

Typst – Hat L^AT_EX abgedankt?



Eine kurze Einführung in Typst

Tristan Pieper tristan.pieper@uni-rostock.de · 1. Juni 2023

Inhaltsverzeichnis

1. Kurzes Kennenlernen	3
2. Probleme von L ^A T _E X.....	4
3. Die Lösung aller Probleme(?)	12
4. Die Web-App	14
5. Grundlegende Formatierung	16
6. Die Typst-Dokumentation...	30
7. Eigene Templates und Skripts	32
8. Typst kann noch mehr!	37
9. Abschluss und Weiteres	41

1. Kurzes Kennenlernen

2. Probleme von L^AT_EX

2.1. Alles begann mit...



Donald E. Knuth¹ (geb. 10. Januar 1938)

2.2. Dann kam...



Leslie Lamport¹ (geb. 7. Februar 1941)

2.3. Die Probleme

1. Riesige Programmgröße
2. Auswahl an Compilern
3. Unverständliche Fehler

2.4. Größe des Programms

```
% du -sch /usr/share/texmf-dist/* | sort -hr
2,5G    insgesamt
1,9G    /usr/share/texmf-dist/fonts
499M    /usr/share/texmf-dist/tex
58M     /usr/share/texmf-dist/scripts
44M     /usr/share/texmf-dist/tex4ht
24M     /usr/share/texmf-dist/bibtex
15M     /usr/share/texmf-dist/metapost
7,5M    /usr/share/texmf-dist/dvips
3,8M    /usr/share/texmf-dist/xindy
3,4M    /usr/share/texmf-dist/ls-R
2,6M    /usr/share/texmf-dist/asymptote
1,7M    /usr/share/texmf-dist/context
516K    /usr/share/texmf-dist/omega
344K    /usr/share/texmf-dist/makeindex
```

Verglichen mit 21MB des Typst-Compilers...

```
% du -sch /usr/bin/typst
21M    /usr/bin/typst
21M    insgesamt
```

2.5. Die Vielfalt

„ \LaTeX “ ist kein Programm, sondern:

- pdfTeX
- LuaTeX
- XeTeX
- MikTeX
- KaTeX
- ...

2.6. Beispiel-Fehlermeldung (Typst)

Typst:

```
$  
+ Dies  
+ Ist  
+ Eine  
+ Liste!
```

```
error: expected dollar sign  
└ test.typ:5:8  
5 | + Liste!  
   ^
```

2.7. Beispiel-Fehlermeldung (L^AT_EX)

L^AT_EX:

```
\documentclass{article}

\begin{document}

\$

\begin{enumerate}
\item Dies
\item Ist
\item Eine
\item Liste!
\end{enumerate}

\end{document}
```

```
Latexmk: This is Latexmk, John Collins, 17 Mar. 2022. Version 4.77,
version: 4.77.
Latexmk: applying rule 'pdflatex'...
Rule 'pdflatex': File changes, etc:
    Changed files, or newly in use since previous run(s):
    /path/Desktop/Projekte/Typst/typst-seminar/.lt/test.tex
    test.tex
Rule 'pdflatex': The following rules & subrules became out-of-date:
    pdflatex
-----
Run number 1 of rule 'pdflatex'
-----
-----
Running 'pdflatex -synctex=1 -interaction=nonstopmode -file-line-
error -recorder "/path/Desktop/Projekte/Typst/typst-seminar/.lt/
test.tex"'
-----
This is pdfTeX, Version 3.141592653-2.6-1.40.24 (TeX Live 2022/Arch
Linux) (preloaded format=pdflatex)
  restricted \write18 enabled.
entering extended mode
(/path/Desktop/Projekte/Typst/typst-seminar/.lt/test.tex
LaTeX2e <2021-11-15> patch level 1
L3 programming layer <2022-04-10> (/usr/share/texmf-dist/tex/latex/
base/article.cls
Document Class: article 2021/10/04 v1.4n Standard LaTeX document
class
(/usr/share/texmf-dist/tex/latex/base/size10.clo)) (/usr/share/
texmf-dist/tex/latex/l3backend/l3backend-pdftex.def) (./test.aux)
/path/Desktop/Projekte/Typst/typst-seminar/.lt/test.tex:5: Missing
$ inserted.
<inserted text>
                $
1.5

[1{/var/lib/texmf/fonts/map/pdftex/updmap/pdftex.map}] (./
test.aux) )
(see the transcript file for additional information)</usr/share/
texmf-dist/fonts/type1/public/amsfonts/cm/cmr10.pfb>
Output written on test.pdf (1 page, 13646 bytes).
SyncTeX written on test.synctex.gz.
Transcript written on test.log.
Latexmk: If appropriate, the -f option can be used to get latexmk
  to try to force complete processing.
Latexmk: Getting log file 'test.log'
Latexmk: Examining 'test.fls'
Latexmk: Examining 'test.log'
Latexmk: Log file says output to 'test.pdf'
Latexmk: Errors, so I did not complete making targets
Collected error summary (may duplicate other messages):
```

3. Die Lösung aller Probleme(?)

3.1. Ein kleiner Vergleich

LaTeX	Typst	Ergebnis
<pre>\documentclass{article} \begin{document} \begin{enumerate} \item Dies \item Ist \item Eine \item Liste! \end{enumerate} \end{document}</pre>	<ul style="list-style-type: none">+ Dies+ Ist+ Eine+ Liste!	<ol style="list-style-type: none">1. Dies2. Ist3. Eine4. Liste!

4. Die Web-App

4.1. Ab ans Werk!

Vorteile:

- alle Dateien online
- verschiedene Projekte erstellbar
- guter online Editor
- eingebaute Dokumentation

<https://typst.app/>

5. Grundlegende Formatierung

5.1. Überschriften

```
// Hier ein Kommentar,  
// er wird ignoriert!
```

= Überschrift 1!
== Überschrift 2!
== Überschrift 3!
Text

Neuer Abstand!

6. Überschrift 1!

6.1. Überschrift 2!

6.1.1. Überschrift 3!

Text

Neuer Absatz!

5.2. Listen

- + Eine nummerierte Liste
 - + Kann sehr schön sein!
 - 1. So geht sie auch!
 - 2. Jawohl!
 - Und hier nicht-nummeriert!
 - Ganz ohne Nummern!
- 1. Eine nummerierte Liste
 - 2. Kann sehr schön sein!
 - 1. So geht sie auch!
 - 2. Jawohl!
 - Und hier nicht-nummeriert!
 - Ganz ohne Nummern!

5.3. Schriftart

```
#text(font: "Arial", [Hallo Welt!])
```

Hallo Welt!

```
#text(font: "Courier New", [Hallo  
Welt!])
```

Hallo Welt!

```
#text(font: "New Computer Modern",  
[Hallo Welt!])
```

Hallo Welt!

5.4. Text¹

```
*Hallo!* #strong[Hallo!]
```

Hallo! Hallo!

```
_Hallo!_ #emph[Hallo!]
```

Hallo! Hallo!

```
Hallo!#super([Hallo!])
```

Hallo!^{Hallo!}

```
Hallo!#sub([Hallo!])
```

Hallo!_{Hallo!}

```
#text(fill: red, [Hallo rot!])
```

Hallo rot!

```
#text(fill: rgb("#ff00ff"), [Hallo  
pink!])
```

Hallo pink!

```
#text(fill: rgb("#ff00ff"),  
smallcaps(strong(emph[Hallo pink!])))
```

HALLO PINK!

5.5. Ausrichtung

```
#align(left, [Hallo!])      Hallo!  
#align(center, [Hallo!])    Hallo!  
#align(right, [Hallo!])     Hallo!
```

5.6. Abstände

Vertikaler
`#v(2cm)`
Abstand `#h(2cm)` Horizontaler

Vertikaler
Abstand Horizontaler

5.7. Bilder

```
#image(height: 50%, "leslie_lamport.png")
```



5.8. Tabellen

```
#table(  
  columns: (auto, 3cm, auto),  
  [Hallo1!],  
  [2a],  
  [Hallo3!],  
  [Welt1!],  
  [2b],  
  [Welt3!]  
)
```

Hallo1!	2a	Hallo3!
Welt1!	2b	Welt3!

5.9. Mathematik

```
$ sum_(k=0)^n k = 1 + ... + n $  
  
$ A = pi r^2 $  
  
$ "area" = pi dot.op "radius"^2 $  
  
$ cal(A) :=  
  { x in RR | x "is natural" } $  
  
$ frac(a^2, 2) $
```

$$\sum_{k=0}^n k = 1 + \dots + n$$

$$A = \pi r^2$$

$$\text{area} = \pi \cdot \text{radius}^2$$

$$\mathcal{A} := \{x \in \mathbb{R} \mid x \text{ is natural}\}$$

$$\frac{a^2}{2}$$

5.10. Set-Regeln¹

Hier ist noch die Standard-Schriftart!

```
#set text(font: "New Computer Modern", fill: blue)
```

Ab jetzt ist alles vollkommen in der anderen Schriftart und sogar blau!

```
#set par(first-line-indent: 1.5em, justify: true)
```

Ab jetzt wird jede erste Zeile eines Absatzes eingerückt und Blocksatz!

Wirklich, versprochen!
`#lorem(20)`

Hier ist noch die Standard-Schriftart!

Ab jetzt ist alles vollkommen in der anderen Schriftart und sogar blau!

Ab jetzt wird jede erste Zeile eines Absatzes eingerückt und Blocksatz!

Wirklich, versprochen! Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat.

5.11. Show-Regeln¹

```
#show heading: set text(navy)
```

==== Hallo!

===== Welt!

```
// aus dem offiziellem  
// Typst-Tutorial  
#show "Project": smallcaps  
#show "badly": "great"
```

We started Project in 2019
and are still working on it.
Project is progressing badly.

5.11.1. Hallo!

5.11.1.1. Welt!

We started PROJECT in
2019 and are still
working on it. PROJECT
is progressing great.

```

\documentclass[14pt,a4paper]{extarticle}
\usepackage{bold-extra}
\usepackage{amssymb}
\usepackage[T1]{fontenc}
\usepackage[left=2cm,right=2cm,top=2cm,bottom=2cm]
{geometry}

\setlength{\parskip}{0.65em}
\setlength{\parindent}{0pt}

\begin{document}
    \noindent\textbf{\textsc{Definition 1.}}\\
\textit{Sei  $D \subsetneq \mathbb{R}$  und sei  $f: D \rightarrow \mathbb{R}$  eine Funktion.  $f$  ist stetig in  $x_0 \in D$  genau dann, wenn die folgende Aussage gilt:}

    \textit{Für alle  $\epsilon > 0$  existiert ein  $\delta > 0$ , sodass  $|f(x) - f(x_0)| < \epsilon$  für alle  $x \in D$  mit  $|x - x_0| < \delta$ .}

    \textit{Oder Alternativ:  $\forall \epsilon > 0 \exists \delta > 0 \forall x \in D : |y - y_0| < \delta \Rightarrow |f(x) - f(x_0)| < \epsilon$ }

    \bigskip
    (\LaTeX)
\end{document}

```

```

#set page(margin: 2cm)
#set text(size: 14pt, font: "New Computer Modern")
#set par(justify: true)

*#smallcaps([Definition 1.])* _Sei  $D \subsetneq \mathbb{R}$  und sei  $f: D \rightarrow \mathbb{R}$  eine Funktion.  $f$  ist stetig in  $x_0 \in D$  genau dann, wenn die folgende Aussage gilt:_

_Für alle  $\epsilon > 0$  existiert ein  $\delta > 0$ , sodass  $|f(x) - f(x_0)| < \epsilon$  für alle  $x \in D$  mit  $|x - x_0| < \delta$ ._

_Oder Alternativ:  $\forall \epsilon > 0 \exists \delta > 0 \forall x \in D : |y - y_0| < \delta \Rightarrow |f(x) - f(x_0)| < \epsilon$ _

#v(1em)
(Typst)

```

DEFINITION 1. Sei $D \subseteq \mathbb{R}$ und sei $f : D \rightarrow \mathbb{R}$ eine Funktion. f ist stetig in $x_0 \in D$ genau dann, wenn die folgende Aussage gilt:

Für alle $\epsilon > 0$ existiert ein $\delta > 0$, sodass $|f(x) - f(x_0)| < \epsilon$ für alle $x \in D$ mit $|x - x_0| < \delta$.

Oder Alternativ: $\forall \epsilon > 0 \exists \delta > 0 \forall x \in D : |y - y_0| < \delta \Rightarrow |f(x) - f(x_0)| < \epsilon$

(LATEX)

DEFINITION 1. Sei $D \subseteq \mathbb{R}$ und sei $f : D \rightarrow \mathbb{R}$ eine Funktion. f ist stetig in $x_0 \in D$ genau dann, wenn die folgende Aussage gilt:

Für alle $\varepsilon > 0$ existiert ein $\delta > 0$, sodass $|f(x) - f(x_0)| < \varepsilon$ für alle $x \in D$ mit $|x - x_0| < \delta$.

Oder Alternativ: $\forall \varepsilon > 0 \exists \delta > 0 \forall x \in D : |y - y_0| < \delta \Rightarrow |f(x) - f(x_0)| < \varepsilon$

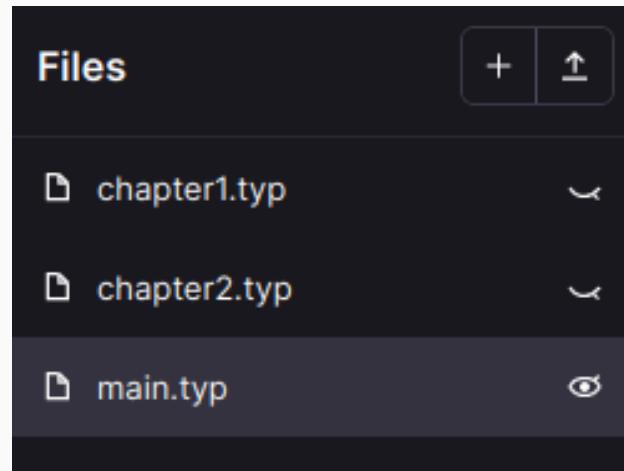
(Typst)

6. Die Typst-Dokumentation...

- <https://typst.apps/docs> als Nachschlagwerk
- Dokumentation ist wichtig!

7. Eigene Templates und Skripts

7.1. Mehrere Typst-Dateien verwenden



main.typ:

= Mein tolles Buch

Hier, Kapitel!

```
#include "chapter1.typ"
#include "chapter2.typ"
```

chapter1.typ:

== Kapitel 1

```
#lorem(10)
```

chapter2.typ:

== Kapitel 2

```
#lorem(10)
```

7.2. Eigene Funktionen und Variablen #1

```
#let var = 3.14159
#let double(e) = {
    return 2*e
}

$pi$ ist etwa #var!
$tau$ ist etwa #double(var)!

$pi approx var$ \
$tau approx double(var)$
```

π ist etwa 3.14159!
 τ ist etwa 6.28318!

$\pi \approx 3.14159$
 $\tau \approx 6.28318$

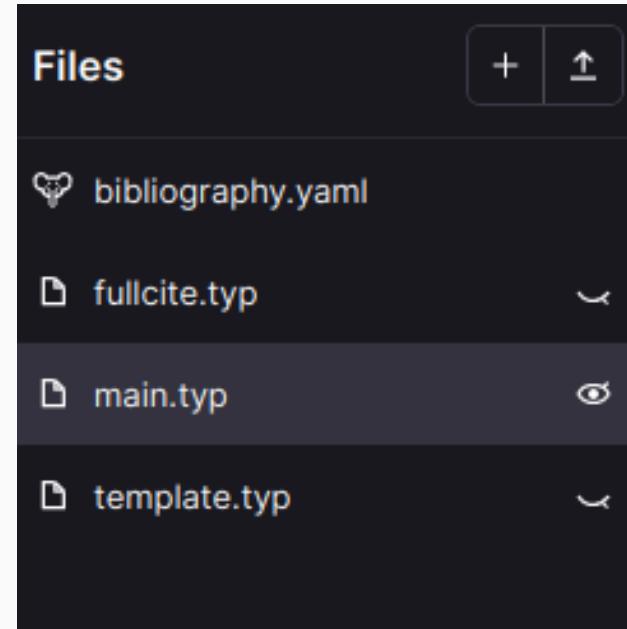
7.3. Eigene Funktionen und Variablen #2

```
#let names = ("Peter", "Petra", "Josef", "Josefa")
#let greet(names) = {
    [Hallo ]
    names.join(", ", last: " und ")
    [! #names.len() wundervolle Namen!]
}

#greet(names)
```

Hallo Peter, Petra, Josef und Josefa! 4 wundervolle Namen!

7.4. Funktionen auslagern



```
#import "fullcite.typ": *
#import "fullcite.typ": load_bibliography, fullcite
```

fullcite.typ definiert:

- load_bibliography
- print_bibliography
- fullcite
- ...

8. Typst kann noch mehr!

8.1. Figuren und Referenzen¹

@glacier shows a glacier. Glaciers are complex systems.

```
#figure(  
  image("glacier.jpg", height: 80%),  
  caption: [A curious figure.],  
) <glacier>
```

Figure 1 shows a glacier. Glaciers are complex systems.



Figure 1: A curious figure.

8.2. Bibliographie¹

works.bib:

```
@article{netwok,
  title={At-scale impact of the {Net Wok}: A culinarily holistic investigation of distributed dumplings},
  author={Astley, Rick and Morris, Linda},
  journal={Armenian Journal of Proceedings},
  volume={61},
  pages={192--219},
  year={2020},
  publisher={Automattic Inc.}

}

@article{arrgh,
  title={The Pirate Organization},
  author={Leeson, Peter T.},
}
```

This was already noted by pirates long ago. @arrgh

Multiple sources say ...
`#cite("arrgh", "netwok").`

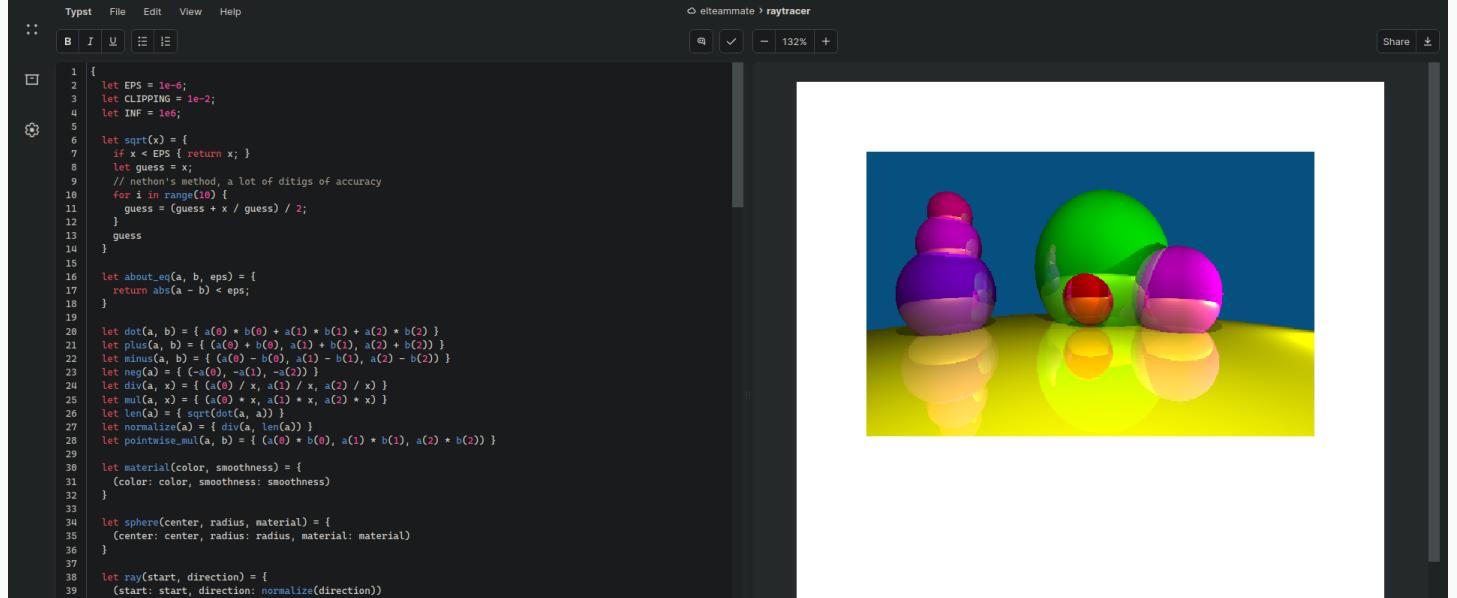
`#bibliography("works.bib")`

This was already noted by pirates long ago. [1]
Multiple sources say ... [1, 2].

Bibliography

- [1] P. T. Leeson, “The pirate organization.”
- [2] R. Astley, and L. Morris, “At-scale impact of the Net Wok: a culinarily holistic investigation of distributed dumplings,” *Armenian J. Proc.*, vol. 61, pp. 192–219, 2020.

8.3. Raytracing



The image shows a screenshot of a code editor and a rendered 3D scene. The code editor on the left displays a Typst script for a raytracer, with syntax highlighting for variables and functions. The rendered scene on the right shows several spheres of different sizes and colors (green, purple, red, yellow) on a reflective surface against a blue background.

```
1 {
2     let EPS = 1e-6;
3     let CLIPPING = 1e-2;
4     let INF = 1e6;
5
6     let sqrt(x) = {
7         if x < EPS { return x; }
8         let guess = x;
9         // newton's method, a lot of digits of accuracy
10        for i in range(10) {
11            guess = (guess + x / guess) / 2;
12        }
13        guess
14    }
15
16    let about_eq(a, b, eps) = {
17        return abs(a - b) < eps;
18    }
19
20    let dot(a, b) = { a(0) * b(0) + a(1) * b(1) + a(2) * b(2) }
21    let plus(a, b) = { (a(0) + b(0), a(1) + b(1), a(2) + b(2)) }
22    let minus(a, b) = { (a(0) - b(0), a(1) - b(1), a(2) - b(2)) }
23    let neg(a) = { (-a(0), -a(1), -a(2)) }
24    let div(a, x) = { (a(0) / x, a(1) / x, a(2) / x) }
25    let mul(a, x) = { (a(0) * x, a(1) * x, a(2) * x) }
26    let len(a) = { sqrt(dot(a, a)) }
27    let normalize(a) = { div(a, len(a)) }
28    let pointwise_mul(a, b) = { (a(0) * b(0), a(1) * b(1), a(2) * b(2)) }
29
30    let material(color, smoothness) = {
31        (color, smoothness: smoothness)
32    }
33
34    let sphere(center, radius, material) = {
35        (center: center, radius: radius, material: material)
36    }
37
38    let ray(start, direction) = [
39        (start: start, direction: normalize(direction))
40    ]
41
42    let intersect(ray, spheres) = {
43        let (start, direction) = ray;
44        let closest = null;
45        let closest_dist_sq = null;
46
47        for sphere in spheres {
48            let center = sphere.center;
49            let radius_sq = sphere.radius * sphere.radius;
50
51            let a = dot(direction, direction);
52            let b = 2 * dot(start - center, direction);
53            let c = dot(start - center, start - center) - radius_sq;
54
55            let discriminant = b * b - 4 * a * c;
56            if discriminant >= 0 {
57                let dist_sq = (-b - sqrt(discriminant)) / (4 * a);
58
59                if closest == null || dist_sq < closest_dist_sq {
60                    closest = dist_sq;
61                    closest_dist_sq = dist_sq;
62                }
63            }
64        }
65
66        if closest != null {
67            let t = sqrt(closest);
68            let hit_pos = start + direction * t;
69
70            return (hit_pos, closest);
71        }
72
73        null
74    }
75
76    let shade(hit_pos, hit_normal, spheres) = {
77        let (color, smoothness) = material;
78
79        if smoothness > 0 {
80            let eye_normal = normalize(hit_normal);
81            let light_dir = normalize((0, 1, 0));
82
83            let dot_product = dot(eye_normal, light_dir);
84            let diffuse_color = color * dot_product;
85
86            let reflectance = 1 - dot_product;
87            let reflectance_sq = reflectance * reflectance;
88
89            let reflect_pos = hit_pos + eye_normal * reflectance_sq;
90            let reflect_normal = normalize(reflect_pos - hit_pos);
91
92            let reflect_color = shade(hit_pos, reflect_normal, spheres);
93            let reflect_color_sq = reflect_color * reflect_color;
94
95            let final_color = diffuse_color + reflect_color_sq;
96
97            return final_color;
98        }
99
100       color
101    }
102
103    let render(spheres) = {
104        let rays = [];
105
106        for y in range(-10, 10) {
107            for x in range(-10, 10) {
108                let ray_start = (0, 0, 0);
109                let ray_direction = (x, y, 0);
110
111                let ray_end = intersect(ray, spheres);
112
113                if ray_end != null {
114                    let (hit_pos, closest_dist_sq) = ray_end;
115
116                    let hit_normal = normalize(hit_pos - center);
117
118                    let material_color = shade(hit_pos, hit_normal, spheres);
119
120                    let ray_end_color = material_color;
121
122                    rays.push(ray_end_color);
123                }
124            }
125        }
126
127        rays
128    }
129
130    let main() = {
131        let spheres = [
132            sphere((0, 0, 0), 1, material((0, 1, 0), 0.5)),
133            sphere((1, 0, 0), 0.5, material((1, 0, 0), 0.5)),
134            sphere((-1, 0, 0), 0.5, material((0, 0, 1), 0.5)),
135            sphere((0, 1, 0), 0.5, material((1, 0, 0), 0.5)),
136            sphere((0, -1, 0), 0.5, material((0, 1, 0), 0.5)),
137            sphere((0, 0, 1), 0.5, material((0, 0, 1), 0.5)),
138            sphere((0, 0, -1), 0.5, material((1, 0, 0), 0.5)),
139            sphere((0, 0, 0), 0.1, material((1, 0, 0), 100))
140        ];
141
142        render(spheres);
143    }
144
145    main();
146}
```

Voll funktionsfähiger Raytracer für 3D-Rendering.¹

9. Abschluss und Weiteres

9.1. Was noch fehlt¹

- Fußnoten
- Paketmanager (kurzfristige Alternative: GitHub)
- StackOverflow (kurzfristige Alternative: Discord)

9.2. Erwartete Neuerungen¹

- Fußnoten (und die komplette Überarbeitung der Layout-Engine)
- Paketmanager
- Verbesserung des Mathe-Layouts
- ...

9.3. Wer sollte Typst (nicht) benutzen?

Pros:

- ✓ **steile** Lernkurve
- ✓ sehr dynamisch
- ✓ aktive Community
- ✓ schnelle Kompilierzeit
- ✓ verständliche Fehlermeldungen

Cons:

- viele Programmierer-Ansätze
- komplexes Layouting & Fußnoten schwer umsetzbar
- Pure Functions können schwer sein (States, Counter, ...)

9.4. Weiteres

Übrigens: Diese gesamte Präsentation wurde alleine in Typst erstellt.

Typst Dokumentation:

- <https://typst.app/docs/>

Offizielles Typst-Tutorial:

- <https://typst.app/docs/tutorial>

Offizieller Typst-Discord:

- <https://discord.gg/2uDybryKPe>

Code für diese Präsentation und weitere Beispiele:

- <https://github.com/survari/typst-seminar>