichts anderes als „große Trompete“. Der deutsche Name entwickelte sich aus der altfranzösischen Bezeichnung „buisine“.

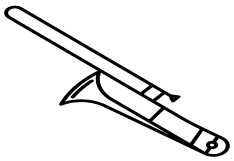


Abbildung 7.6.: Die Posaune

Es sind weitere *Packages* für das Umfließen von Bildern mit Text geschaffen worden.

[2] https://de.wikipedia.org/wiki/Posaune (Dezember 2016).

Empfehlen würde ich `\usepackage{wrapfig}` oder `\usepackage{picins}`.

Beide Varianten haben ihre Vorzüge.
`\picins` muss in TeXShop allerdings „händisch“ zugefügt werden.
Das heißt, der .sty-File muss hier[3]geladen werden – das Manual dazu ist hier[4]zu finden.

Der .sty-File kommt entweder in den selben Ordner, in dem auch der LaTeX-Quelltext liegt – diese Variante funktioniert nur, wenn beide Files (der .sty und der .tex) im gemeinsamen Ordner liegen.
Oder wenn man möchte, dass `\picins` genau so in die *Präambel* eingebunden werden kann, wie alle anderen *Packages*, muss auf Apple ab OSX 10.8

1. Im Ordner /Users/<user name>/Library ein Ordner „texmf“ angelegt werden.
2. Innerhalb dieses Ordners ein weiterer Ordner „tex“ und
3. innerhalb „tex“ wiederum ein Ordner „latex“ angelegt werden.
4. In diesen letzten Ordner werden alle .sty-Files „händisch“ eingefügt.

Nun kann mit `\usepackage{picins}` wie mit allen anderen *Packages* gearbeitet werden.

Dieser Work-Around könnte sich zum Einbinden zusätzlicher Erweiterung von LaTeX an anderer Stelle wieder einmal als nützlich erweisen.

[3] https://www.ctan.org/tex-archive/macros/latex209/contrib/picins?lang=de (Dezember 2016).
[4] http://michael-prokop.at/latextug/mpic.pdf (Dezember 2016).

8. Linkempfehlungen

The Latex Project
http://www.latex-project.org(Dezember 2016)

Comprehensive TEX Archive Network
http://ctan.org(Dezember 2016)

TeX Users Group
http://www.tug.org(Dezember 2016)

Share LaTeX
https://de.sharelatex.com/learn/(Dezember 2016)

CTAN Comprehensive TEX Archive Network
http://www.ctan.org/topic/german-doc(Dezember 2016)

LaTeX-Kompendium
https://de.wikibooks.org/wiki/LaTeX-Kompendium(Dezember 2016)

Typolexikon
http://www.typolexikon.de(Dezember 2016)

Homepage von Jürgen Weinelt
http://www.weinelt.de(Dezember 2016)

Partitur auf Knopfdruck?
http://www.michael-enzenhofer.at/masterarbeit(Dezember 2016)

Die aktuelle Ausgabe dieses Leitfadens
http://www.michael-enzenhofer.at/LaTeXfürMusiker(Dezember 2016)