

Die T_EXnische Komödie

dante
Deutschsprachige
Anwendervereinigung T_EX e.V.

32. Jahrgang Heft 2/2020 Mai 2020

2/2020

Impressum

»Die T_EXnische Komödie« ist die Mitgliedszeitschrift von DANTE e.V. Der Bezugspreis ist im Mitgliedsbeitrag enthalten. Namentlich gekennzeichnete Beiträge geben die Meinung der Autoren wieder. Reproduktion oder Nutzung der erschienenen Beiträge durch konventionelle, elektronische oder beliebige andere Verfahren ist nicht gestattet. Alle Rechte zur weiteren Verwendung außerhalb von DANTE e.V. liegen bei den jeweiligen Autoren.

Beiträge sollten in Standard-L^AT_EX-Quellcode unter Verwendung der Dokumentenklasse dtk erstellt und per E-Mail oder Datenträger (z. B. CD/DVD) an unten stehende Adresse der Redaktion geschickt werden. Sind spezielle Makros, L^AT_EX-Pakete oder Schriften notwendig, so müssen auch diese komplett mitgeliefert werden. Außerdem müssen sie auf Anfrage Interessierten zugänglich gemacht werden. Weitere Informationen für Autoren findet man auf der Projektseite <https://projekte.dante.de/DTK/AutorInfo> von DANTE e.V.

Diese Ausgabe wurde mit LuaHBTeX, Version 1.13.0 (TeX Live 2020) erstellt. Als Standardschriften kamen Libertinus Serif, Libertinus Sans Serif, Anonymous Pro und Libertinus Math zum Einsatz.

Erscheinungsweise: vierteljährlich

Erscheinungsort: Heidelberg

Auflage: 2400

Herausgeber: DANTE, Deutschsprachige Anwendervereinigung T_EX e.V.
Postfach 11 03 61
69072 Heidelberg

E-Mail: info@dante.de (DANTE e.V.)
dtkred@dante.de (Redaktion)

Druck: Schleunungdruck GmbH
Eltertstraße 27, 97828 Marktheidenfeld

Redaktion: Herbert Voß (verantwortlicher Redakteur)

Mitarbeit: Lukas C. Bossert Rudolf Herrmann Gert Ingold
Eberhard Lisse Rolf Niepraschk Christine Römer
Martin Sievers

Editorial

Liebe Leserinnen und liebe Leser,

in der letzten Ausgabe hatte ich Ihnen mitgeteilt, dass ich die Arbeit an der »Komödie« zu beenden gedenke. Leider hat niemand Interesse an der Übernahme der Verantwortung für »Die T_EXnische Komödie« gezeigt, sodass zum Jahresende ein Problem entstehen wird. Ich möchte daher nochmal einen Aufruf zur Mitarbeit und insbesondere zur Übernahme der Leitungsfunktion starten. Selbstverständlich kann jeder neue Verantwortliche eigene Ideen einbringen. Es ist auch nicht »in Stein gemeißelt«, dass es immer eine gedruckte Ausgabe sein muss.

In den Zeiten von »Corona« hat sich auch das E-Mail-Aufkommen verändert; die mit sinnvollem Inhalt nahmen ab und die mit Spam gefüllten zu. So gab es auch in der letzten Zeit keine eingereichten Manuskripte, sodass wir einen Beitrag bringen, der schon auf <https://www.typolexikon.de/schriftwahl/> erschienen ist. Wolfgang Beinert gibt Einblicke in die Wahl einer oder mehrerer Schriften für ein zu erstellendes Dokument. Es ist ja immer noch selten, dass man für die verschiedenen Schriftfamilien, einschließlich mathematischer Zeichen, auf eine einzelne Schrift zurückgreifen kann, wie beispielsweise »Lucida« oder »Libertinus«. Wie man in unserem Impressum nachlesen kann, wird selbst für einen solchen Fall manchmal eine andere Teilfamilie gewählt, weil man mit der vorgegebenen nicht zufrieden ist. Lukas Bossert zeigt in einem seiner beiden Beiträge, wie man mit dem Programm Biber im so genannten tool-Modus Bibliografieinträge einer Datei bearbeiten kann. Der andere Beitrag beschäftigt sich mit der uralten Frage: »Wie kommentiert man Quellcode«. Knuth hatte seinerzeit das »Literate Programming« in die Welt des Programmierens eingeführt.

Ich wünsche Ihnen wie immer viel Spaß beim Lesen und verbleibe

mit T_EXnischen Grüßen

Ihr Herbert Voß

Hinter der Bühne

Vereinsinternes

Grußwort

Liebe Mitglieder,

normalerweise hätte ich Ihnen an dieser Stelle von meinen noch frischen Eindrücken der Frühjahrstagung in Lübeck berichtet. Ich hätte mich über viele spannende Themen gefreut und darüber, dass wir einige neue Gesichter bei Teilnehmern wie auch Vortragenden begrüßen konnten. Die Teilnehmer von Firmen hätte ich wohl erwähnt, weil ich es wichtig finde, zu zeigen, was T_EX und Co. auch außerhalb der Wissenschaft zu leisten im Stande sind. »Last, but not least«, hätte ich sicherlich von der Organisation durch Till Tantau und seinem Team sowie von der Stadt Lübeck geschwärmt, nicht nur wegen meiner Leidenschaft für Marzipan.

Doch wie sagte schon der große Philosoph Lothar Matthäus, den manche noch aus seiner Zeit als Fußballer kennen: »Wäre, wäre, Fahrradkette!« Es ist leider alles ganz anders gekommen, als gedacht und geplant. »Normal« ist nur noch wenig. Der Alltag der allermeisten von uns hat sich in den vergangenen Wochen schlagartig und vermutlich auch nachhaltig verändert. Besonders die Dynamik der Entwicklungen, die Änderungen im 24-Stunden-Rhythmus haben viele von uns vor große Herausforderungen gestellt. Gerade als wir uns im Vorstand durchgerungen hatten, die Veranstaltung abzusagen, kam auch schon das behördliche Verbot. Zum Zeitpunkt der Tagung selbst waren dann sogar keine touristischen Reisen nach Schleswig-Holstein mehr möglich.

Ich wage nicht zu prognostizieren, in welcher Situation Sie diese DTK in den Händen halten. Sind Schulen und Kitas noch immer geschlossen? Müssen viele Geschäfte weiterhin geschlossen bleiben? Dürfen wir uns wieder in größeren Gruppen im Büro, in der Öffentlichkeit oder privat treffen? In diesen Tagen vor Ostern ist nur klar, dass der Schalter nicht von jetzt auf gleich wieder umgelegt werden wird. Wie auch immer die Situation sich entwickelt und egal wann wieder eine gewisse Normalität eintreten wird, ich wünsche Ihnen in jedem Fall, dass Sie alle, Ihre Familien und Freunde ohne bleibende Schäden – gesundheitliche oder wirtschaftliche – aus dieser extremen Situation herauskommen.

Natürlich tut es mir sehr leid für alle, die sich auf die Tagung in Lübeck gefreut haben. Einige wollten das erste Mal eine Tagung von DANTE e.V. besuchen. Manche

hatten dafür lange Reisen geplant, teilweise sogar aus Übersee. Wieder andere hatten sich vorbereitet, uns allen neue Einblicke in $\TeX$ nische Entwicklungen und Erweiterungen zu geben. Das alles ist hoffentlich zu einem großen Teil nur verschoben. Wie es genau weitergeht, kann ich zum jetzigen Zeitpunkt jedoch nur mutmaßen. Vermutlich werden wir die Frühjahrstagung nicht nachholen, sondern die Herbsttagung in diesem Jahr etwas ausdehnen und versuchen, sie zeitlich ein wenig nach vorne zu ziehen. Ein Grund ist auch die Mitgliederversammlung, bei der u. a. die Wahl eines neuen Vorstands auf der Tagesordnung stand. Der alte Vorstand bleibt gemäß unserer Satzung bis zur Wahl eines neuen Vorstands im Amt. Dieser Zustand sollte jedoch aus meiner Sicht nach Möglichkeit nicht über Gebühr strapaziert werden.

Ein weiterer Punkt der Mitgliederversammlung wäre die Bekanntgabe des Ehrenpreisträgers 2020 gewesen. Ich möchte Ihnen die Entscheidung nicht vorenthalten. Der Ehrenpreis geht in diesem Jahr an das $\LaTeX$ 3-Team. Neben Frank Mittelbach und Ulrike Fischer, die viele von Ihnen kennen werden, gehören viele weitere (internationale) Mitglieder dazu. Unter <https://www.latex-project.org/about/team/> finden Sie eine Übersicht dazu. Einige Teammitglieder hatten für Lübeck Vorträge vorbereitet, die nun leider warten müssen.

Apropos warten. $\LaTeX$ 3 drohte zwischenzeitlich das gleiche Schicksal wie manch anderer Open-Source-Entwicklung. Mit großen Ambitionen und Ansprüchen gestartet, ging dem Projekt ein wenig die Luft aus. Nicht wenige hatten eine »Fertigstellung« im Sinne der öffentlichen Verfügbarkeit neuer Funktionalitäten wohl schon aufgegeben. Nach den Entwicklungen der letzten Jahre – inhaltlich und personell – kann man aber mit Fug und Recht behaupten, dass sich $\LaTeX$ 3 bester Gesundheit erfreut.

Alle $\LaTeX$ -Nutzer profitieren von der Arbeit des Teams, auch wenn manche dies gar nicht wahrnehmen, weil sich vieles »unter der Haube« abspielt. Auch bei der neuen $\TeX$ Collection, die Sie zusammen mit dieser DTK erhalten, sind viele Entwicklungen des $\LaTeX$ 3-Teams eingeflossen. Ebenso sind alle neuen und aktualisierten Pakete des letzten Jahres sowie die aktualisierten Distributionen $\TeX$ Live und $\text{MiK}\TeX$ enthalten. Wieder einmal geht mein Lob an die zahlreichen Freiwilligen rund um den Globus, die ihren Teil zum Gelingen beigetragen haben. Auch in diesem Jahr wurden 5000 Exemplare durch DANTE e.V. in Auftrag gegeben und verteilt.

Ich wünsche Ihnen viel Spaß beim Erkunden der Inhalte, zunächst aber viel Vergnügen bei der weiteren Lektüre dieser DTK.

Herzlichst Ihr/Euer
Martin Sievers

Bretter, die die Welt bedeuten

Die Eingabe von Sonderzeichen

Herbert Voß

Die Eingabe von Sonderzeichen über die Tastatur wird von vielen Betriebssystemen unterschiedlich gehandhabt. In der Regel hat der Anwender auch keine direkte Informationen über die einzugebenden Tastenkombinationen, um bestimmte Zeichen zu erreichen, beispielsweise das deutsche öffnende und schließende Anführungszeichen. Diese Sonderzeichen unterscheiden sich von dem Zeichen, welches durch $\boxed{\text{⌘}} + \boxed{2}$ ausgegeben wird. Letzteres ist eigentlich nur der »Programmiererei« vorbehalten. Im Folgenden wird daher für die drei gängigsten Betriebssysteme eine Zusammenstellung über die Tastenkombinationen angegeben, um Sonderzeichen in einen Editor eingeben zu können.

Tastaturen

Wünschenswert wäre eine Tastaturbelegung, die möglichst viele der für wissenschaftliche Texte benötigten Zeichen enthält. Da man die Zahl der vorhandenen Tasten sinnvollerweise nicht beliebig erhöhen kann, bleibt nur das Arbeiten mit mehreren Ebenen.

Die sogenannte Neo-Tastaturbelegung [3], die über sechs verschiedene Ebenen verfügt (<https://neo-layout.org/>), konnte sich bislang nicht durchsetzen. [2] Sie hat folgende Eigenschaften:

- Optimierung bezüglich deutscher Buchstabenhäufigkeiten und Bi- und Triogrammen (Ebene 1 und 2) (vgl. <https://de.wikipedia.org/wiki/N-Gramm>)
- Sonderzeichen sind gut erreichbar (Ebene 3)
- Unicode-Zeichen wie „ « » ∫ √ α β ... sind direkt eingebbar (alle Ebenen)
- Navigation mittels Pfeiltasten $\leftarrow \uparrow \downarrow \rightarrow$ direkt auf der Haupttastatur (Ebene 4)

Ebene 1



Ebene 2



Ebene 3



Ebene 4



Ebene 5



Ebene 6



Die meisten deutschen Rechner verfügen über eine PC-Tastatur mit einer sogenannten T1-Tastaturbelegung. [4] Diese Belegung entspricht der DIN 2137-2:2018 und ist seit Jahren im Wesentlichen unverändert. [1]



Abb. 1: Tasten einer IBM/PC-Tastatur (deutsche/österreichische Belegung „T1“)

Obwohl diese Tastatur eine Dreifachbelegung für *alle* Tasten enthält, werden nur einige davon auf den Kappen angegeben. Dies erschwert es dem Anwender, sich die jeweiligen Tastenkombinationen zu merken, um beispielsweise ein Integralzeichen $\int$ direkt eingeben zu können. Alle Betriebssysteme verfügen zwar über entsprechende Programme, die die aktuelle Tastaturbelegung anzeigen, erfordern jedoch mehrere Aktionen, um Zeichen in einen Editor einzugeben.

Die folgende Zusammenstellung enthält für die drei hauptsächlich genutzten Betriebssysteme Windows, macOS und Linux (alle Varianten) eine Liste der Tastenkombinationen, um direkt häufig benutzte Sonderzeichen zu erreichen, ohne dass eine Umstellung des Tastaturlayouts notwendig ist. Aus Platzgründen werden Fußnoten nicht wiederholt, sondern beziehen sich teilweise auf bereits vorhandene.

Anführungszeichen

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
„ Deutsch öffnend	quotedblbase	Alt + 0 1 3 2	Alt + ^	AltGr + v
“ Deutsch schließend, Englisch öffnend	quotedblleft	Alt + 0 1 4 7	Alt + 2	AltGr + b
, Deutsch einfach öffnend	quotesinglbase	Alt + 0 1 3 0	Alt + s	⇧ + AltGr + v
‘ Deutsch einfach schließend, Englisch öffnend	quoteleft	Alt + 0 1 4 5	Alt + #	⇧ + AltGr + b
» Deutsch öffnend	guillemotright	Alt + 0 1 8 7	⇧ + Alt + q	AltGr + y
« Deutsch schließend	guillemotleft	Alt + 0 1 7 1	Alt + q	AltGr + x
> Deutsch einfach öffnend	guilsinglright	Alt + 0 1 5 5	⇧ + Alt + n	⇧ + AltGr + y
< Deutsch einfach schließend	guilsinglleft	Alt + 0 1 3 9	⇧ + Alt + b	⇧ + AltGr + x
” Englisch schließend	quotedblright	Alt + 0 1 4 8	⇧ + Alt + 2	AltGr + n
’ Apostroph, Englisch einfach schließend	quoteright	Alt + 0 1 4 6	⇧ + Alt + #	AltGr + #

Interpunktionszeichen

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
- Bindestrich/Trennstrich	hyphen	-	-	-
– Gedankenstrich (n-dash)	ndash	Alt + 0 1 5 0	Alt + -	AltGr + -
— Geviertstrich (m-dash)	mdash	Alt + 0 1 5 1	⇧ + Alt + -	⇧ + AltGr + -
... Auslassungspunkte (»Ellipse«)	ellipsis	Alt + 0 1 3 3	Alt + .	AltGr + .
’ Apostroph	quotesingle	Alt + 0 1 4 6	⇧ + Alt + #	AltGr + #
· Zentrierter Punkt	middot	Alt + 0 1 8 3	⇧ + Alt + 9	AltGr + .
• Zentrierter Punkt (fett)	bullet	Alt + 0 1 4 9	Alt + ü	⇧ + Strg + u + 2 0 2 2 ¹
¡ Spanisches öffnendes Ausrufezeichen	exclamdown	Alt + 0 1 6 1	Alt + ¡	⇧ + AltGr + ¡

¹ ⇧ und Strg beim Eingeben der Ziffern gedrückt lassen. Die Taste u braucht nur kurz betätigt zu werden.

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
! Ausrufezeichen	exclam	⇧ + 1	⇧ + 1	⇧ + 1
¿ Spanisches öffnendes Fragezeichen	questiondown	Alt + 0 1 9 1	Alt + ß	⇧ + AltGr + ß
? Fragezeichen	question	⇧ + ß	⇧ + ß	⇧ + ß
(Öffnende runde Klammer	parenleft	⇧ + 8	⇧ + 8	⇧ + 8
) Schließende runde Klammer	parenright	⇧ + 8	⇧ + 8	⇧ + 9
[Öffnende eckige Klammer	bracketleft	AltGr + 8	Alt + 5	AltGr + 8
] Schließende eckige Klammer	bracketright	AltGr + 9	Alt + 6	AltGr + 9
{ Öffnende geschweifte Klammer	braceleft	AltGr + 7	Alt + 8	AltGr + 7
} Schließende geschweifte Klammer	braceright	AltGr + 0	Alt + 9	AltGr + 0
/ Schrägstrich	slash	⇧ + 7	⇧ + 7	⇧ + 7
\ Umgekehrter Schrägstrich (»Backslash«)	backslash	AltGr + ß	⇧ + Alt + 7	AltGr + ß
Senkrechter Strich	bar	AltGr + <	Alt + 7	AltGr + <

Währungszeichen

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
€ Euro	Euro	AltGr + e	Alt + e	AltGr + e
\$ Dollar	dollar	⇧ + 4	⇧ + 4	⇧ + 4
¥ Yen	yen	Alt + 0 1 6 5	Alt + y	⇧ + AltGr + z
£ Pfund	sterling	Alt + 0 1 6 3	⇧ + Alt + 4	⇧ + AltGr + 3
¢ Cent	cent	Alt + 0 1 6 2	Alt + 4	AltGr + c
f Florin	florin	Alt + 0 1 6 2	Alt + f	⇧ + Strg + u + 1 9 2 ¹
¤ Allgem. Währungszeichen	currency	Alt + 0 1 6 4	—	⇧ + AltGr + 4

Sonstige Sonderzeichen

<i>Glyph</i>	<i>Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
@	At-Zeichen (»Klammeraffe«)	at	AltGr + q	Alt + l	AltGr + q
&	Et-Zeichen (»Kaufmanns-Und«)	ampersand	⇧ + 6	⇧ + 6	⇧ + 6
fi	fi-Ligatur	fi	—	⇧ + Alt + 5	⇧ + Strg + u + f b 0 1 ¹
fl	fl-Ligatur	fl	—	⇧ + Alt + l	⇧ + Strg + u + f b 0 2 ¹
§	Paragraphzeichen	section	⇧ + 3	⇧ + 3	⇧ + 3
%	Prozentzeichen	percent	⇧ + 5	⇧ + 5	⇧ + 5
‰	Promillezeichen	perthousand	Alt + 0 1 3 7	⇧ + Alt + e	⇧ + Strg + u + 2 0 3 0 ¹
¶ ™	Absatzzeichen (»Alinea«)	paragraph	Alt + 0 1 8 2	Alt + 3	AltGr + r
®	Trademark	trademark	Alt + 0 1 5 3	⇧ + Alt + d	⇧ + AltGr + 8
©	Copyright	copyright	Alt + 0 1 6 9	Alt + g	⇧ + AltGr + c
®	Registered	registered	Alt + 0 1 7 4	Alt + r	⇧ + AltGr + r
*	Sternchen (»Asterisk«)	asterisk	⇧ + +	⇧ + +	⇧ + +
†	Kreuz (Dagger)	dagger	Alt + 0 1 3 4	Alt + t	⇧ + Strg + u + 2 0 2 0 ¹
‡	Lothringer Kreuz (Double Dagger)	daggerdbl	Alt + 0 1 3 5	⇧ + Alt + y	⇧ + Strg + u + 2 0 2 1 ¹
°	Gradzeichen	degree	⇧ + ^	⇧ + ^	⇧ + ^
'	Hochkomma (Fuß, Minuten)	minute	⇧ + #	⇧ + #	⇧ + #
''	Anführungszeichen (Zoll, Sekunden)	second	⇧ + 2	⇧ + 2	⇧ + 2
♀	Ordinalzeichen (feminin)	ordfeminine	Alt + 0 1 7 0	Alt + h	⇧ + AltGr + f
♂	Ordinalzeichen (maskulin)	ordmasculine	Alt + 0 1 8 6	Alt + j	⇧ + AltGr + m

Diakritische Zeichen (»Akzentbuchstaben«)

<i>Glyph</i> <i>Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
À A mit Gravis	Agrave	Alt+0 1 9 2 ☐+`+☐+a	☐+`+☐+a	☐+`+☐+a
à a mit Gravis	agrave	Alt+0 2 2 4 ☐+`+a	☐+`+a	☐+`+a
Á A mit Akut	Aacute	Alt+0 1 9 3 ' ☐+a	' ☐+a	' ☐+a
á a mit Akut	aacute	Alt+0 2 2 5 ' a	' a	' a
Â A mit Zirkumflex	Acircumflex	Alt+0 1 9 4 ^ ☐+a	^ ☐+a	^ ☐+a
â a mit Zirkumflex	acircumflex	Alt+0 2 2 6 ^ a	^ a	^ a
Ã A mit Tilde	Atilde	Alt+0 1 9 5 Alt+n ☐+a	AltGr+`+☐+a <sup>2</sup>	AltGr+`+☐+a ²
ã a mit Tilde	atilde	Alt+0 2 2 7 Alt+n+a	AltGr+`+a <sup>2</sup>	AltGr+`+a ²
Ä A mit Trema	Adieresis	☐+ä	☐+ä	☐+ä
ä a mit Trema	adieresis	ä	ä	ä
Å A mit Ring	Aring	Alt+0 1 9 7 ☐+Alt+a	☐+AltGr+ü+☐+a	☐+AltGr+ü+☐+a
å a mit Ring	aring	Alt+0 2 2 9 Alt+a	☐+AltGr+ü+a	☐+AltGr+ü+a
Æ AE-Ligatur	AE	Alt+0 1 9 8 ☐+Alt+ä	☐+AltGr+a	☐+AltGr+a
æ ae-Ligatur	ae	Alt+0 2 3 0 Alt+ä	AltGr+a	AltGr+a
Ç C mit Zedille	Ccedilla	Alt+0 1 9 9 ☐+Alt+c	AltGr+`+☐+c	AltGr+`+☐+c
ç c mit Zedille	ccedilla	Alt+0 2 3 1 Alt+c	AltGr+`+c	AltGr+`+c
È E mit Gravis	Egrave	Alt+0 2 0 0 ☐+`+☐+e <sup>2</sup>	☐+`+☐+e ²	☐+`+☐+e ²
è e mit Gravis	egrave	Alt+0 2 3 2 ☐+`+e <sup>2</sup>	☐+`+e ²	☐+`+e ²
É E mit Akut	Eacute	Alt+0 2 0 1 ' ☐+e ²	' ☐+e ²	' ☐+e ²
é e mit Akut	eacute	Alt+0 2 3 3 ' e ²	' e ²	' e ²
Ê E mit Zirkumflex	Ecircumflex	Alt+0 2 0 2 ^ ☐+e ²	^ ☐+e ²	^ ☐+e ²
ê e mit Zirkumflex	ecircumflex	Alt+0 2 3 4 ^ e ²	^ e ²	^ e ²

² Die Taste des Akzents muss als Tottaste festgelegt sein!

Glyph Bedeutung		Glyphname	Windows	macOS	Linux
Ë	E mit Trema	Edieresis	Alt +0 2 0 3	Alt +u ☐ +e	AltGr +ü ☐ +e
è	e mit Trema	edieresis	Alt +0 2 3 5	AltGr +ü ☐ e	AltGr +ü ☐ e
ì	I mit Gravis	Igrave	Alt +0 2 0 4	☐ + Alt +g	☐ +` ☐ +i
ï	i mit Gravis	igrave	Alt +0 2 3 6	☐ +` i	☐ +` ☐ +i
í	i mit Akut	iacute	Alt +0 2 0 5	☐ + Alt +s	` ☐ +i ☐ +i
í	i mit Akut	iacute	Alt +0 2 3 7	` i	` i
î	I mit Zirkumflex	Icircumflex	Alt +0 2 0 6	^ ☐ +i ²	^ ☐ +i ²
î	i mit Zirkumflex	icircumflex	Alt +0 2 3 8	^ i ²	^ i ²
ï	I mit Trema	Idieresis	Alt +0 2 0 7	☐ + Alt +f	AltGr +ü ☐ +i
ï	i mit Trema	idieresis	Alt +0 2 3 9	Alt +u i	AltGr +ü i
ı	i ohne Punkt	dotlessi	—	☐ + Alt +j	☐ + AltGr +i
Ñ	N mit Tilde	Ntilde	Alt +0 2 0 9	Alt +n ☐ +n	AltGr +~ ☐ +n ²
ñ	n mit tilde	ntilde	Alt +0 2 4 1	Alt +n n	AltGr +~ n ²
Ò	O mit Gravis	Ograve	Alt +0 2 1 0	☐ +` ☐ +o <sup>2</sup>	☐ +` ☐ +o ²
ò	o mit Gravis	ograve	Alt +0 2 4 2	☐ +` o <sup>2</sup>	☐ +` o ²
Ó	O mit Akut	Oacute	Alt +0 2 1 1	☐ + Alt +h	` +☐ +o
ó	o mit Akut	oacute	Alt +0 2 4 3	` +o	` +o
Ô	O mit Zirkumflex	Ocircumflex	Alt +0 2 1 2	^ +☐ +o	^ ☐ +o
ô	o mit Zirkumflex	ocircumflex	Alt +0 2 4 4	^ o	^ o
Õ	O mit Tilde	Otilde	Alt +0 2 1 3	Alt +n ☐ +o	AltGr +~ ☐ +o ²
õ	o mit Tilde	otilde	Alt +0 2 4 5	Alt +n o	AltGr +~ o ²
Ö	O mit Trema	Odieresis	☐ +ö	☐ +ö	☐ +ö
ö	o mit Trema	odieresis	ö	ö	ö
Ø	O mit Schrägstrich	Oslash	Alt +0 2 1 6	☐ + Alt +o	☐ + AltGr +o
ø	o mit Schrägstrich	oslash	Alt +0 2 4 8	AltGr +o	AltGr +o

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
Œ OE-Ligatur	OE	Alt + 0 1 4 0	⇧ + Alt + ö	⇧ + Strg + u + 1 5 2 ¹
œ oe-Ligatur	oe	Alt + 0 1 5 6	Alt + ö	⇧ + Strg + u + 1 5 3 ¹
Š S mit Hatschek	Scaron	Alt + 0 1 3 8	–	⇧ + AltGr + ä ⇧ + s
š s mit Hatschek	scaron	Alt + 0 1 5 4	–	⇧ + AltGr + ä s
ß Scharfes s	ß	ß ³	ß	ß
Ù U mit Gravis	Ugrave	Alt + 0 2 1 7	⇧ + Alt + x	⇧ + ` ⇧ + u
ù u mit Gravis	ugrave	Alt + 0 2 4 9	⇧ + ` u	⇧ + ` u
Ú U mit Akut	Uacute	Alt + 0 2 1 8	⇧ + ´ ⇧ + u	⇧ + ´ ⇧ + u
ú u mit Akut	uacute	Alt + 0 2 5 0	´ u	´ u
Û U mit Zirkumflex	Ucircumflex	Alt + 0 2 1 9	⇧ + Alt + i	^ ⇧ + u
û u mit Zirkumflex	ucircumflex	Alt + 0 2 5 1	^ u	^ u
Ü U mit Trema	Udieresis	⇧ + ü ⁴	⇧ + ü	⇧ + ü
ü u mit Trema	udieresis	ü	ü	ü
Ý Y mit Akut	Yacute	Alt + 0 2 2 1	–	´ + ⇧ + y
ý y mit Akut	yacute	Alt + 0 2 5 3	–	´ + y
ÿ Y mit Trema	Ydieresis	Alt + 0 1 5 9	Alt + u ⇧ + y	AltGr + ü ⇧ + y
ÿ y mit Trema	ydieresis	Alt + 0 2 5 5	Alt + u y	AltGr + ü + y
Ž Z mit Hatschek	Zcaron	Alt + 0 1 4 2	–	⇧ + AltGr + ä ⇧ + z
ž z mit Hatschek	zcaron	Alt + 0 1 5 8	–	⇧ + AltGr + ä z
Ð Eth (Isländisch)	Eth	Alt + 0 2 0 8	–	⇧ + AltGr + d
ð eth (Isländisch)	eth	Alt + 0 2 4 0	–	AltGr + d
Þ Thorn (Isländisch)	Thorn	Alt + 0 2 2 2	–	⇧ + AltGr + p
þ thorn (Isländisch)	thorn	Alt + 0 2 5 4	–	AltGr + p

³ Oder Alt + 0 2 2 3

⁴ Oder Alt + 0 2 2 0

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
‘ Gravis (Grave)	grave	⇧ + `	⇧ + `	⇧ + `
’ Akut	acute	⇧ + Leerz. ⁵	⇧ + Leerz. ⁵	⇧ + Leerz. ⁵
^ Zirkumflex (≠ ^)	circumflex	Alt + 0 1 3 6	⇧ + Alt + k	^
~ Tilde	tilde	Alt + 0 1 5 2	⇧ + Alt + 8	AltGr + +
° Trema (Dieresis, Diärese, Umlaut)	dieresis	Alt + 0 1 6 8	Alt + u	AltGr + ü
◌̇ Ring-Akzent (≠ Gradzeichen °)	ring	—	⇧ + Alt + ‘	⇧ + AltGr + ü
◌̈ Hatschek (Caron, Häkchen)	caron	—	⇧ + Alt + z	⇧ + AltGr + ä
◌̋ Doppelakut (Hungarumlaut)	hungarumlaut	—	⇧ + Alt + t	AltGr + ö
◌̣ Punkt-Akzent	dotaccent	—	⇧ + Alt + ß	⇧ + AltGr + j
◌̄ Makron	macron	Alt + 0 1 7 5	⇧ + Alt + 0	⇧ + AltGr + +
◌̆ Breve	breve	—	⇧ + Alt + m	⇧ + AltGr + #
◌̈́ Zedille (Cedilla)	cedilla	—	⇧ + Alt + r	AltGr + ‘
◌̇ Ogonek	ogonek	—	⇧ + Alt + .	⇧ + AltGr + ‘

Mathematische Zeichen etc.

<i>Glyph Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
+ Plus	plus	+	+	+
× Malzeichen	multiply	Alt + 0 2 1 5	—	⇧ + AltGr + .
÷ Geteiltzeichen	divide	Alt + 0 2 4 7	⇧ + Alt + .	⇧ + AltGr + .
= Gleichheitszeichen	equal	⇧ + 0	⇧ + 0	⇧ + 0
≈ Fast gleich	approxequal	—	Alt + x	⇧ + Strg + u + 2 2 4 8 ¹
≠ Ungleich	notequal	—	Alt + 0	⇧ + Strg + u + 2 2 6 0 ¹
< Kleiner als	less	<	<	<

⁵ Ohne Leertaste, falls keine Tottaste für Akut aktiviert ist.

<i>Glyph</i>	<i>Bedeutung</i>	<i>Glyphname</i>	<i>Windows</i>	<i>macOS</i>	<i>Linux</i>
>	Größer als	greater	⇧ + >	⇧ + >	⇧ + >
≧	Kleiner gleich	lessequal	—	Alt + <	⇧ + Strg + u + 2 2 6 4 ¹
≡	Größer gleich	greaterequal	—	⇧ + Alt + <	⇧ + Strg + u + 2 2 6 5 ¹
+/-	Plusminus	plusminus	Alt + 0 1 7 7	Alt + +	⇧ + AltGr + 9
⌞	Unterbrochener senkrechter Strich	brokenbar	Alt + 0 1 6 6	—	⇧ + AltGr + <
ˆ	Ascii-Zirkumflex	circumflex	Alt + 9 4	⇧ + Alt + 6	^ + ^
~	Ascii-Tilde	tilde	AltGr + +	Alt + n	AltGr + + Leerz.
¬	Logisches Nicht	logicalnot	Alt + 0 1 7 2	⇧ + Alt + 1	AltGr + 6
μ	Mikrozeichen (mu)	mu	AltGr + m	Alt + m	AltGr + m
Ω	Ohmzeichen	Omega	—	Alt + z	⇧ + AltGr + q
Δ	Differenz/Größenzuwachs (Delta)	Delta	—	Alt + k	⇧ + Strg + u + 2 2 0 6 ¹
π	pi	pi	—	Alt + p	⇧ + Strg + u + 3 c 0 ¹
∂	Partielles Differential	partialdiff	—	Alt + d	⇧ + Strg + u + 2 2 0 2 ¹
∏	Produkt	product	—	⇧ + Alt + p	⇧ + Strg + u + 2 2 0 f ¹
Σ	Summe	summation	—	Alt + w	⇧ + Strg + u + 2 2 1 1 ¹
√	Wurzel	radical	—	Alt + v	⇧ + Strg + u + 2 2 1 a ¹
∞	Unendlich	infinity	—	Alt + .	⇧ + Strg + u + 2 2 1 e ¹
∫	Integral	integral	—	⇧ + Alt + b	⇧ + Strg + u + 2 2 2 b ¹
◇	Raute	lozenge	—	⇧ + Alt + v	⇧ + Strg + u + 2 2 c 4 ¹
/	Bruchstrich	fraction	—	Alt + i	⇧ + Strg + u + 2 0 4 4 ¹
¹	Hochgestellte 1	onesuperior	Alt + 0 1 8 5	—	AltGr + 1
²	Hochgestellte 2	twosuperior	AltGr + 2	—	AltGr + 2
³	Hochgestellte 3	threesuperior	AltGr + 3	—	AltGr + 3
¼	Ein Viertel	onequarter	Alt + 0 1 8 8	—	AltGr + 4
½	Einhalb	onehalf	Alt + 0 1 8 9	—	AltGr + 5
¾	Drei Viertel	threequarters	Alt + 0 1 9 0	—	⇧ + Strg + u b e

Beliebige Unicodezeichen

Die Eingabe beliebiger Unicodezeichen über ihre dezimale oder hexadezimale Platznummer (Slot) ist ebenfalls bei allen Betriebssystemen unterschiedlich. Dies soll am Beispiel des deutschen Lang-s »f« gezeigt werden, welches im Unicode die Platznummer 017F₁₆ hat, beziehungsweise 383₁₀.

Windows

Für Windows benötigt man die Zeichenummer im Dezimalsystem. Bei gedrückter **[Alt]**-Taste wird die Platznummer des Zeichens auf den numerischen Tasten eingegeben: **[Alt] + [3] [8] [3]**. Danach erscheint auf dem Bildschirm das Lang-s.

macOS

Für macOS muss das Tastaturlayout »Unicode Hex-Eingabe« gewählt werden, welches über die Systemeinstellungen **Systemeinstellungen** **Tastatur** **Eingabequellen** verfügbar gemacht werden kann. Durch die Eingabe von **⌘ + [0] [1] [7] [f]** wird dann ein f ausgegeben.

Eine dauerhafte Symbol in der Kopfleiste ist hierbei hilfreich, um ein späteres Umschalten zu erleichtern (Abb. 2). Zu beachten ist lediglich, dass bei der Wahl der Unicode-Eingabe die Tastatur zusätzlich auf das englische Layout (QWERTY) gesetzt wird und somit für deutschsprachige Texte nach der Eingabe eines Unicodezeichens wieder auf das deutsche Layout zurückgesetzt werden muss.

Abb. 2: Umschaltmöglichkeiten zwischen verschiedenen Tastaturlayouts bei macOS.



Linux

Die Eingabe von Unicodezeichen erfolgt über die Tastenkombination **⌘ + [Strg] + [u]**, welches in der Anzeige zu einem u im Editor oder Terminal führt. Nach Eingabe von **[1] [7] [f]** mit einem anschließenden **↵** wird dann u durch das Unicodezeichen ersetzt, in diesem Fall durch f.

Literatur und Software

- [1] Beuth Verlag GmbH: DIN 2137-2:2018-12, Tastaturen für die Daten- und Texteingabe - Teil 2: Zusätzliche Anforderungen; Text Deutsch und Englisch, 2018, DOI <https://dx.doi.org/10.31030/2890218>, <https://www.beuth.de/de/norm/din-2137-2/295731036> (besucht am 20. 2. 2020).
- [2] Arno Trautmann u. a.: »Neo & X_YL^AT_EX – Ergonomie und Zeichenvielfalt«, *DTK*, 22.2 (2010), 48–55.
- [3] WIKIPEDIA: Neo (Tastaturbelegung), [https://de.wikipedia.org/wiki/Neo_\(Tastaturbelegung\)](https://de.wikipedia.org/wiki/Neo_(Tastaturbelegung)) (besucht am 20. 2. 2020).
- [4] — Tastaturbelegung, <https://de.wikipedia.org/wiki/Tastaturbelegung> (besucht am 20. 2. 2020).

Schriftwahl¹

Wolfgang Beinert

Einführung

Die Wahl einer geeigneten Schrift bzw. eines Fonts ist in allen Teildisziplinen der Typografie von hoher Bedeutung. Sie beeinflusst nicht nur maßgeblich die Lesbarkeit und die Ästhetik eines Kommunikationsmediums, sie verursacht auch nachhaltige Konklusionen bei der Implementierung.

Ziel einer professionellen Schriftwahl ist es, Schriftschnitte zu finden, die sich für eine bestimmte Aufgabe und für einen bestimmten Schriftträger optimal eignen und die keine Opportunitätskosten² im Satz, in der Produktion, bei der Publikation, in der Rechtsfolge oder beim Rezipienten verursachen.

Terminologie

Grundsätzlich basiert die Terminologie und die Klassifikation von Schriften auf typografischen Schriftklassifikationsmodellen, beispielsweise der »Typeface Design

¹ Der Originalbeitrag erschien am 2. 3. 2020 auf <https://www.typolexikon.de/schriftwahl/>

² Opportunitätskosten sind entgangene Erlöse (entgangener Nutzen), die dadurch entstehen, dass vorhandene Möglichkeiten (Opportunitäten) nicht wahrgenommen werden. Anders formuliert: »Ignorantia legis non excusat« (»Unwissenheit schützt vor Strafe nicht«). Eine falsch gewählte Schrift, auch wenn sie »kostenlos« oder »günstig« angeboten wird, kann *summa summarum* im Nachhinein extrem hohe Kosten verursachen.

Grouping according to AFI« oder der »Matrix Beinert«.³ Im Großen und Ganzen orientieren sich alle Professionals, auch Type Designer und Font Foundries, an dieser Nomenklatur für Schriften römischen Ursprungs.

Makro- und Mikrotypografie

Die Vorauswahl einer Schrift nach Schriftgattung (z. B. Gebrochene Schriften), Hauptschriftgruppe (z. B. Grotesk), Schriftart (z. B. Klassizistische Antiqua), Schriftnebangruppe (z. B. Bodoni-Varianten) und Schriftstil (z. B. normal)⁴ sowie die Schrifttechnologie⁵ gehören in das Segment der Makrotypografie.

Die finale Schriftwahl eines bestimmten Schriftstils, beispielsweise einer Schriftreplik⁶, eines Schriftremakes⁷, eines Schriftklons⁸ oder einer speziellen Formvariante⁹ – also die endgültige Wahl der Schriftschnitte (Schriftstile) in Bezug auf Typometrie, Schriftbreite (z. B. normal), Schriftlage (z. B. kursiv) und Schriftstärke (z. B. halbfett) mit einer konkreten Schriftbezeichnung (z. B. Bauer Bodoni EF Medium Italic), der Font Foundry (z. B. Elsner+Flake), dem Dateinamen (z. B. BauerBodoniEFCEOP-MediumSC.otf) und der Zeichentabelle (z. B. CE, 296 Characters) gehören in das Segment der Mikrotypografie.

Auswahlkriterien

Unsere Kommunikationsmittel, unsere Kommunikationszielgruppen und unsere Lese- und Betrachtungsgewohnheiten sind so different und schnelllebig geworden, dass »die Typografie« – ehemals war damit ausschließlich der Satz für bzw. in Buch- und Zeitungsverlagen gemeint – spätestens seit dem digitalen Paradigmenwechsel nicht mehr existiert.

Heute wird die angewandte Typografie – das reproduzierbare Schriftbild als solches – differenziert in Teildisziplinen dargestellt, in denen teils völlig unterschiedliche visuelle und phonologische Prozesse beim Lesen, Leseabstände, Lesegehnheiten, gestalterische Regeln, typometrische und technologische Anforderungen für unterschiedliche Schriftträger gelten.

³ <https://www.typolexikon.de/schriftklassifikation-matrix-beinert/>

⁴ Schrift ist nicht gleich Schrift, auch wenn Schriften auf den ersten Blick ähnlich aussehen können oder vordergründig sogar den gleichen Namen tragen.

⁵ Unter »Schrifttechnologie« versteht man beispielsweise physische oder virtuelle Schrifttechnologien, PostScript oder TrueType Formate etc.

⁶ Eine Schriftreplik ist eine erneute Ausführung eines bereits vorhandenen Originals durch den Schriftgestalter*in selbst.

⁷ Ein Schriftremake ist eine spätere Nachbildungen einer bereits existierenden, aber nicht mehr am Markt verfügbaren Schrift, die sich streng am Original orientiert.

⁸ Ein Schriftklon ist eine 1:1-Kopie einer Schrift.

⁹ Eine Formvariante ist eine abweichende Interpretation einer bereits existierenden Schrift.

So ordnet sich beispielsweise in der »Lesetypografie« die Schrift dem Inhalt völlig unter, da hier die Lesbarkeit an vorderster Stelle steht. Folglich werden hier exzellente Textschriften benötigt, die der Leser auch beim Erfassen längerer Textpassagen, z. B. in einem Buch, einem Geschäftsbericht oder auf einer Website, als angenehm, richtig und als nicht störend empfindet.

In der »Gebrauchstypografie«, der »Kunsttypografie« oder in der »Plastischen Typografie« gelten wiederum völlig andere Rahmenbedingungen. Hier wäre vielleicht bei kurzen Texten ein Designer-Font sinnvoll, der die Aufmerksamkeit auf sich zieht – der allerdings bei längeren Texten nur schwer lesbar und somit im Segment der Lesetypografie inakzeptabel wäre. In der »Corporate Typography« können hingegen Anforderungen aus allen Teildisziplinen in Frage kommen, weshalb die Schriftwahl für »crossmediale« CDs (visuelle Erscheinungsbilder) sehr komplex sein kann.

Unabhängig von subjektiven Kriterien – also Vorlieben, Formgefühl, Stilempfinden, Gepflogenheiten, Zeitgeschmack oder kulturellen Ressourcen – basiert die Wahl von Schriften insbesondere auf praxisbezogenen Kriterien. Dazu zählen:

Funktion

- Typografische Teildisziplinen

In welcher(n) typografischen Disziplin(en) wird die Schrift(en) verwendet:

- Animationstypografie (Schrift in Bewegung, Schriftanimation)
- Corporate Typography (Schrift im Corporate Design, Leit- und Informationssystemen)
- Gebrauchstypografie (Akzidenz- und Werbetypografie)
- Kunsttypografie (Typodesign, Typografik)
- Lesetypografie (Basistypografie)
- Plastische Typografie (Lapidartypografie, dreidimensionale Schrift im Raum)

- Typometrie

Welche Typometrie (Letternarchitektur) benötigt eine Schrift? Muss es eine Textschrift (Werksatzschrift), Akzidenzschrift oder Zierschrift sein? Eine Antiqua mit Serifen, betonten Serifen oder ohne Serifen? Muss sie auf einem bestimmten Schriftträger optimal lesbar, dekorativ oder ein »Eyecatcher« sein? Welche Schriftbreiten, Schriftlagen und Schriftstärken werden für Grundschrift und Auszeichnungen benötigt? Welche Abstufung ist bei einer bestimmten Schrift im Detail vorteilhaft? Wird eine proportionale oder dicktengleiche Schrift benötigt?

- Optische Größen

In welchen muss die Schrift optimiert sein? Beispielsweise für

- Konsultationsgrößen für Kleingedrucktes

- Lesegrößen für Bücher und buchähnliche Publikationen
- Lesegrößen für Webseiten und digitale Publikationen
- Lese- und Schaugrößen für Animation und Bewegtbild
- Lese- und Schaugrößen für Drucksachen, digitalen Benutzeroberflächen oder plastischen Schriftträger
- Optische Größen für crossmedialen Einsatz
- Plakatgrößen bzw. Ferngrößen für Plakate, Leit- und Informationssysteme
- Figurenverzeichnis
Welchen Umfang benötigt das Figurenverzeichnis? Welche Schriftzeichen, Interpunktionszeichen, Sonderzeichen, Ligaturen, Arabischen Ziffern, Glyphen oder OpenType-Funktionen benötigt die Schrift? Wie beeinflusst der Umfang des Figurenverzeichnisses die Ladezeit einer Website oder mobilen Anwendung?
- Sprachen
Für welche Sprachen ist die Schrift optimiert? Werden nicht-römische Schriftschnitte aus der gleichen Schriftfamilie benötigt?
- Schriftfamilie
Welche Auszeichnungsschnitte werden benötigt? Wie umfangreich muss die Schriftfamilie bzw. Schriftsippe sein? Wird ein Expertensatz benötigt? Mit oder ohne spezielle Designgrößen?
- Semantisch-typografische Auszeichnungsmatrix
Wie viele Schriftstilvarianten werden für die Umsetzung der semantisch-typographischen Auszeichnungsmatrix benötigt?

Form

Charakter einer Schrift

- Welche Schrift passt zum Thema, Produkt oder Auftraggeber?
Welche Ästhetik, Anmutung, Modernität und Kontinuität braucht die Schrift? Entspricht sie dem Geschmack des Auftraggebers oder der Zielgruppe? Benötigt man eine Schrift, die gleich, ähnlich oder anders ist als die eines Mitbewerbers? Soll es eine traditionelle oder modische Schrift sein? Wie wirkt die Konditionierung der Schrift im Kontext?¹⁰
- Welche Formvariante ist optimal?

¹⁰ Schriften können in einem konkreten Bezug unterschiedlich wirken. Die gleiche Schrift kann in der einen Anwendung entweder unseriös, altbacken oder kontraproduktiv wirken, in einem anderen Bezug seriös, modern und konstruktiv. Beispielsweise wenn ein Wirtschaftsanwalt*in eine Comic-Schrift oder ein jüdisches Museum eine Fraktur-Schrift benutzt, könnte das eher destruktiv wirken. Wenn allerdings ein Kindergarten eine Comic-Schrift oder eine Hardrock-Band eine Fraktur-Schrift benutzt, könnte das möglicherweise genau die richtige Schriftwahl sein.

Welche Formvariante mit gleichen Klassifikationsmerkmalen ist die passende Schrift? Beispielsweise eine Bauer Bodoni, eine Monotype Bodoni oder eine bestimmte Bodoni von Elsner+Flake oder von Adobe? Welche Schriftvariante der gleichen Nebengruppe verfügt über die gewünschten typometrischen Eigenschaften?

- Implementierung

Schrift in der Produktion, Herstellung und Publikation

- Wie steht die Schrift auf unterschiedlichen Schriftträgern?

Wie gut steht die Schrift auf unterschiedlichen Papieren, Folien, Stoffen, Monitoren, etc.?

- Wie funktioniert die Schrift in analogen Produktionsverfahren?

Wie gut lässt sich die Schrift in gewerblichen Druckverfahren reproduzieren, beispielsweise im Digitaldruck, Flachdruck, Durchdruck, Hochdruck, Offsetdruck, Prägedruck, Siebdruck, Tampondruck oder Tiefdruck?

- Wie reagiert die Schrift auf technische Peripheriegeräte?

Wie reagiert die Schrift auf Peripheriegeräte, beispielsweise auf VGA-Monitore, Retina-Displays, Multi-Touch-Screendisplay oder niedrigauflösende Beamer? Ist sie mit Tintenstrahl- und Laserdruckern (PostScript)-kompatibel?

- Welche Schrifttechnologie wird benötigt?

Wird ein TrueType Font, PostScript Font oder Webfont benötigt? Mit welcher Software oder welchen Betriebssystemen muss der Font kompatibel sein?

- Kerning und Font Hinting

Welche Qualitätsanforderungen werden an das Kerning der Schrift gestellt? Wie wirkt das Hinting des Webfonts auf eine bestimmte Web Software (beispielsweise ein bestimmtes WordPress-Thema) oder unterschiedliche Web Browser (Firefox, Google, Chrome, ...)? Wie gut ist das Umbruchverhalten der Schrift? Wie reagiert die Schrift auf den optischen Randausgleich handelsüblicher Desktop Publishing Software?

- Ist der Nutzer qualifiziert, mit der gewählten Schrift zu arbeiten?

Können die gewählten Schriften in ihrer Anwendung komplex oder müssen sie für minderqualifiziertes Personal und Laien in der Anwendung simpel sein?¹¹

- Ist die Schrift bei Dritten verfügbar?

Ist die Schrift frei verfügbar oder muss sie von Dritten, z. B. einem Kunden oder einer Druckerei, erst gekauft, abonniert oder lizenziert werden?

¹¹ Dies betrifft beispielsweise extrafamiliäre Schriftmischungen. Hier ist es möglich, attraktive und individuelle Schriftmischungen zu kreieren, die allerdings eine durchgehende Anpassung der H-Linien oder x-Linien und ggf. der Normalschriftweiten erfordern. Dieser Prozess ist für minderqualifiziertes Personal nicht oder nur sehr bedingt machbar. Dagegen sind Schriftsippen, z. B. die Compatil von Linotype®, auch für Laien problemlos handhabbar.

Erwerb, Kosten, Lizenzen und Rechtsfolgen

Kosten und Nutzung einer Schrift

- Preview
 - Welcher Type Designer/Font Foundry bietet welche Schriftstilvariante zu welchem Preis, in welchem Figurenverzeichnisumfang, in welcher Qualität und zu welchen juristischen Konditionen an?
 - Druckschriften bzw. PostScript Fonts: Gibt es vor Erwerb eine gedruckte Schriftprobe?¹²
 - Screen Fonts bzw. TrueType Fonts: Gibt es vor Erwerb eine Schriftprobe per PDF?¹³
 - Webfonts: Gibt es vor Erwerb eine ausreichende Schriftprobe auf einer Website?¹⁴
 - Kann man die Schrift praxisbezogen testen (z. B. mit einer begrenzten Testversion), bevor man diese erwirbt?
- Kosten
 - Ist die Schrift kostenfrei oder kostenpflichtig?
 - Muss die Schrift gekauft, abonniert oder lizenziert werden?
 - Fallen Folgekosten pro Medium, Page Views oder PC-Arbeitsplatz an?¹⁵
 - Wie hoch sind die Folgekosten der gewählten Schrift in der Herstellung?¹⁶
- Nutzung
 - Ist die Schrift lizenzfrei oder lizenzpflichtig?¹⁷
 - Ist die Schrift nutzungsfrei oder nutzungsgebunden?

¹² Eine Druckschrift kann man nicht am Bildschirm beurteilen, sondern nur gedruckt. Schriftproben per PDF, die über einen Bürodruker ausgedruckt werden, sind in der Regel zu ungenau.

¹³ Die Darstellung eines TrueType Fonts ist in einem Webbrowser technisch bedingt anders, als beispielsweise in einem Anwendungsprogramm oder einem PDF.

¹⁴ Für die Beurteilung von Webfonts benötigt man kein PDF oder Druckmuster. Allerdings sollte auf der Website des Anbieters der Textumfang in unterschiedlichen Schriftgraden und Längen hinterlegt sein, damit man z. B. das Font Hinting in unterschiedlichen Web Browsern beurteilen kann. Aber Vorsicht: Wie Schriften in einem Web Browser dargestellt werden, hängt auch an der final verwendeten Software. So können beispielsweise unterschiedliche WordPress-Themen den gleichen Webfont völlig unterschiedlich darstellen.

¹⁵ Beispielsweise Webfonts bei Abrechnung nach Pageviews, bei Abonnementmodellen oder Anwendung auf anderen Medien. Oder was kostet eine Schriftlizenz pro PC-Arbeitsplatz?

¹⁶ Die Folgekosten einer Schrift können weit über den Anschaffungskosten liegen. Beispielsweise kann sich die Schriftbreite und die Normalschriftweite spürbar auf den Umfang einer Publikation bzw. Mediums auswirken und somit Druck-, Lager- und Versandkosten verursachen oder einsparen.

¹⁷ Siehe Kommentar zu »Urheberrecht für Schriften und Mythos Schriftsoftware«: <https://www.typolexikon.de/schriftlizenzen/>

- Fallen besondere Lizenzbestimmungen an und welche juristischen Auswirkungen haben diese?¹⁸
- Hosting und SEQ
 - Soll es eine Systemschrift oder ein Webfont sein?
 - Wie beeinflusst das Figurenverzeichnis des Webfonts die Ladezeit (PageSpeed) der Website und was sind die Konsequenzen für die SEO (Suchmaschinenoptimierung) bzw. das Ranking bei Suchmaschinen, beispielsweise bei Google?
 - Soll die Schrift auf dem eigenen Server oder bei einem Fremdhoster hinterlegt oder gestreamt werden (z. B. bei Google oder Adobe)?
 - Wie schnell und sicher ist das Hosting des Webfonts?
 - Wie reagieren Firewalls auf den Webfont?
- Datenschutz
 - Wer hostet oder streamt den Font wie und in welchem Land?¹⁹
 - Wird der Font von der Font Foundry oder dem Hoster getrackt, beispielsweise Google Fonts von Google?
 - Zeichnen abonnierte oder gestreamte Fonts die Verbindungsdaten der Website oder des PC's auf?
 - Unterlaufen abonnierte oder gestreamte Fonts den Datenschutz, z. B. die Datenschutz-Grundverordnung (DSGVO), und welche Rechtsfolgen hat das?

¹⁸ Manche Font Foundries untersagen beispielsweise das Erzeugen eines PDFs. Andere tracken Webfonts und haben somit Einsicht in den Traffic einer Website, wiederum andere verbieten die Nutzung eines OT-Fonts im Internet etc.

¹⁹ Eigener oder fremder Server? Welche Gefahren drohen – insbesondere bei Abomodellen – bei Insolvenz oder Einstellung des Webfontdienstes (z. B. WebINK, der ab Juni 2015 seinen Dienst einstellt und dadurch erhebliche Kosten bei Agenturen und Firmen verursacht. Quelle: Pressemitteilung Dr. Web: Webfonts adé: Typekit-Konkurrent WebINK gibt auf, 17.2.2015). Wie schnell sind die Server? Wie gut ist die Anbindung an das Web?

Mit »biber --tool« Bibliografieeinträge bearbeiten

Lukas C. Bossert

Es gibt verschiedene Programme, mit denen .bib-Dateien bearbeitet werden können (bspw. JabRef, BibDesk etc.). Möchte man hingegen Bibliografieeinträge in einem automatisierten Prozess verändern, dann bedarf es eines anderen Werkzeugs.

In diesem Beitrag wird gezeigt, wie mit biber im tool-Modus beliebig große .bib-Dateien bearbeitet werden können.

Ausgangspunkt meiner Beschäftigung mit biber und dem Werkzeugmodus war ein konkreter Anlass: Ich brauchte fortlaufend aktualisierte Listen mit Publikationen, die innerhalb eines Sonderforschungsbereichs (SFB) entstanden sind. Da eine händische Pflege dieser Liste mir deutlich zu aufwendig schien und die Daten über die Hochschulbibliografie der Bibliothek ohnehin gepflegt werden,¹ wollte ich einen effizienteren Weg gehen.²

Fast alle Bibliothekskataloge bieten die Möglichkeit an, Bibliografieeinträge im Bib_{TEX}-Format zu exportieren. Dabei sind die Publikationsmetadaten sehr unterschiedlich, was beispielsweise die Ausführlichkeit der Beschreibung oder deren Ausgabe angeht.³ Eine Nachbearbeitung der Metadaten ist daher von Fall zu Fall notwendig. Diese wollte ich aber so automatisiert wie möglich gestalten. Genau dafür eignet sich biber hervorragend.

Ausgangslage

Dank einer API-gestützten Abfrage konnte ich mir über die Kommandozeile eine .bib-Datei mit allen Publikationen des SFBs herunterladen und als sfb.bib speichern.

```
curl "http://publications.rwth-aachen.de/PubExporter.py?p=pid:G:(GEPRIS)
↳403224013&of=hx" > sfb.bib
```

Ein willkürlich herausgegriffener Datensatz sieht so aus:

¹ <https://publications.rwth-aachen.de/record/783071>

² Die nachfolgenden Beispiele sind zwar konkrete Bib_{TEX}-Einträge, die Vorgehensweise mit biber ist jedoch exemplarisch zu verstehen.

³ Oft werden noch »veraltete« Feldnamen wie address oder journal, exportiert.


```

1 % IMPORTANT: The following is UTF-8 encoded. This means that in the presence
2 % of non-ASCII characters, it will not work with BibTeX 0.99 or older.
3 % Instead, you should use an up-to-date BibTeX implementation like "bibtex8" or
4 % "biber".
5
6 @ARTICLE{Babbar:782496,
7   author      = {Babbar, Anshu and Hitch, Thomas C. A. and others},
8   title       = {{T}he {C}ompromised {M}ucosal {I}mmune {S}ystem of  $\beta$ 
9                 {I}ntegrin-{D}eficient {M}ice {H}as {O}nly {M}inor {E}
10                  cffects on the {F}ecal {M}icrobiota in {H}omeostasis},
11   journal     = {Frontiers in microbiology},
12   volume      = {10},
13   issn        = {1664-302X},
14   address     = {Lausanne},
15   publisher   = {Frontiers Media},
16   reportid    = {RWTH-2020-01913},
17   pages       = {2284},
18   year        = {2019},
19   cin         = {526000-2 / 525500-2 / 537500-2},
20   ddc         = {570},
21   cid         = {$I:(DE-82)526000-2_20140620$ / $I:(DE-82)525500-2_20140620$ / $
22                  cI:(DE-82)537500-2_20140620$},
23   pnm         = {SFB 1382 - SFB 1382: Die Darm-Leber-Achse - Funktionelle
24                  cZusammenhänge und therapeutische Strategien (403224013)},
25   pid         = {G:(GEPRIS)403224013},
26   typ         = {PUB:(DE-HGF)16},
27   UT          = {WOS:000491333100001},
28   doi         = {10.3389/fmicb.2019.02284},
29   url         = {https://publications.rwth-aachen.de/record/782496},
30 }

```

Neben den bereits genannten »veralteten« Feldnamen, wie `address` oder `journal` gibt es noch weitere Feldnamen, die nicht zum BibLaTeX-Format gehören: `reportid`, `cin`, `ddc`, `cid`, `pnm`, `pid`, `UT`. Diese Felder werden von BibLaTeX standardmäßig nicht verarbeitet und der gewählte Bibliografiestil müsste diese explizit in ein eigenes Datenmodell aufnehmen.⁴ Zudem werden alle Großbuchstaben im Titel mit `{...}` eingefasst, um eine Kleinsetzung zu vermeiden. Diese Großschreibung (Capitalizing) sollte aber nicht aus dem Datensatz heraus erzwungen, sondern vom Bibliografiestil umgesetzt werden.

⁴Da es sich bei diesen Feldern um interne Informationen handelt, die im Normalfall ohnehin von BibLaTeX ignoriert werden, ist es nur mein eigener ästhetischer Anspruch an die BibTeX-Darstellung, diese Felder zu löschen.

Die Bearbeitung des Datensatzes mit biber kümmert sich ohne besondere Spezifikation um mehrere Punkte. Mit ein paar Optionen lassen sich weitere Anpassungen vornehmen:

- `--tool` startet biber im Werkzeugmodus.
- Mit `--outfile <FILE.bib>` wird eine explizite Ausgabedatei angegeben. In unserem Fall ist es derselbe Name der Eingabedatei; ansonsten wird biber eine `.bib`-Datei mit dem Zusatz `_bibertool` erstellen.
- `--output_align` sorgt für eine ausgerichtete Darstellung der Feldnamen und -inhalte.
- `--output_indent` regelt, um wie viele Leerzeichen eingerückt werden soll.
- Über `--output_fieldcase=lower` wird angegeben, dass alle Feldnamen und der Datensatztyp kleingeschrieben werden.
- Über `--output-field-order=<FELDNAMEN>` kann man die interne Reihenfolge der Felder festlegen.

Am Ende des Befehls steht der Dateiname der zu bearbeitenden `.bib`-Datei.

```
biber --tool --outfile sfb.bib --output_align --output_indent=1 --
  ↪output_fieldcase=lower --output-field-order=author,title,date sfb.bib
```

Das Ergebnis sieht für den Datensatz wie folgt aus:

```
1 @article{Babbar:782496,
2   author    = {Babbar, Anshu and Hitch, Thomas C. A. and others},
3   title     = {{T}he {C}ompromised {M}ucosal {I}mmune {S}ystem of  $\beta$ 7 {I}
  ↪ntegrin-{D}eficient {M}ice {H}as {O}nly {M}inor {E}ffects on the {F}ecal {M}
  ↪icrobiota in {H}omeostasis},
4   date      = {2019},
5   doi       = {10.3389/fmicb.2019.02284},
6   issn      = {1664-302X},
7   journaltitle = {Frontiers in microbiology},
8   location  = {Lausanne},
9   pages     = {2284},
10  publisher  = {Frontiers Media},
11  url        = {https://publications.rwth-aachen.de/record/782496},
12  volume     = {10},
13 }
```

Alle Bib^LA_TE_X nicht bekannten Feldnamen werden mit ihren Inhalten gelöscht. Die »veralteten« Feldnamen werden umbenannt (zu `location` und `journaltitle`); und der Kommentar über dem Eintrag wird ebenfalls entfernt. Die ersten drei Felder werden wie gewünscht sortiert: `author`, `title`, `date`, der Rest wird alphabetisch angeordnet.

Eigene Modifikationen einbauen

Der Datensatz ist an sich Bib^LA_TE_X-kompatibel, aber in meinem konkreten Fall mussten zusätzlich mehrere Punkte an die eigenen Bedürfnisse und Vorgaben angepasst werden:

- Die Einklammerung der Großbuchstaben im Titel störte noch beim Lesen.
- Das Feld issn war für alle Datensatztypen nicht notwendig.
- Die Ortsangabe (location) sollte nur für @article gelöscht werden.
- Für interne Zwecke musste das Feld reportid jedoch erhalten bleiben.
- Wenn im Feld prn die DFG-Fördernummer geschrieben steht, musste
 - in das Feld keywords *CRC1382* nachgetragen und
 - der Name des SFBs in das Feld note geschrieben werden.

Diese weiteren Bearbeitungswünsche sind an sich problemlos umzusetzen, müssen jedoch in einer extern gespeicherten Konfigurationsdatei definiert werden.

Die Konfigurationsdatei (sfb-configuration.conf) basiert auf xml. Das Grundgerüst, das für die Anpassungen notwendig war, sieht so aus:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <config>
3   ... <!-- Anpassungen -->
4   <sourcemap>
5   ... <!-- Anpassungen -->
6   </sourcemap>
7   <datamodel>
8     <fields>
9     ... <!-- Anpassungen -->
10    </fields>
11    <entryfields>
12    ... <!-- Anpassungen -->
13    </entryfields>
14  </datamodel>
15 </config>

```

Da mit weiteren Anpassungen ohnehin eine Konfigurationsdatei benötigt wurde, konnten einige Anweisungen, die ansonsten in der Kommandozeile eingegeben werden, ausgelagert werden. Diese werden direkt unterhalb des Tags <config> eingegeben:

```

<output_fieldcase>lower</output_fieldcase>
<output_indent>1</output_indent>
<output_align>true</output_align>

```

Die nächste Anpassung erfolgt in der Umgebung `<sourcemap>`. In `<sourcemap>` wird, vereinfacht gesagt, ein »Suchen-Ersetzen«-Prinzip (nach bestimmten Bedingungen) ausgeführt. In der Standardkonfigurationsdatei⁵ gibt es ebenfalls eine `<sourcemap>`-Umgebung, die unter anderem die Umbenennung der »veralteten« Feldnamen auslöst. Damit diese wichtige Anpassung nicht verloren geht, müssen alle Zeilen aus dieser originalen `<sourcemap>`-Umgebung⁶ in die eigene Konfigurationsdatei kopiert werden:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <config>
3   <output_fieldcase>lower</output_fieldcase>
4   <output_indent>2</output_indent>
5   <output_align>true</output_align>
6   <sourcemap>
7     <!-- original sourcemap from biber-tool.conf -->
8     <maps datatype="bibtex">
9       <map>
10        <map_step map_type_source="conference" map_type_target="inproceedings"/>
11        <map_step map_type_source="electronic" map_type_target="online"/>
12        <map_step map_type_source="www" map_type_target="online"/>
13      </map>
14      <map map_overwrite="1">
15        <map_step map_final="1" map_type_source="mastersthesis" map_type_target="
16        ↳thesis"/>
17        <map_step map_field_set="type" map_field_value="mathesis"/>
18      </map>
19      <map map_overwrite="1">
20        <map_step map_final="1" map_type_source="phdthesis" map_type_target="thesis"/
21        ↳>
22        <map_step map_field_set="type" map_field_value="phdthesis"/>
23      </map>
24      <map map_overwrite="1">
25        <map_step map_final="1" map_type_source="techreport" map_type_target="report"
26        ↳>
27        <map_step map_field_set="type" map_field_value="techreport"/>
28      </map>
29      <map>
30        <map_step map_field_source="address" map_field_target="location"/>
31        <map_step map_field_source="school" map_field_target="institution"/>
32        <map_step map_field_source="annote" map_field_target="annotation"/>

```

⁵ <https://github.com/plk/biber/blob/master/data/biber-tool.conf>

⁶ <https://github.com/plk/biber/blob/bee5567eb70e86c22bbfb7696612fd9ec2146eb2/data/biber-tool.conf#L321-L352>

```

30 <map_step map_field_source="journal" map_field_target="journaltitle"/>
31 <map_step map_field_source="archiveprefix" map_field_target="eprinttype"/>
32 <map_step map_field_source="primaryclass" map_field_target="eprintclass"/>
33 <map_step map_field_source="key" map_field_target="sortkey"/>
34 <map_step map_field_source="pdf" map_field_target="file"/>
35 </map>
36 </maps>
37 </sourcemap>
38 <datamodel>
39 <fields>
40 ... <!-- Anpassungen -->
41 </fields>
42 <entryfields>
43 ... <!-- Anpassungen -->
44 </entryfields>
45 </datamodel>
46 </config>

```

In `<sourcemap>` werden die Anpassungen in einzelnen Schritten nacheinander ausgeführt (`<map_step ... />`) und zu größeren Einheiten gebündelt (`<map>...</map>` bzw. `<maps>...</maps>`).

An der Umbenennung von `address` zu `location` sieht man die Funktionsweise.

```
<map_step map_field_source="address" map_field_target="location"/>
```

In einem Bearbeitungsschritt wird innerhalb eines Datensatzes (@article, @book etc.) nach dem Feldnamen `address` gesucht (`map_field_source`) und durch den Inhalt von `map_field_target` ersetzt (`location`). Auf diese Weise lassen sich Feldnamen ändern. Feldinhalte, wie bspw. Titel, werden nachfolgend am Beispiel der erzwungenen Großschreibung gezeigt.

```
<map_step map_field_source="title" (map_match="{{[A-Z]])" map_replace="$1"/>
```

Es wird ausschließlich das Feld `title` berücksichtigt (`map_field_source`). Darin wird nach einem Großbuchstaben innerhalb eines geschweiften Klammerpaares gesucht (`map_match="{{[A-Z]])"`). Für die Suche können reguläre Ausdrücke verwendet werden, wie es auch mit `[A-Z]` (jeder Großbuchstabe von A bis Z) gemacht wird. Der Inhalt vom Klammerpaar `()` wird zwischengespeichert und bei `map_replace` mit `$1` wieder ausgegeben.

In einem nächsten Schritt soll für alle @article-Einträge das Feld `location` gelöscht werden:

```

<per_type>article</per_type>
<map_step map_field_set="location" map_null="1"/>

```

Über `<per_type><Datensatztyp></per_type>` wird festgelegt, dass die nachfolgenden Schritte nur für den gewählten Datensatztyp `article` gelten. Mit `map_null="1"` wird das Feld `location` gelöscht. Ganz analog wird auch das Feld `issn` gelöscht, nur dass in diesem Schritt keine Einschränkung hinsichtlich des Datensatztyps erfolgt.

Was jetzt noch fehlt, sind die Anpassungen, die nur bei bestimmten Kriterien ausgeführt werden dürfen. Zunächst muss geprüft werden, ob im Feld `pnm` die DFG-Fördernummer (hier: 403224013) steht.

```
<map_step map_field_source="pnm" map_final="1" map_match="\(403224013\)"/>
```

Wenn diese Bedingung nicht erfüllt ist, dann werden wegen `map_final="1"` die nachfolgenden Schritte nicht ausgeführt. Wenn die Bedingung jedoch erfüllt ist, dann soll zunächst ein Keyword gesetzt

```
<map_step map_append="1" map_field_set="keywords" map_field_value=", CRC1382"/>
```

und anschließend der vollständige Name des SFBs in das Feld `note` geschrieben werden.

```
<map_step map_field_set="note" map_field_value="SFB 1382: Die Darm-Leber-Achse --  
↪ Funktionelle Zusammenhänge und therapeutische Strategien (403224013)"/>
```

Man kann also durchaus auch komplexere Anpassungen vornehmen, die auf bedingten Anweisungen beruhen.

Der Block mit den eigenen Anpassungen sieht insgesamt wie folgt aus:⁷

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <config>
3   ...
4   <sourcemap>
5     <!-- original sourcemap from biber-tool.conf -->
6     <maps datatype="bibtex">
7       ...
8     </maps>
9     <!-- additional configurations -->
10    <maps datatype="bibtex">
11      <map>
12        <!-- In the title get rid off curly brackets -->
13        <map_step map_field_source="title" map_match="{([A-Z])}" map_replace="$1"/>
14      </map>
15      <map>
16        <!-- delete 'location' but only for @article -->
17        <per_type>article</per_type>
```

⁷ Weitere Beispiele gibt es in Kapitel 3.13 in der Dokumentation von [1].

```

18 <map_step map_field_set="location" map_null="1"/>
19 </map>
20 <map>
21 <!-- drop these fields -->
22 <map_step map_field_set="issn" map_null="1"/>
23 </map>
24 <map map_overwrite="1">
25 <!-- execute steps only if first step is true -->
26 <map_step map_field_source="pnm" map_final="1" map_match="\ (403224013\)"/>
27 <!-- append 'CRC1382' to 'keywords' -->
28 <map_step map_append="1" map_field_set="keywords" map_field_value=", CRC1382"
  ↪>/>
29 <!-- write SFB name into field 'note' -->
30 <map_step map_field_set="note" map_field_value="SFB 1382: Die Darm-Leber-
  ↪Achse -- Funktionelle Zusammenhänge und therapeutische Strategien
  ↪(403224013)"/>
31 </map>
32 </maps>
33 </sourcemap>
34 <datamodel>
35 <fields>
36 ... <!-- Anpassungen -->
37 </fields>
38 <entryfields>
39 ... <!-- Anpassungen -->
40 </entryfields>
41 </datamodel>
42 </config>

```

Die letzten beiden notwendigen Anpassungen betreffen das Feld reportid, das erhalten bleiben soll. Hierfür wird zuerst in der <fields>-Umgebung das Feld an sich definiert.

```
<field datatype="literal" fieldtype="field">reportid</field>
```

In der <entryfields>-Umgebung wird nochmals der Feldname aufgeführt:

```
<field>reportid</field>
```

Wiederum im Gesamtzusammenhang:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <config>
3   ...
4   <sourcemap>

```

```

5      ...
6      </sourcemap>
7      <datamodel>
8        <fields>
9          <!-- do not drop these fields -->
10         <field datatype="literal" fieldtype="field">reportid</field>
11       </fields>
12       <entryfields>
13         <!-- do not drop these fields -->
14         <field>reportid</field>
15       </entryfields>
16     </datamodel>
17 </config>

```

Der Befehl für die Kommandozeile muss modifiziert werden, indem mit der Eingabe von `--configfile=sfb-configuration.conf` die Konfigurationsdatei geladen wird.

```

biber --tool --outfile sfb.bib --output-field-order=author,title,date --
↳configfile=sfb-configuration.conf sfb.bib

```

Der Eintrag ist nun entsprechend gesäubert und nach den eigenen Wünschen angepasst:

```

1 @article{Babbar:782496,
2   author      = {Babbar, Anshu and Hitch, Thomas C. A. and others},
3   title       = {The Compromised Mucosal Immune System of  $\beta$ 7 Integrin-
4   ↳Deficient Mice Has Only Minor Effects on the Fecal Microbiota in Homeostasis
5   ↳},
6   date        = {2019},
7   doi         = {10.3389/fmicb.2019.02284},
8   journaltitle = {Frontiers in microbiology},
9   keywords    = {,CRC1382},
10  note        = {SFB 1382: Die Darm-Leber-Achse -- Funktionelle Zusammenhänge
11  ↳und therapeutische Strategien (403224013)},
12  pages       = {2284},
13  publisher    = {Frontiers Media},
14  reportid     = {RWTH-2020-01913},
15  url         = {https://publications.rwth-aachen.de/record/782496},
16  volume      = {10},
17 }

```


Fazit

Es wurde gezeigt, wie man über den Werkzeugmodus von biber beliebig viele Bibliografieeinträge nach eigenen Wünschen und Vorgaben bearbeiten kann. Dabei lässt sich der Prozess über die Kommandozeile steuern, sodass dies in einen vollautomatisierten Ablauf eingebunden werden kann, der sich vom Abrufen der Daten aus einem OPAC, über die Bereinigung und Anpassung bis zur Erstellung von Publikationslisten kümmert und darüber hinaus die Daten auf einen Server lädt, um sie mit einer Instanz von *bibtex-browser*⁸ darzustellen.

Dank

»DFG (German Research Foundation) – Project-ID 403224013 – SFB 1382«

Literatur und Software

- [1] Philip Kime, François Charette: The Biber package, A Bib_{TE}X replacement for users of Bib_{LA}T_EX, Version 2.14, 2019, CTAN: /biblio/biber (besucht am 6. 3. 2020).

Eine weitere Schrift für Menschen mit Leseschwäche

Herbert Voß

Digitale Schriften für Menschen mit einer Leseschwäche sind sehr selten. Hier soll eine Neuentwicklung vorgestellt werden.

Vor zwei Jahren hatte ich hier die Schrift »Dyslexie« vorgestellt, die speziell für Menschen mit einer ausgebildeten Leseschwäche geeignet ist. [3] Die häufigsten Lesefehler bei einer Leseschwäche [2] sind das Vertauschen, Spiegeln, Ändern, Drehen und Verschmelzen von Buchstaben. Dies gilt es bei der Entwicklung einer Schrift zu beachten. Mit »Luciole« liegt nun eine weitere Schrift vor, die das Lesen von Texten für diese Personengruppe erleichtern soll [1]:

‘Luciole’ (French for “firefly”) is a new typeface developed explicitly for visually impaired people. The result of a two-year collaboration between the Centre Technique Régional pour la Déficience Visuelle (the Regional Technical Center for Visual Impairment) and the type-design studio typographies. fr, this project received a grant from the

⁸ <https://www.monperrus.net/martin/bibtexbrowser/>

Swiss Ceres Foundation and support from the DIPHE laboratory at the Université Lumière Lyon 2.

Auch für »Luciole« gelten dieselben Designkriterien, wie sie schon in [3] angegeben wurden. Eine Abbildung soll nur knapp wesentliche Unterschiede zeigen: größere Liniendicke, ausgeprägte Satz- und Akzentzeichen. Zum besseren Vergleich wurde die »Luciole« herunterskaliert, um die gleiche Textbreite für die Gegenüberstellung zu erreichen. Grundsätzlich sind die Zeichen der »Luciole« größer als die der Latin Modern.

Latin Modern: **lésibilité abcbdudjl.,H**
 Luciole: **lésibilité abcbdudjl.,H**

Die Schrift Luciole umfasst die serifenlosen Schriftschnitte Regular, Italic, Bold und BoldItalic im TrueType-Format:

Luciole-Regular.ttf
 Luciole-Bold.ttf
 Luciole-Regular-Italic.ttf
 Luciole-Bold-Italic.ttf

Für Luciole gilt die freie »Creative Commons Attribution license« und sie kann unter der oben angegebenen Webseite heruntergeladen werden. Für die Verwendung mit $\text{\LaTeX}$ oder $\text{\LuaTeX}$ kann sie als Hauptschrift oder alternativ per Schriftschalter, wie folgt, verfügbar gemacht werden:

```
\usepackage{fontspec}
\setmainfont{Luciole}%           Dokumentenweite Festlegung
\newfontfamily\Luciole{Luciole}% Schalter \Luciole
```

In den folgenden Schriftbeispielen wird die Größe `\small` angewendet.

Regular:

Die Schrift Luciole umfasst die serifenlosen Schriftschnitte Regular, Italic, Bold und BoldItalic im TrueType-Format. Sie kann kostenfrei von der angegebenen Webseite heruntergeladen werden und lässt sich einfach, wie für das jeweilige Betriebssystem üblich, installieren.

Bold:

Die Schrift Luciole umfasst die serifenlosen Schriftschnitte Regular, Italic, Bold und BoldItalic im TrueType-Format. Sie kann kostenfrei von der angegebenen Webseite heruntergeladen werden und lässt sich einfach, wie für das jeweilige Betriebssystem üblich, installieren.

Italic:

Die Schrift Luciole umfasst die serifenlosen Schriftschnitte Regular, Italic, Bold und BoldItalic im TrueType-Format. Sie kann kostenfrei von der angegebenen Webseite heruntergeladen werden und lässt sich einfach, wie für das jeweilige Betriebssystem üblich, installieren.

Bold-Italic:

Die Schrift Luciole umfasst die serifenlosen Schriftschnitte Regular, Italic, Bold und BoldItalic im TrueType-Format. Sie kann kostenfrei von der angegebenen Webseite heruntergeladen werden und lässt sich einfach, wie für das jeweilige Betriebssystem üblich, installieren.

Die Schrift »Luciole« beinhaltet alle wesentlichen Zeichen für den europäischen Sprachraum. Abbildung 1 zeigt eine entsprechende Zusammenstellung.



Abb. 1: Zusammenstellung der vorhandenen Zeichen der Schrift »Luciole«

Literatur und Software

- [1] Laurent Bourcellier, Jonathan Perez: Luciole, 2019, <https://www.luciole-vision.com/luciole-en.html> (besucht am 3. 4. 2020).
- [2] Britta Büchner u. a.: Die medizinische Diagnose »Legasthenie« ist irreführend und schadet den Interessen der Kinder, Stellungnahme zur Leitlinie: »Diagnostik und Behandlung von Kindern und Jugendlichen mit Lese- und/oder Rechtschreibstörung«, http://www.legakids.net/fileadmin/user_upload/Downloads/Info/Wissenschaft/LegaKids_Stellungnahme_Leitlinien_Mai_2015_01.pdf (besucht am 3. 4. 2020).
- [3] Herbert Voß: »Schriften für Menschen mit einer Leseschwäche«, *Die T_EXnische Komödie*, 30.1 (2018), 54–56.

Verschiedene Möglichkeiten zum Kommentieren und Dokumentieren von Code

Lukas C. Bossert

Dieser Artikel ist Herbert Voß anlässlich seines Abschiedes aus der Redaktion der T_EXnischen Komödie gewidmet. In der Redaktion sowie bei Seminaren oder auf TeX.SE hat er sicherlich nicht nur mir, sondern vielen anderen mit hilfreichen Kommentaren zum Code geholfen.

Selbst geschriebener Code ist Forschungsleistung: Eigene Gedanken und Ideen fließen in die Software/das Skript ein und sind daher mitunter erklärungsbedürftig. Diese Erklärungen zu liefern ist nicht immer eine leichte Aufgabe.¹ Dabei spielt es an sich keine Rolle, in welcher Programmiersprache der Code verfasst wird. Die Dokumentation ist hinsichtlich eines guten Forschungsdatenmanagements unerlässlich, wobei die Kommentare die Metadaten sind, die das notwendige Verständnis für die Forschungsdaten (Code) liefern.

Es gibt verschiedene Wege, diese Metadaten bereitzustellen. Dieser Beitrag zeigt drei Möglichkeiten dazu auf. Ausgehend von der einfachen Art anhand des Kommentarzeichens, dann mittels spezieller Pakete bis hin zu *literate programming*, als eine formvollendete Art Code zu beschreiben. Die Prinzipien des Kommentierens

¹ Viele hilfreiche Tipps und Anmerkungen zum Kommentieren von Code finden sich im äußerst lesenswerten Buch von *Weniger schlecht programmieren* von Passig und Jander, besonders die Seiten 61–76. Einige dieser Tipps sind in den Artikel eingeflossen. Andere sagen, es sollte nicht dokumentiert werden, was der Code tut, sondern vielmehr warum; siehe dazu [11, S. 243–248]. Manchmal ist das »Was« (speziell für Anfänger) ebenso wichtig.

und Dokumentierens können auf andere Sprachen übertragen werden, nachfolgend werden sie für L^AT_EX erläutert.

Für Anfänger: Dokumentieren in Kommentaren

Die schnellste und unkomplizierteste Form der Kommentierung von Code ist die mittels %. Dabei schreibt man nach dem Prozentzeichen seinen Kommentar, welcher dann vom Compiler ignoriert wird. Als T_EX-Anfänger habe ich diese Form des Kommentierens gern genutzt, um nach dem Aufruf des Pakets dessen Funktion zu notieren, oder die der einzelnen Optionen.

```
1 \usepackage[
2   left  = 2cm, % Abstand links/innen
3   right = 4cm, % Abstand rechts/außen
4   top   = 3cm, % Abstand oben
5 ]{geometry} % Paket kümmert sich um Seitenmaße etc.
```

Sobald man anfängt, größere Code-Blöcke zu tippen, die man beispielsweise aus Internetforen hat, ist es äußerst hilfreich, sich die Quelle zu notieren, damit man gegebenenfalls dort nochmals nachschauen kann.

Listing 1: Beispiel entnommen von <https://tex.stackexchange.com/a/524900/98739>.

```
1 %COMMAND FOUND ON INTERNET AND WORKS
2 % \printlist[<sep>]{<list macro>}
3 \newcommand{\printlist}[2][,]{% Print list
4   % http://tex.stackexchange.com/a/89187/5764
5   \def\listsep{\def\listsep{#1}}% Delayed execution of list separator
6   \renewcommand{\do}[1]{\listsep`##1'}%
7   [\dolistloop\language\list]
8 }
```

Ebenso ist es hilfreich, wenn man einen neuen Befehl mit Argumenten einführt, die Belegung der Argumente in einem Kommentar festzuhalten.

```
1 %% Argumentbelegung:
2 %% 1: [optional] Farbe
3 %% 2: {pflicht} Schlagwort
4 %% 3: {pflicht} Erklärung
5 %% 4: {pflicht} Schlusszeichen
6 \newcommand\MeinNeuerBefehl[3][black]{%
7   \bgroup
8   \color{#1}
9   \textbf{#2}: #3#4
10  \egroup
```

```
11 }
```

Es gibt bei dieser Kommentiermöglichkeit ein kleines Caveat: Man sollte darauf achten, dass das Kommentarzeichen % nicht direkt ans Wortende gesetzt wird, da ansonsten ein notwendiges Leerzeichen »verschluckt« wird. Also lieber % erst nach einem vorangegangenen Leerzeichen setzen.

Es zeigt sich, dass der Gebrauch von %, speziell für T_EX-Neulinge sehr zu empfehlen ist: Man kann die wichtigen Informationen direkt beim auszuführenden Code unterbringen, um dort gegebenenfalls noch einmal nachzuschauen wie beispielsweise ein Befehl funktioniert. Inline- und mehrzeilige Kommentare haben unbestritten Vorteile und man sollte darauf nicht verzichten. [vgl. 9, S. 61–76]

Für Fortgeschrittene: Dokumentieren anhand von Paketen

Während die Nutzung von % mit anschließendem Kommentar besonders für das eigene Verständnis von Code hilfreich ist, braucht es eine andere Art der Dokumentation, wenn man einer Leserschaft Code präsentiert: Möchte man ausführbaren Code sichtbar in ein PDF setzen, um dessen Funktionsweise zu beschreiben, bedarf es einer anderen Herangehensweise.

Kurze Wörter oder Befehle kann man in einem Fließtext kommentieren und den Code mittels `\verb|befehl|` ausgeben: Alles, was zwischen den beiden gleichen Zeichen nach `\verb` steht, wird buchstabengetreu (verbatim) abgedruckt. Es findet in dieser Umgebung also keine Verarbeitung von Befehlen oder sonstigem Code statt. Für große Code-Blöcke ist diese Form der Kommentierung aber nicht geeignet. Dafür greift man besser zu Paketen, die darauf spezialisiert sind. Für Code-Darstellung sind besonders die Pakete `listings` [7], `minted` [10] oder `tcolorbox`² [13] verbreitet. Jedes Paket hat seine eigenen Vor- und Nachteile und es lohnt sich, in deren Dokumentationen einen Blick zu werfen.

Ich greife `listings` für ein Beispiel heraus. Der Code wird in eine `lstlisting`-Umgebung gesetzt. Davor oder danach kann man den Code dokumentieren:

²`tcolorbox` lädt selbst das Paket `listings`. Die Darstellungsweise von Code und der »Ausgabe« ist mit `tcolorbox` jedoch vielfältig modifizierbar, sodass es hier zusätzlich genannt wird.

Um eine Liste mit Aufzählungen zu erstellen, eignet sich die Umgebung `itemize`. Für jedes Aufzählungszeichen verwendet man `\item`. `\item` hat nur gegebenenfalls ein optionales Argument, dessen Inhalt als lokales Aufzählungszeichen übernommen wird. Das nachfolgende Beispiel erstellt eine Liste mit drei Punkten, wobei das letzte Aufzählungszeichen ein `!` ist.

```
\begin{itemize}
  \item \LaTeX
  \item \LuaTeX
  \item[!] \TeX
\end{itemize}
```

Um eine Liste mit Aufzählungen zu erstellen, eignet sich die Umgebung `\verb\itemize|`. Für jedes Aufzählungszeichen verwendet man `\verb\item|`. `\verb\item|` hat nur gegebenenfalls ein optionales Argument, dessen Inhalt als lokales Aufzählungszeichen übernommen wird. Das nachfolgende Beispiel erstellt eine Liste mit drei Punkten, wobei das letzte Aufzählungszeichen ein `\verb|!` ist.

```
\begin{lstlisting}
\begin{itemize}
  \item \LaTeX
  \item \LuaTeX
  \item[!] \TeX
\end{itemize}
\end{lstlisting}
```

Die genannten Pakete haben zusätzlich die Möglichkeit, Wörter des in der Umgebung stehenden Codes hervorzuheben, die bspw. typisch für die Programmiersprache sind. Dies lässt sich auf globaler oder lokaler Ebene, das heißt auch für jede `lstlisting`-Umgebung individuell, einstellen. Dies wird im optionalen Argument definiert.

```
\newcommand\myVerbatim[1]{%
  \texttt{#1}}
```

```
\begin{lstlisting}[basicstyle=\ttfamily,
  keywordstyle=\bfseries,
  language={\AlLaTeX}{TeX}]
\newcommand\myVerbatim[1]{\texttt{#1}}
\end{lstlisting}
```

Will man die Hervorhebung für `LATEX` jedoch innerhalb des gesamten Dokuments bestimmen, dann übergibt man in der Präambel die Sprache dem Befehl `\lstset`:

```
\lstset{language={\AlLaTeX}{TeX}}
```

Der Befehl `\lstlistinginput{<Pfad/Datei.Endung>}` ist für die Code-Dokumentation von ganzen Dateien geeignet, anhand dessen der komplette Inhalt einer externen Datei buchstabengetreu in die PDF eingefügt wird. Diese Funktion ist hilfreich, wenn man einzelne Code-Blöcke der externen Datei kommentiert, die man passagenweise (`linrange=<Start>-<Ende>`) einfügt.

Im nachfolgenden Beispiel wird der Code einer fiktiven Dokumentenklasse beschrieben.

```

1 Zunächst werden alle neuen Dokumentenbefehle eingeführt.
2 \lstinputlisting[linerange=4-10]{my-class.cls}
3 und anschließend findet die Definition der Farbwelt
4 entsprechend dem Corporate Design statt:
5 \lstinputlisting[linerange=11-16]{my-class.cls}

```

Diese Vorgehensweise ist jedoch aufwendig, da sich der Code in der externen Datei ändern kann und man dann die Zeilenauswahl händisch nachjustieren muss.

So praktisch die `lstlisting`-Umgebung ist, eignet sie sich vor allem für kurze Code-Blöcke. Für die Dokumentation von Code, wie beispielsweise in Dokumentenklassen oder Bibliografie-Stilen, bedienen sich versierte Nutzer eines Vorgehens, das man in etwa mit *literarischem Programmieren* übersetzen kann. Davon handelt der nächste Abschnitt.

Für Experten: *Literate Programming*

»Documentation must be regarded as an integral part of the process of design and coding. A good programming language will encourage and assist the programmer to write clear, self-documenting code, and even perhaps to develop and display a pleasant style of writing.« [3, S. 195]

Diesem Ansatz hatte sich auch Knuth verschrieben, als er *literate programming* [4] für die Dokumentation von *The T_EXbook* [5] entwickelte.³ Ein Text, der auf angenehme Weise Code beschreibt.

Das Besondere von *literate programming* ist, dass innerhalb einer Datei den Programm-Code und gleichzeitig auch die Dokumentation enthalten ist. In zwei Schritten wird zuerst der Programm-Code extrahiert und in die entsprechende Datei geschrieben. Anschließend wird die Dokumentation erstellt, in der die einzelnen Programm-Code-Blöcke beschrieben sind.

Zwei Dateien werden in der Regel benötigt, um Programm-Code und Dokumentation zu erstellen: Die Datei mit dem dokumentierten Programm-Code (`.dtx`-Datei) und eine Datei, in der unter anderem Anweisungen enthalten sind, welche Code-Blöcke in welche Datei geschrieben werden sollen (`.ins`-Datei).

Der Clou bei diesem Verfahren ist, dass man die Zeilen des Programm-Codes direkt kommentieren kann, ohne dass man das Dokument wechseln oder den Programm-Code anschließend händisch in die entsprechende Code-Datei kopieren müsste.

³ Siehe die Sammlung an Texten zu *literate programming* in [6].

Dieses Vorgehen ist der de-facto-Standard für die Dokumentation von L^AT_EX-Paketen.⁴ Mittlerweile ist diese Form der Code-Dokumentation weit verbreitet und wird auch für andere Programmiersprachen verwendet.⁵

Die Dokumentation eines Beispiel-Pakets

Im folgenden Abschnitt wird anhand eines fiktiven (aber funktionierenden) Pakets namens *MeinPaket* die Funktionsweise des *literate programming* gezeigt.

Die Datei zum Paket *MeinPaket* heißt *MeinPaket.dtx*.⁶ In der Datei sind der eigentliche Code sowie die Dokumentation hinterlegt. Hier folgt zunächst einmal der Code, die Detailbeschreibung wird danach angegangen.

Listing 2: Dokumentation und Programm-Code des fiktiven Pakets *MeinPaket*.

```

1 % \iffalse
2 %<batchfile>
3 \begingroup % startet eine Gruppe
4 \input docstrip.tex
5 \keepsilent % kein detailliertes Feedback
6 \askforoverwritfalse % überschreibe alte Version
7 \generate{\file{MeinPaket.sty}{\from{\jobname.dtx}{sty}}}
8 \endgroup % beendet die Gruppe
9 %</batchfile>
10 %<driver>
11 \documentclass{ltxdoc}
12 \author{Lukas C. Bossert}
13 \title{Mein erstes \LaTeX-Paket}
14 \usepackage{MeinPaket} % lädt das neue Paket
15 \begin{document}
16 \maketitle % erstellt eine Titelseite
17 \DocInput{\jobname.dtx} % fügt sich selbst ein
18 \end{document}
19 %</driver>
20 %<sty>
21 % \fi
22 %\section{Überblick}
23 % Mit dem Paket lässt sich die Schriftfarbe
24 % und Schriftgröße innerhalb einer Umgebung ändern.
```

⁴ Siehe etwa <https://www.ctan.org/help/upload-pkg>. Zum Thema Paketdokumentation (und zur Frage, ob diese auf deutsch oder auf englisch verfasst werden kann/soll), siehe [12, S. 32 f.].

⁵ <http://www.literateprogramming.com/index.html>

⁶ Der vollständige Code ist online hinterlegt: <https://gist.github.com/LukasCBossert>, dann unter *MeinPaket.dtx*

```

25 %\section{Code-Dokumentation}
26 % Als erstes wird die Grundvoraussetzung für das
27 % Paket spezifiziert,
28 % anschließend erfolgt die Namensnennung.
29 % \begin{macrocode}
30 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
31 \ProvidesPackage{MeinPaket.sty}
32 % \end{macrocode}
33 % Es folgt der Code des Pakets. \par
34 % Das Paket "MeinPaket" lädt ein anderes Paket.
35 % \begin{macrocode}
36 \RequirePackage{xcolor}
37 % \end{macrocode}
38 % Es stellt auch eine eigene Umgebung zur Verfügung.
39 % \begin{environment}{MeinBeispiel}
40 % Innerhalb der Umgebung \meta{MeinBeispiel}
41 % wird die Schriftfarbe und -größe geändert.
42 % \begin{macrocode}
43 \newenvironment{MeinBeispiel}
44 % \end{macrocode}
45 % Zu Beginn wird die Schrift rot eingefärbt und
46 % vergrößert.
47 % \begin{macrocode}
48 {\color{red}\large}
49 % \end{macrocode}
50 % Am Ende wird alles wieder in den ursprünglichen
51 % Zustand versetzt.
52 % \begin{macrocode}
53 {\normalcolor\normalsize}
54 % \end{macrocode}
55 % \end{environment}
56 % Nun erfolgt noch ein \begin{MeinBeispiel} Beispieltext\end{MeinBeispiel}
57 % wie er durch das Paket \meta{MeinPaket} verändert wird.
58 % \iffalse
59 %</sty>
60 % \fi
61 %\endinput

```

Wenn die Datei kompiliert wird, lädt sie sich selbst zweimal: In Zeile 7 wird aus dem Bereich zwischen `%<sty>` und `%</sty>` die Datei `MeinPaket.sty` extrahiert. In Zeile 17 wird alles noch einmal eingelesen, was nicht zwischen einem `\iffalse` und einem `\fi` steht (also alles außer Zeile 1–21 und 58–60). Besonderheit bei diesem zweiten Kompilierdurchgang ist, dass der normale Code gewissermaßen verbatim

ausgegeben wird. Alles, was hinter einem % steht, wird als normaler Code ausgeführt. So entsteht die Dokumentations-PDF. Der Bereich um Zeile 17 (genauer: alles, was zwischen `%<driver>` und `%</driver>` steht) ist das eigentliche $\TeX$ -Dokument, aus dem die PDF erstellt wird.

Gehen wir das Dokument im Einzelnen durch. Wir sehen in Zeile 2 `%<batchfile>` was mit `%</batchfile>` in Zeile 9 korrespondiert. Dazwischen steht die Information, welche Code-Blöcke in welche Dateien zu schreiben sind. Es braucht also in unserem Fall keine externe `.ins`-Datei, da sie in die `.dtx`-Datei integriert ist.

```

1 %<batchfile>
2 \begingroup % startet eine Gruppe
3 \input docstrip.tex
4 \keepsilent % kein detailliertes Feedback
5 \askforoverwritfalse % überschreibe alte Version
6 \generate{\file{MeinPaket.sty}{\from{\jobname.dtx}{sty}}}
7 \endgroup % beendet die Gruppe
8 %</batchfile>

```

Mit `\keepsilent` wird unterdrückt, dass Zeile für Zeile ein sehr detailliertes (und nicht wirklich hilfreiches) Feedback in der Kommandozeile ausgegeben wird. `\askforoverwritfalse` überschreibt vorhandene Versionen einer Datei beim Ausführen der `.dtx`-Datei. Dieser Befehl sollte gesetzt sein, da man ansonsten bei jedem Kompilieren eine JA/NEIN-Abfrage beantworten muss. Das Makro `\generate{<Ziel>}{<Quelle>}` sorgt für die Verteilung der Code-Blöcke. Fangen wir von hinten an: Alle `<sty>`-Blöcke (siehe unten) aus der Datei `\jobname.dtx` werden in die Datei `MeinPaket.sty` geschrieben.

```

1 %<driver>
2 \documentclass{ltxdoc}
3 \author{Lukas C. Bossert}
4 \title{Mein erstes \LaTeX-Paket}
5 \usepackage{MeinPaket} % lädt das neue Paket
6 \begin{document}
7 \maketitle % erstellt eine Titelseite
8 \DocInput{\jobname.dtx} % fügt sich selbst ein
9 \end{document}
10 %</driver>

```

Der zweite Block (`%<driver>...</driver>`) beinhaltet alles, was das eigentliche Paket ausmacht. Wir sehen, es ist der Code für ein normales $\LaTeX$ -Dokument: In Zeile 14 wird das Paket *MeinPaket* geladen, sodass wir es gleich innerhalb des Dokuments nutzen können.

In Zeile 20 beginnt kurz vor dem schließenden `\fi` mit `%<{*sty}` der elementare Code-Block des Pakets. In Zeile 59 wird er wieder geschlossen.

Im zweiten Abschnitt ist auffallend, dass (fast) alle Zeilen auskommentiert sind. Dies ist ein wesentliches Merkmal des Programm-Code-Abschnitts. Text, der als Beschreibung in die Dokumentation kommt, hat ein vorangestelltes `%`. Man kann also die Beschreibung wie gewohnt verfassen, mit `\section{<NAME>}` zur Unterteilung etc., man muss nur auf das `%` am Zeilenanfang achten.

Ein Programm-Code für das Paket wird in die `macrocode`-Umgebung gesetzt. Entscheidend ist die Einrückung von `\begin{macrocode}` um genau vier (4) Leerzeichen nach `%`. Gleiches gilt auch für `\end{macrocode}`. Der Programm-Code selbst wird ohne `%` oder Einrückung geschrieben.

```
1 % \begin{macrocode}
2 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
3 \ProvidesPackage{MeinPaket.sty}
4 % \end{macrocode}
```

Es gibt noch die Umgebung `environment` mit einem notwendigen Argument für den Umgebungsnamen. Der Umgebungsname wird in der Dokumentation in die Marginalspalte gesetzt. Innerhalb der `environment`-Umgebung kann wiederum mit `\begin{macrocode}... \end{macrocode}` gearbeitet werden. Die Beschreibung des Codes ist dabei stets außerhalb der `macrocode`-Umgebung.

```
1 % \end{macrocode}
2 % Es folgt der Code des Pakets. \par
3 % Das Paket "MeinPaket" lädt ein anderes Paket.
4 % \begin{macrocode}
```

So enthält also eine einzige Datei die Anweisungen, wie aus ihr selbst die eigentliche Paket-sty-Datei erstellt werden muss und gleichzeitig die Dokumentations-PDF.

Extraktion von Programm-Code und Dokumentationserstellung

Um nun die Paket-Datei und deren Dokumentation zu erstellen, muss über die Kommandozeile der Befehl

```
lualatex MeinPaket.dtx
```

ausgeführt werden. Das Ergebnis ist die Paket-Datei, wie sie in Listings 3 abgedruckt ist und die dazugehörige Dokumentation (Abb. 1).

Das Beispiel mit dem fiktiven Paket war nur ein kurzer Einblick, was mittels *literate programming* an sich möglich ist. Allen Interessierten sei das Tutorial von Pakin wärmstens empfohlen [8]. Dort gibt es ausführlichere Beschreibungen zu den

einzelnen Befehlen und Hintergrundinformationen, was $\text{\TeX}$ -nisch beim *literate programming* passiert.

Fazit

Die drei vorgestellten Möglichkeiten zur Dokumentation und Kommentierung von Code haben Vor- und Nachteile. Es ist nicht nur eine Frage der persönlichen $\text{\LaTeX}$ -Kompetenz, eine der drei Varianten zu verwenden, sondern auch eine Frage, des richtigen Einsatzes der Dokumentation- und Beschreibungsart: Anhand von Kommentaren direkt am Code wird dieser für einen selbst und für eine Gruppe mit der man am Code arbeitet auch später noch nachvollziehbar bleiben. Es ist die schnellste und oft auch effizienteste Methode für kurze Code-Sequenzen.

Der Einsatz von Pakten, allen voran `listing`, lohnt sich dann, wenn man das Ergebnis von ausführbarem Code unmittelbar daneben zeigen möchte.

Literate programming sollte vor allem dann zum Einsatz kommen, wenn man viel Code, wie beispielsweise bei einem Paket oder Dokumentenklasse, transparent dokumentieren möchte. Die Effizienz dieser Methode ist beeindruckend, besonders, wenn man dabei noch eine `makefile`-Datei einsetzt, die sich um die Datei-Erstellung und -Verwaltung kümmert.⁷ Bei großen Code-Projekten ist diese literarische Dokumentationsart sicherlich die schönste Form, Code zu lesen.

Listing 3: Programm-Code des fiktiven Pakets `MeinPaket`.

```
1 %%  
2 %% This is file `MeinPaket.sty',  
3 %% generated with the docstrip utility.  
4 %%  
5 %% The original source files were:  
6 %%  
7 %% MeinPaket.dtx (with options: `sty')  
8 %%  
9 %% IMPORTANT NOTICE:  
10 %%  
11 %% For the copyright see the source file.  
12 %%  
13 %% Any modified versions of this file must be renamed  
14 %% with new filenames distinct from MeinPaket.sty.  
15 %%  
16 %% For distribution of the original source see the terms  
17 %% for copying and modification in the file MeinPaket.dtx.
```

⁷ Beeindruckend ist dies beim `biblatex`-Stil `oxref` umgesetzt [1]. Zum Einsatz von `makefile`-Dateien siehe Bossert [2].

```
18 %%  
19 %% This generated file may be distributed as long as the  
20 %% original source files, as listed above, are part of the  
21 %% same distribution. (The sources need not necessarily be  
22 %% in the same archive or directory.)  
23 \NeedsTeXFormat{LaTeX2e}[2005/12/01]  
24 \ProvidesPackage{MeinPaket.sty}  
25 \RequirePackage{xcolor}  
26 \newenvironment{MeinBeispiel}  
27 {\color{red}\large}  
28 {\normalcolor\normalsize}  
29 \endinput  
30 %%  
31 %% End of file `MeinPaket.sty'.
```

Dank

Ich danke Florian Claus und Philipp Pilhofer herzlich für Anmerkungen und Korrekturen.

Mein erstes L^AT_EX-Paket

Lukas C. Bossert

March 11, 2020

1 Überblick

Mit dem Paket lässt sich die Schriftfarbe und Schriftgröße innerhalb einer Umgebung ändern.

2 Code-Dokumentation

Als erstes wird die Grundvoraussetzung für das Paket spezifiziert, anschließend erfolgt die Namensnennung.

```
1 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
2 \ProvidesPackage{MeinPaket.sty}
```

Es folgt der Code des Pakets.

Das Paket "MeinPaket" lädt ein anderes Paket.

```
3 \RequirePackage{xcolor}
```

Es stellt auch eine eigene Umgebung zur Verfügung.

MeinBeispiel Innerhalb der Umgebung *(MeinBeispiel)* wird die Schriftfarbe und -größe geändert.

```
4 \newenvironment{MeinBeispiel}
```

Zu Beginn wird die Schrift rot eingefärbt und vergrößert.

```
5 {\color{red}\large}
```

Am Ende wird alles wieder in den ursprünglichen Zustand versetzt.

```
6 {\normalcolor\normalsize}
```

Nun erfolgt noch ein **Beispieltext** wie er durch das Paket *(MeinPaket)* verändert wird.

Abb. 1: Dokumentation des fiktiven Pakets MeinPaket.

Literatur und Software

- [1] Alex Ball: The Biblalex-oxref package, BibLaTeX styles inspired by the Oxford Guide to Style, Version 2.0.1, 2020, <https://github.com/alex-ball/biblalex-oxref> (besucht am 16. 3. 2020).
- [2] Lukas C. Bossert: »Zur Nutzung von makefile-Dateien«, *Die T_EXnische Komödie*, 31.2 (2019), 64–71.
- [3] Charles Antony Richard Hoare, »Hints on programming language design« in Computer Systems Reliability, (Hrsg.: C. Bunyan), State of the Art Report 20, 1973, S. 193–216, <http://flint.cs.yale.edu/cs428/doc/HintsPL.pdf> (besucht am 6. 9. 2018).
- [4] Donald E. Knuth: »Literate Programming«, *The Computer Journal*, 27.2 (1984), 97–111, DOI 10.1093/comjnl/27.2.97.
- [5] — The T_EXbook, Addison-Wesley, 1984.
- [6] — Literate Programming, CSLI Lecture Notes 27, Cambridge University Press, Cambridge, 1992.
- [7] Brooks Moses, Jobst Hoffmann, Carsten Heinz: The Listings package, Typeset source code listings using LaTeX, Version 1.8c, 2019, <http://www.ctan.org/pkg/listings> (besucht am 16. 3. 2020).
- [8] Scott Pakin: The Dtxut package, Tutorial on writing .dtx and .ins files, Version 2.1, 2015, <http://www.ctan.org/pkg/dtxut> (besucht am 16. 3. 2020).
- [9] Kathrin Passig, Johannes Jander: Weniger schlecht programmieren, O'Reilly Germany, Köln, 2013.
- [10] Geoffrey Poore, Konrad Rudolph: The Minted package, Highlighted source code for LaTeX, Version 2.5, 2017, <http://www.ctan.org/pkg/minted> (besucht am 16. 3. 2020).
- [11] Uwe Post: Besser coden, Rheinwerk Verlag, Bonn, 2018.
- [12] Christine Römer: »Pakete in Deutsch dokumentieren«, *Die T_EXnische Komödie*, 23.2 (2011), 28–36, https://archiv.dante.de/DTK/PDF/komoedie_2011_2.pdf (besucht am 11. 3. 2020).
- [13] Thomas F. Sturm: The Tcolorbox package, Coloured boxes, for LaTeX examples and theorems, etc, Version 4.22, 2019, <http://www.ctan.org/pkg/tcolorbox> (besucht am 16. 3. 2020).

Kommaseparierte Listen als Tabellen und Grafiken darstellen

Herbert Voß

Es wird gezeigt, wie man eine kommaseparierte Liste (csv) als Tabelle mit Hilfe von `pgfplotstable` und als Grafik mit Hilfe von `pgfplots` darstellen kann.

Aus dem Netz wurde folgende Liste importiert, die nur um die Zeile der Spaltennamen erweitert wurde:

```
# 12.04.2020, 00:00 Uhr,  
n, Tote, Bundesland  
24078, 649, Baden-Württemberg  
32282, 786, Bayern  
4567, 50, Berlin  
1857, 38, Brandenburg  
480, 15, Bremen  
3742, 56, Hamburg  
5859, 123, Hessen  
605, 11, Mecklenburg-Vorpommern  
7602, 189, Niedersachsen  
24267, 495, Nordrhein-Westfalen  
4734, 64, Rheinland-Pfalz  
2058, 43, Saarland  
3600, 67, Sachsen  
1166, 21, Sachsen-Anhalt  
2118, 41, Schleswig-Holstein  
1464, 25, Thüringen
```

Für eine Darstellung als L^AT_EX-Tabelle müsste man normalerweise die Kommata durch den Spaltentrenner und das Zeilenende durch `\\` ersetzen. Mit Hilfe des Pakets `pgfplotstable` lässt sich dies automatisieren:

```
\usepackage{booktabs, pgfplotstable, subcaption}
```

```
\begin{table}[!htb]  
\caption{Automatisch generierte Tabelle einer \texttt{.csv}-Datei.}\label{table1.  
↪csv}  
\begin{subtable}{0.49\linewidth}\centering  
\caption{Die Originalliste}  
\pgfplotstabletypeset[
```

```

use comma,
col sep=comma,
columns/n/.style={column type=S,column name={\emph{Fälle}}, dec sep align},
columns/Tote/.style={column type=S, dec sep align},
columns/Bundesland/.style={column type=l, string type, column name={\emph{
↪Bundesland}}},
every head row/.style={
  before row={\toprule}, % have a rule at top
  after row={\midrule}
},
every last row/.style={after row=\bottomrule}, % rule at bottom
]{table.csv} % full path and filename
\end{subtable}
\hfill
\begin{subtable}{0.49\linewidth}\centering
\caption{Die Liste sortiert nach den Fallzahlen.}
\pgfplotstableread[
  sort, sort cmp={int <}, sort key=n, % sort the data
  use comma,
  col sep=comma,
  columns/n/.style={column type=S,column name={\emph{Fälle}}, dec sep align},
  columns/Tote/.style={column type=S, column name={\emph{Tote}}, dec sep align},
  columns/Bundesland/.style={column type=l, string type, column name={\emph{
↪Bundesland}}},
  every head row/.style={
    before row={\toprule}, % have a rule at top
    after row={\midrule}
  },
  every last row/.style={after row=\bottomrule}, % rule at bottom
]{table.csv} % full path and filename
\end{subtable}
\end{table}

```

Table 1: Automatisch generierte Tabelle einer .csv-Datei.

(a) Die Originalliste			(b) Die Liste sortiert nach den Fallzahlen.		
Fälle	Tote	Bundesland	Fälle	Tote	Bundesland
24.078	649	Baden-Württemberg	480	15	Bremen
32.282	786	Bayern	605	11	Mecklenburg-Vorpommern
4.567	50	Berlin	1.166	21	Sachsen-Anhalt
1.857	38	Brandenburg	1.464	25	Thüringen
480	15	Bremen	1.857	38	Brandenburg
3.742	56	Hamburg	2.058	43	Saarland
5.859	123	Hessen	2.118	41	Schleswig-Holstein
605	11	Mecklenburg-Vorpommern	3.600	67	Sachsen
7.602	189	Niedersachsen	3.742	56	Hamburg
24.267	495	Nordrhein-Westfalen	4.567	50	Berlin
4.734	64	Rheinland-Pfalz	4.734	64	Rheinland-Pfalz
2.058	43	Saarland	5.859	123	Hessen
3.600	67	Sachsen	7.602	189	Niedersachsen
1.166	21	Sachsen-Anhalt	24.078	649	Baden-Württemberg
2.118	41	Schleswig-Holstein	24.267	495	Nordrhein-Westfalen
1.464	25	Thüringen	32.282	786	Bayern

Die Ausgabe als Grafik kann mit dem Paket pgfplots erfolgen, welches faktisch eine Erweiterung von pgfplotstable darstellt. Für die grafische Darstellung benötigt man sinnvolle Angaben für die Beschriftung der Achsen. Die Namen der Bundesländer sollen wegen ihrer Länge auf der *y*-Achse erscheinen, während die Zahl der positiven Fälle (*n*) auf der *x*-Achse erscheinen soll.

Zuerst wird die Tabelle mit \pgfplotstableread eingelesen und im Makro \myTable gespeichert. Danach wird diese mit \pgfplotstablesort nach der Zahl der positiven Fälle (*n*) sortiert und im Makro \mytable gespeichert. Um den einzelnen Bundesländern einen numerischen Wert zuzuordnen zu können, wird mit

```
\pgfplotstableset{create on use/new/.style =
  {create col/expr={\pgfplotstablerow+1}}}
```

eine Spalte hinzugefügt, die als Wert einfach die Zeilenzahl erhält, sodass Bremen dann intern die 1 und Bayern intern die 16 zugeordnet wird. Die Tabelle sieht dann wie folgt aus:

```
\usepackage{pgfplotstable, booktabs}

\pgfplotstableread[col sep=comma]{table.csv}\myTable % Tabelle einlesen
\pgfplotstablesort[sort key={n}]{\mytable}{\myTable} % Tabelle sortieren
\pgfplotstableset{create on use/new/.style =
  {create col/expr={\pgfplotstablerow+1}}}
```

```
\pgfplotstabletypeset[
  use comma, col sep=comma,
  columns={new, n, Tote, Bundesland},
  columns/new/.style={column type=r,column name={\emph{n}}, dec sep align},
  columns/n/.style={column type=S,column name={\emph{Fälle}}, dec sep align},
  columns/Tote/.style={column type=S, column name={\emph{Tote}}, dec sep align},
  columns/Bundesland/.style={column type=l, string type, column name={\emph{
    ↵Bundesland}}},
  every head row/.style={
    before row={\toprule}, % have a rule at top
    after row={\midrule}
  },
  every last row/.style={after row=\bottomrule}, % rule at bottom
]{\mytable} % full path and filename
```

<i>n</i>	<i>Fälle</i>	<i>Tote</i>	<i>Bundesland</i>
1	480	15	Bremen
2	605	11	Mecklenburg-Vorpommern
3	1.166	21	Sachsen-Anhalt
4	1.464	25	Thüringen
5	1.857	38	Brandenburg
6	2.058	43	Saarland
7	2.118	41	Schleswig-Holstein
8	3.600	67	Sachsen
9	3.742	56	Hamburg
10	4.567	50	Berlin
11	4.734	64	Rheinland-Pfalz
12	5.859	123	Hessen
13	7.602	189	Niedersachsen
14	24.078	649	Baden-Württemberg
15	24.267	495	Nordrhein-Westfalen
16	32.282	786	Bayern

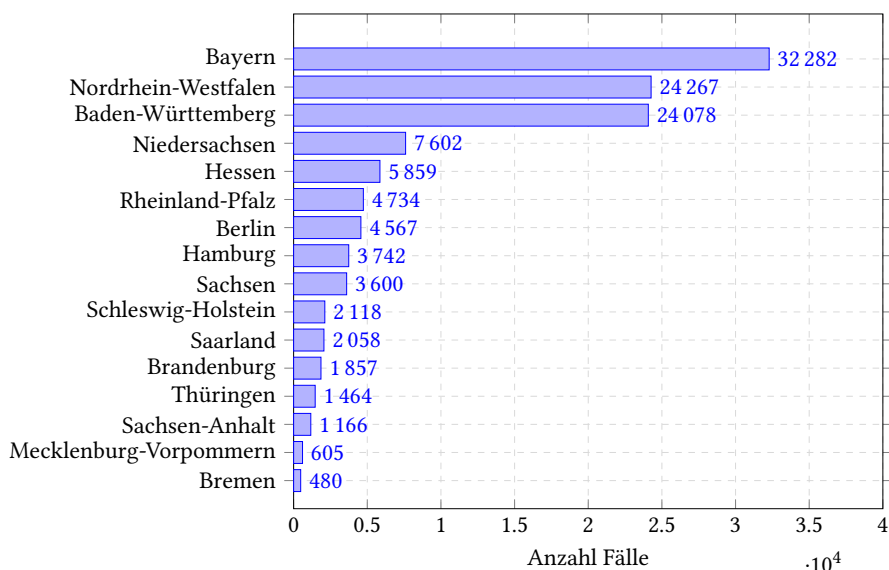
Somit kann die Spalte der Zeilennummern für die Werte der y -Achse benutzt werden. Statt dieser internen Nummern wird aber der zugehörige Name des Bundeslandes ausgegeben.

```
\usepackage{tikz, pgfplotstable, booktabs, pgfplots} \pgfplotsset{compat=newest}

\pgfplotstableread[col sep=comma]{table.csv}\myTable % Tabelle einlesen
\pgfplotstablesort[sort key={n}]{\mytable}{\myTable} % Tabelle sortieren
\pgfplotstableset{create on use/new/.style =
```

```
{create col/expr={\pgfplotstablerow+1}}

\begin{tikzpicture}
\begin{axis}[
  xbar, xmin=0, xmax=40000, ymin=0, width=0.75\linewidth,
  grid=major, grid style={dashed,gray!30},
  xlabel=Anzahl Fälle, ylabel=,
  /pgf/number format/1000 sep={\,}, /pgf/number format/fixed,
  ytick=data, yticklabels from table={\mytable}{Bundesland},
  nodes near coords, nodes near coords align={horizontal}]
\addplot table[x=n,y=new]{\mytable};
\end{axis}
\end{tikzpicture}
```



Eine Ausgabe, sortiert nach den Todeszahlen, lässt sich dann einfach daraus ableiten:

```
\usepackage{tikz, pgfplotstable, booktabs, pgfplots} \pgfplotsset{compat=newest}
```

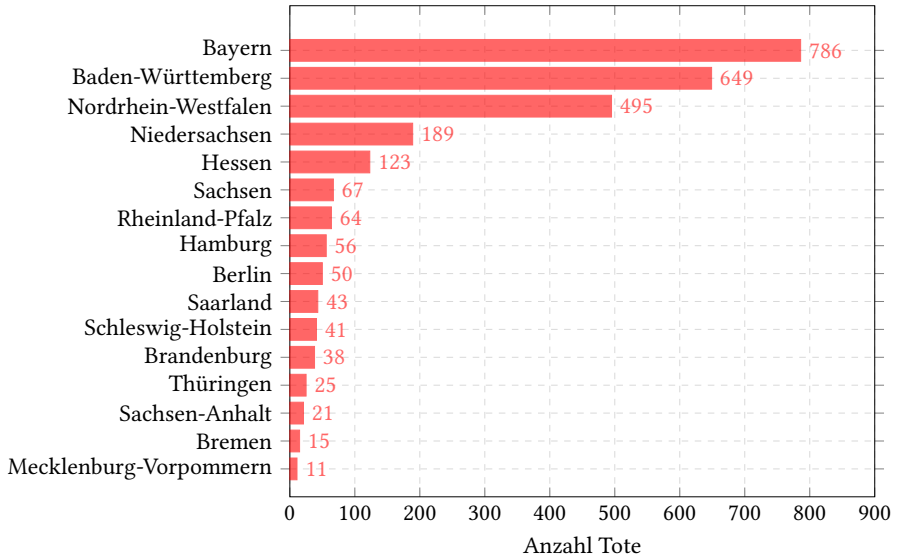
```
\pgfplotstableread[col sep=comma]{table.csv}\myTable % Tabelle einlesen
\pgfplotstablesort[sort key={Tote}]{\mytable}{\myTable} % Tabelle sortieren
\pgfplotstableset{create on use/new/.style =
  {create col/expr={\pgfplotstablerow+1}}}

\begin{tikzpicture}
\begin{axis}[
```

```

xbar, xmin=0, xmax=900, ymin=0, width=0.75\linewidth,
grid=major, grid style={dashed,gray!30},
xlabel=Anzahl Tote, ylabel=,
/pgf/number format/1000 sep={\,}, /pgf/number format/fixed,
ytick=data, yticklabels from table={\mytable}{Bundesland},
nodes near coords, nodes near coords align={horizontal}}
\addplot[red, fill, opacity=0.6] table[x=Tote,y=new]{\mytable};
\end{axis}
\end{tikzpicture}

```



Von fremden Bühnen

Neue Pakete auf CTAN

Jürgen Fenn

Der Beitrag stellt neue Pakete auf CTAN seit der letzten Ausgabe bis zum Redaktionsschluss in umgekehrter chronologischer Reihenfolge vor. Bloße Updates können auf der moderierten *CTAN-ann*-Mailingliste verfolgt werden.

musical von *Dave Howell* ist eine Klasse zum Setzen von Musicals und stellt Features insbesondere für das Einstudieren während der Probenzeit bereit.

CTAN:macros/latex/contrib/musical

media4svg von *Alexander Grahn* erlaubt es, Video- und Audio-Inhalte mittels SVG einzubetten.

CTAN:macros/latex/contrib/media4svg/

expkv-cs von *Jonathan P. Spratte* dient zum Definieren von Key-Value-Paaren gemeinsam mit dem Paket *expkv* desselben Autors.

CTAN:macros/generic/expkv-cs

annee-scolaire von *Yvon Henel* stellt einige Befehle bereit, um das »Akademische Jahr« bzw. das »Schuljahr« semantisch auszuzeichnen und in einem Dokument (mit Datum) auszugeben.

CTAN:macros/latex/contrib/annee-scolaire

git-latexdiff von *Matthieu Moy* ist ein weiteres Paket, mit dem man Git und *latexdiff* zusammen verwenden kann.

CTAN:support/git-latexdiff

tikz-planets von *Isabelle M. Santos* stellt Makros zum Zeichnen von Planeten bereit, mit denen die Autorin ursprünglich Materialien für einen Astronomiekurs illustriert hatte.

CTAN:graphics/pgf/contrib/tikz-planets

lua-uca von *Michal Hoftich* stellt eine grundlegende Unterstützung für den Unicode Collation Algorithm in Lua bereit.

CTAN:support/lua/lua-uca

yquant von *Benjamin Desef* ist ein Paket, mit dem man Quantenschaltungen (für Quantencomputer) mittels pgf/TikZ zeichnen kann.

CTAN:graphics/pgf/contrib/yquant

rest-api von *Dennis Klein* dient zum Beschreiben einer Rest-API.

CTAN:macros/latex/contrib/rest-api

expose-expl3-dunkerque-2019 von *Yvon Henel* enthält einen Vortrag in Aufsatzform über das Programmieren in expl3, der im Juni 2019 bei einer Tagung in Dunkerque auf Französisch gehalten wurde.

CTAN:info/expose-expl3-dunkerque-2019

shortmathj von *Nikolai Avdeev* stellt Abkürzungen für die Namen mathematischer Zeitschriften bereit.

CTAN:macros/latex/contrib/shortmathj

xepersian-hm von *Hossein Movahhedian* behebt einen Fehler in dem Kashida-Feature in dem Paket xepersian von *Vafa Khalighi*.

CTAN:macros/xetex/latex/xepersian-hm

hitszbeamer von *Jingxuan Yang* ist ein Beamer-Thema für Präsentationen am *Harbin Institute of Technology*.

CTAN:macros/latex/contrib/beamer-contrib/themes/hitszbeamer

biblatex-german-legal von *Dominik Brodowski* ist ein neuer BibL^AT_EX-Zitierstil für deutsche rechtswissenschaftliche Texte. Das Paket kann (und soll) um weitere Stile ergänzt werden.

CTAN:macros/latex/contrib/biblatex-contrib/biblatex-german-legal

lua-ul von *Marcel Krüger* ermöglicht es, Text mit LuaT_EX zu unterstreichen – und verträgt sich dabei ganz formidabel mit anderen Features.

CTAN:macros/luatex/latex/lua-ul

oops von *Erwann Rogard* erlaubt es, mathematische Definitionen *inline* in einen Fließtext einzufügen.

CTAN:macros/latex/contrib/oops

langsci-avm von *Felix Kopecky* dient zum Erstellen von Attribut-Wert-Matrizen und von Merkmalstrukturen für linguistische Arbeiten.

CTAN:macros/latex/contrib/langsci-avm

biblatex-ajc2020unofficial von *Nikolai Avdeev* ist eine Klasse für Aufsätze in der Zeitschrift *Australasian Journal of Combinatorics*, einschließlich eines BibL^AT_EX-Stils.

CTAN:macros/latex/contrib/biblatex-contrib/biblatex-ajc2020unofficial

biblatex2bibitem von *Nikolai Avdeev* gibt ein mit BibL^AT_EX erstelltes Literaturverzeichnis als bibitem-Umgebung aus.

CTAN:macros/latex/contrib/biblatex2bibitem

expkv-def von *Jonathan P. Spratte* stellt eine Schnittstelle zum Definieren von Schlüsselpaaren für das Paket `expkv` desselben Autors bereit.

CTAN:macros/generic/expkv-def

autofancyhdr von *Mojtaba Baghban* berechnet automatisch die Kopfbreite für das Paket `fancyhdr` von *Piet van Oostrum*.

CTAN:macros/latex/contrib/autofancyhdr

twemoji-colr von *Xiangdong Zeng* ist eine OpenType-Schrift mit farbigen Emojis aus der Twemoji-Sammlung.

CTAN:fonts/twemoji-colr

noto-emoji von *Xiangdong Zeng* enthält die Schriftart Noto Emoji mit allen Emojis, die in der jüngsten Version von Unicode enthalten sind.

CTAN:fonts/noto-emoji

emoji von *Xiangdong Zeng* stellt eine Unterstützung zur Verwendung von Emojis in Lua $\TeX$ bereit.

CTAN:macros/luatex/latex/emoji

fontsize von *Ivan Valbusa* erlaubt es, die Hauptschriftart eines Dokuments in einer beliebigen Größe und in einer beliebigen Maßeinheit global vorzugeben.

CTAN:macros/latex/contrib/fontsize

etbb von *Michael Sharpe* ist eine Erweiterung von *Daniel Benjamin Millers* Schriftsippe XETBook, die wiederum eine Erweiterung der Schrift ETBook darstellt, in diversen Fontkodierungen.

CTAN:fonts/etbb

schulmathematik von *Keno Wehr* ist ein Bundle, mit dem Mathematik- und Physiklehrer Arbeitsblätter, Bildschirmpräsentationen, Klausuren, Kompetenzlisten, Abiturgutachten und mündlichen Abiturprüfungsaufgaben setzen können.

CTAN:macros/latex/contrib/schulmathematik

haranoaji-extra von *Masamichi Hosoda* enthält eine erweiterte Sammlung der beiden japanischen Schriften Harano Aji Mincho und Harano Aji Gothic.

CTAN:fonts/haranoaji-extra

haranoaji von *Masamichi Hosoda* enthält die beiden japanischen Schriften Harano Aji Mincho und Harano Aji Gothic mit jeweils sieben Schnitten.

CTAN:fonts/haranoaji

texlive-dummy-enterprise-linux-8 von *Bob Tennent* enthält ein »Dummie-RMP«, das es erlaubt, Pakete, die eine $\TeX$ -Distribution voraussetzen, unter RHEL 8 zu installieren, ohne dass das Standard- $\TeX$ -Paket des Linux-Distributors vorliegt.

CTAN:support/texlive/texlive-dummy/EnterpriseLinux-8

hitszthesis von *Jingxuan Yang* ist eine Dokumentenklasse für Bachelorarbeiten am *Harbin Institute of Technology*.

CTAN:macros/latex/contrib/hitszthesis

fewerfloatpages von *Frank Mittelbach* verbessert die Ausgabe von Gleitobjekten, indem es halbleere Float-Seiten vermeidet.

CTAN:macros/latex/contrib/fewerfloatpages

optex von *Petr Olšák* ist eine Kombination aus Lua $\TeX$ und erweiterten Plain $\TeX$ -Makros (OPmac) in einem sehr frühen Versuchsstadium (»maybe working«).

CTAN:macros/luatex/optex

brandeis-thesis von *John Burt* ist eine Klasse für Abschlussarbeiten der *Brandeis University Graduate School of Arts and Sciences*.

CTAN:macros/latex/contrib/brandeis-thesis

leiletter von *Michiel Kreutzer* ist eine Briefklasse für die niederländische Universität Leiden.

CTAN:macros/latex/contrib/leiletter

euclideangeometry von *Claudio Beccari* ist eine Erweiterung des Pakets *curve2e* desselben Autors, mit dem man geometrische Konstruktionen erstellen kann.

CTAN:macros/latex/contrib/euclideangeometry

metastr von *Niklas Beisert* dient dazu, beliebige Zeichenfolgen (Strings) auf strukturierte Art zu definieren und zur späteren Verwendung bereitzustellen.

CTAN:macros/latex/contrib/metastr

lie-hasse von *Benjamin McKay* ist ein Paket, mit dem man Hasse-Diagramme mit *pgf/TikZ* zeichnen kann.

CTAN:graphics/pgf/contrib/lie-hasse

mpfonts von *Daniel Benjamin Miller* ist eine Type 3-Ausgabe der Computer Modern Fonts, die ein kräftigeres Schriftbild aufweisen als die sonst übliche Type-1-Version.

CTAN:fonts/mpfonts

physunits von *Brian W. Mulligan* definiert Abkürzungen für die wichtigsten physikalischen Maßeinheiten.

CTAN:macros/latex/contrib/physunits

physconst von *Brian W. Mulligan* stellt Makros zur Verwendung der wichtigsten physikalischen Konstanten zur Verfügung.

CTAN:macros/latex/contrib/physconst

expkv von *Jonathan P. Spratte* ist ein weiteres Paket, das ein Key-Value-Schnittstelle bereitstellt.

CTAN:macros/generic/expkv

pmhanguljamo von *Kangsoo Kim* bietet eine vereinfachte Methode für die Eingabe transliterierter koreanischer Texte für den Textsatz mit Hangul-Lettern.

CTAN: language/korean/pmhanguljamo

cmupint von *Uroš Stefanović* enthält verschiedene aufrechte Integralzeichen für die Standardschriftart Computer Modern. Sie ersetzen das im Englischen übliche schräggestellte Integralzeichen, beispielsweise in deutschen oder serbischen Texten.

CTAN: fonts/cmupint

wasy-type1 von *Michael Sharpe* enthält eine Type-1-Version der Schriftart wasy von *Roland Waldi*.

CTAN: fonts/wasy-type1

thorshammer von *Donald P. Story* ist eine Erweiterung der Möglichkeiten für Fragebögen, die mit dem Paket *exerquiz* als PDF-Datei gestaltet werden können. Auswertungen können zur weiteren Verarbeitung exportiert werden. Benötigt Adobe Acrobat DC oder XI.

CTAN: macros/latex/contrib/thorshammer

tikz-trackschematic von *Martin Scheidt* ist eine TikZ-Bibliothek zum Zeichnen von technischen Schienennetzplänen.

CTAN: graphics/pgf/contrib/tikz-trackschematic

bearwear von *Ulrike Fischer* stellt Befehle bereit, um Zeichnungen von kleinen Teddybären, die mit pgf/TikZ und dem Paket *tikzlings-bears* erstellt worden sind, mit »Kleidung« zu versehen.

CTAN: macros/latex/contrib/bearwear

texplate von *Paulo Roberto Massa Cereda* ist ein Tool, das zu L^AT_EX-Paketen Dokumentvorlagen erzeugen kann.

CTAN: support/texplate

hep-paper von *Jan Hajer* ist ein Paket zum Setzen von Arbeiten aus der Hochenergiephysik, das weitgehend auf bereits bekannte Drittlösungen zurückgreift.

CTAN: macros/latex/contrib/hep-paper

simplebnf von *Jay Lee* ist ein kleines Paket, mit dem man Grammatiken in der Backus-Naur-Form darstellen kann.

CTAN: macros/latex/contrib/simplebnf

verifica von *Riccardo Dossena* ist ein Paket, mit dem Lehrer die typischen Aufgaben für die Abschlussprüfung an italienischen Gymnasien setzen können.

CTAN: macros/latex/contrib/verifica

secnum von *Gau Syu* enthält Makros, mit denen man die Nummerierung der Gliederung eines Dokuments anpassen kann (große und kleine römische Zahlen

und Buchstaben sowie arabische Ziffern und Trennzeichen).

CTAN:macros/latex/contrib/secnum

scholax von *Michael Sharpe* ist eine Erweiterung der Schriftart TeXGyreSchola, einem Klon von New Century Schoolbook, einschließlich der Zahlen und der Zeichen für den Mathematiksat.

CTAN:fonts/scholax

biblatex-jura2 von *Christoph* stellt den zurzeit einzigen funktionierenden BibL^AT_EX-Stil für die die Formatierung von Fundstellen in rechtswissenschaftlichen Arbeiten bereit.

CTAN:macros/latex/contrib/biblatex-contrib/biblatex-jura2

pmdb von *Donald P. Story* dient zum Erstellen von PDF-Dokumenten mit der *poor man's database* (PMDb), aus denen man Inhalte leicht übernehmen kann.

CTAN:macros/latex/contrib/pmdb

qualitytype von *Daniel Benjamin Miller* enthält 45 Schriftarten, die von der Firma *QualiType* unter einer freien Lizenz veröffentlicht worden sind.

CTAN:fonts/qualitytype

apa7 von *Daniel A. Weiss* ist eine Weiterentwicklung des Pakets apa6 von *Brian D. Beitzel* zum Setzen von Texten nach den Vorgaben der siebten Auflage des *American Psychological Association Publication Manual*.

CTAN:macros/latex/contrib/apa7

clara von *Daniel Benjamin Miller* enthält eine relativ gut ausgebaute Serifenschrift, die im Jahr 2015 von *Séamas Ó Brógáin* zum Setzen des Buchs *A Dictionary of Editing* entworfen wurde.

CTAN:fonts/clara

xecyrmongolian von *Apostolos Syropoulos* stellt eine grundlegende Unterstützung zum Setzen von mongolischen Texten in kyrillischer Schrift für X_YL^AT_EX und LuaL^AT_EX bereit.

CTAN:language/mongolian/xecyrmongolian

sdaps von *Benjamin Berg* ist die L^AT_EX-Unterstützung für das Programm SDAPS, mit dem man schriftliche Fragebogen erstellen kann, die automatisiert ausgewertet werden können.

CTAN:macros/latex/contrib/sdaps

parsa von *Farshad Rasuli* ist eine Klasse für Abschlussarbeiten an iranischen Universitäten für X_YL^AT_EX.

CTAN:macros/xetex/latex/parsa

domitian von *Daniel Benjamin Miller* ist ein freier Ersatz für die Schriftart Palatino.

CTAN:fonts/domitian

texlab von Eric Förster enthält eine plattformübergreifende Implementierung des *Language Server Protocol* von Microsoft und kann mit jedem Texteditor verwendet werden, der das Protokoll umsetzt. Die Dokumentation findet man nicht auf CTAN, sondern unter <https://texlab.netlify.com/docs>.

CTAN:support/texlab

erewhon-math von Daniel Flipo ist ein Mathematikfont samt der dazugehörigen L^AT_EX-Unterstützung.

CTAN:fonts/erewhon-math

barracuda von Roberto Giacomelli ist eine Bibliothek, die die Generierung von Strichcodes mit Hilfe von Lua ermöglicht. Das Modul kann auch unabhängig von Lua_T_EX verwendet werden.

CTAN:macros/luatex/generic/barracuda

hvqrurl von Herbert Voß setzt einen QR-Code in den Seitenrand eines Texts.

CTAN:macros/latex/contrib/hvqrurl

pinoutikz von Robert Blazek dient dazu, Chip-Pinouts (das sind die Anschlussbelegungen von elektronischen Bausteinen) mit Hilfe von pgf/TikZ zu zeichnen.

CTAN:graphics/pinoutikz

fontsetup von Antonis Tzolomitis ist ein Frontend zu dem Paket fontspec für das Setup einzelner Mathematikschriften.

CTAN:macros/latex/contrib/fontsetup

newcomputermodern von Antonis Tzolomitis ist eine neue Sammlung der bekannten CM-Fonts, die auch nicht-lateinische Zeichen enthält. Das Ziel ist, schließlich sämtliche Sprachen zu umfassen.

CTAN:fonts/newcomputermodern

letterswitharrows von Maximilian Teegen zeichnet Pfeile über einzelne Buchstaben im Mathematikmodus.

CTAN:macros/latex/contrib/letterswitharrows

chemplants von Elia Arnese Feffin stellt Befehle bereit, um Chemie-Anlagen mit pgf/TikZ zu zeichnen.

CTAN:graphics/pgf/contrib/chemplants

lexend von Yannick Schinko stellt die Schriftart Lexend für X_YL^AT_EX und Lua_T_EX bereit.

CTAN:fonts/lexend

gfsdidotclassic von Antonis Tzolomitis stellt die klassische Version der griechischen Schriftart GFS Didot Classic für Unicode-_T_EX-Engines bereit.

CTAN:fonts/greek/gfs/gfsdidotclassic

algxpar von *Jander Moreira* erweitert das Paket *algorithmicx* von *Szász János*, so dass auch mehrzeiliger Text verarbeitet werden kann.

CTAN:macros/latex/contrib/algxpar

latino-sine-flexione von *Victore Leve* erlaubt die Verwendung der gleichnamigen Plansprache von *Giuseppe Peano* mit L^AT_EX. Auch ein Blindtext wird bereitgestellt.

CTAN:macros/latex/contrib/latino-sine-flexione

circledsteps von *Romano Giannetti* stellt Befehle bereit, um Zahlen in einem Kreis zu setzen, die man beispielsweise verwenden kann, um die Schritte in einer Anleitung zu nummerieren.

CTAN:macros/latex/contrib/circledsteps

bxjatoucs von *Takayuki Yato* konvertiert japanische Kodierungen nach Unicode.

CTAN:language/japanese/BX/bxjatoucs

xkcdcolors von *Romano Giannetti* erschließt die Namen, die im Jahr 2010 bei einer spaßigen Umfrage des Web-Comics *xkcd* herausgekommen waren, für die Verwendung mit L^AT_EX.

CTAN:macros/latex/contrib/xkcdcolors

Spielplan

2020

24. 7. – 26. 7. **TUG 2020**

Rochester Institut of Technology (RIT)
Rochester, New York, USA
<https://www.tug.org/tug2020/>



22. 8. – 23. 8. **FrOSCon**, Free and Open Source Conference

Hochschule Bonn-Rhein-Sieg
Grantham-Allee 20, 53757 Sankt Augustin
<https://www.froscon.de/startseite/>



6. 9. – 12. 9. **14th International ConT_EXt Meeting**

Prag–Sibřina, Tschechische Republik
<https://meeting.contextgarden.net/>



18. 9. – 19. 9. **18. Kieler Open Source und Linux Tage**

Kitz – Kieler Innovations- und Technologiezentrum
Schauenburgerstr. 116
24118 Kiel
<https://www.kieler-linuxtage.de>



2021

10. 3. – 12. 3. **DANTE 2021**

und 64. Mitgliederversammlung von DANTE e.V.
Otto-von-Guericke-Universität
39016 Magdeburg
<https://www.dante.de/veranstaltungen/dante2021/>



Stammtische



In verschiedenen Städten im Einzugsbereich von DANTE e.V. finden regelmäßig Treffen von T_EX-Anwendern statt, die für jeden offen sind. Im Web gibt es aktuelle Informationen unter <https://projekte.dante.de/Stammtische/WebHome>.

Aachen

Torsten Bronger,
bronger@physik.rwth-aachen.de
Restaurant Galilei, Marienbongard 24, 52062 Aachen
Erster Donnerstag im Monat, 20.00 Uhr

Berlin

Michael-E. Voges, Tel.: (03362) 50 18 35,
mevoges@t-online.de
Mantee – Café Restaurant, Chausseestraße 131, 10115 Berlin
Zweiter Donnerstag im Monat, 19.00 Uhr

Erlangen

Walter Schmidt, Peter Seitz,
w.a.schmidt@gmx.net
Gaststätte »Deutsches Haus«, Luitpoldstraße 25, 91052 Erlangen
Dritter Dienstag im Monat, 19.00 Uhr



Göttingen

Holger Nobach,
holger.nobach@nambis.de, <http://goetex.nambis.de/>
Restaurant Mazzoni Cucina Italiana,
Hermann-Rein-Straße 2, 37075 Göttingen
Dritter Donnerstag im Monat, 18.00 Uhr

Hamburg

Günther Zander,
guenther.zander@lug-balista.de
Bürgerhaus in Barmbek, Lorichsstraße 28a, 22307 Hamburg
Zweiter Montag im Monat, 20.00 Uhr

Hannover

Reik Kaps,
kaps@luis.uni-hannover.de
Raum B024 im EG des LUIS,
Schloßwender Str. 5, 30419 Hannover
Zweiter Donnerstag im Monat, 18.30 Uhr

Heidelberg

Martin Wilhelm Leidig, Tel.: 0170 418 33 29,
moss@moss.in-berlin.de
Anmeldeseite zur Mailingliste: <https://tinyurl.com/stammtisch-HD>
Wechselnder Ort
Letzter Freitag im Monat, ab 19.30 Uhr



Köln

Uwe Ziegenhagen
Dingfabrik, Erzbergerplatz 9, 50733 Köln
zur Zeit inaktiv, 19.00 Uhr

Stuttgart

Bernd Raichle,

bernd.raichle@gmx.de

»Trollinger-Stubn«, Rotebühlstr. 50, 70178 Stuttgart

Zweiter Dienstag im Monat, 19.30 Uhr

Trier

Martin Sievers,

tex-stammtisch-trier@schoenerpublizieren.de

Universität Trier

nach Vereinbarung

Wuppertal

Andreas Schrell, Tel.: (02193) 53 10 93,

as@schrell.de

Restaurant Croatia »Haus Johannisberg«, Südstraße 10, 42103 Wuppertal

Zweiter Donnerstag im Monat, 19.30 Uhr

Adressen

DANTE, Deutschsprachige Anwendervereinigung T_EX e.V.

Postfach 11 03 61

69072 Heidelberg

Tel.: (0 62 21) 2 97 66 (Mo., Mi., Do. von 10.00–12.00 Uhr)

Fax: (0 62 21) 16 79 06

E-Mail: info@dante.de

Konto: VR Bank Rhein-Neckar eG

IBAN DE67 6709 0000 0002 3100 07 SWIFT-BIC GENODE61MA2

Vorstand

Vorsitzender:	Martin Sievers	president@dante.de
stv. Vorsitzender:	Herbert Voß	vice-president@dante.de
Schatzmeisterin:	Doris Behrendt	treasurer@dante.de
Schriftführer:	Manfred Lotz	secretary@dante.de
Beisitzer:	Klaus Höppner	
	Harald König	
	Volker RW Schaa	
	Uwe Ziegenhagen	

Ehrenmitglieder

Peter Sandner	22.03.1990	Klaus Thull († 2012)	22.03.1990
Yannis Haralambous	05.09.1991	Barbara Beeton	27.02.1997
Luzia Dietsche	27.02.1997	Donald E. Knuth	27.02.1997
Eberhard Mattes	27.02.1997	Hermann Zapf († 2015)	19.02.1999
Joachim Lammarsch	12.04.2014	Rainer Schöpf	12.04.2014

Webserver und Mailingliste

DANTE: <https://www.dante.de/> (Erik Braun)
CTAN: <https://mirror.ctan.org/> (Gerd Neugebauer)
DANTE-EV: <https://lists.dante.de/mailman/listinfo/dante-ev>

FAQ

DTK: <https://projekte.dante.de/DTK/WebHome>

T_EX: <https://projekte.dante.de/DanteFAQ/WebHome>

T_EXnische Fragen

beraterkreis@dante.de

ak-schule@dante.de

Autoren/Organisatoren

Wolfgang Beinert

wolfgang@beinert.net

[17]

Jürgen Fenn

Christian-Stock-Straße 4

63263 Neu-Isenburg

juergen.fenn@gmx.de

[55]

Lukas C. Bossert

RWTH Aachen University

IT Center

Seffenter Weg 23

52074 Aachen

bossert@itc.rwth-aachen.de

[24,36]

Martin Sievers

siehe Seite 66

[4]

Herbert Voß

Wasgenstraße 21

14129 Berlin

herbert@dante.de

[3,6,33,49]

Die T_EXnische Komödie

32. Jahrgang Heft 2/2020 Mai 2020

Impressum

Editorial

Hinter der Bühne

4 Grußwort

Bretter, die die Welt bedeuten

- 6 Die Eingabe von Sonderzeichen
- 17 Schriftwahl
- 24 Mit »biber --tool« Bibliografieinträge bearbeiten
- 33 Eine weitere Schrift für Menschen mit Leseschwäche
- 36 Kommentieren und Dokumentieren von Code
- 49 Kommaseparierte Listen als Tabellen und Grafiken darstellen

Von fremden Bühnen

55 Neue Pakete auf CTAN

Spielplan

- 63 Termine
- 64 Stammtische

Adressen

67 Autoren/Organisatoren