

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. Was Typst noch so alles kann	33
9. Was noch fehlt	35
10. Abschluss und Weiteres	36

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Rule 'pdflatex': The following rules & subrules became out-of-date: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
<code>\documentclass{article}</code>	+ Dies	1. Dies
<code>\begin{document}</code>	+ Ist	2. Ist
<code>\begin{enumerate}</code>	+ Eine	3. Eine
<code> \item Dies</code>	+ Liste!	4. Liste!
<code> \item Ist</code>		
<code> \item Eine</code>		
<code> \item Liste!</code>		
<code>\end{enumerate}</code>		
<code>\end{document}</code>		

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!

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%, "img/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-Regeln

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)
```

Ab jetzt wird jede erste Zeile eines Absatzes eingerückt!

Wirklich, versprochen!

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!

Wirklich, versprochen!

5.11. Show-Regeln

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

==== Hallo!

===== Welt!

```
// aus dem offiziell  
// 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)

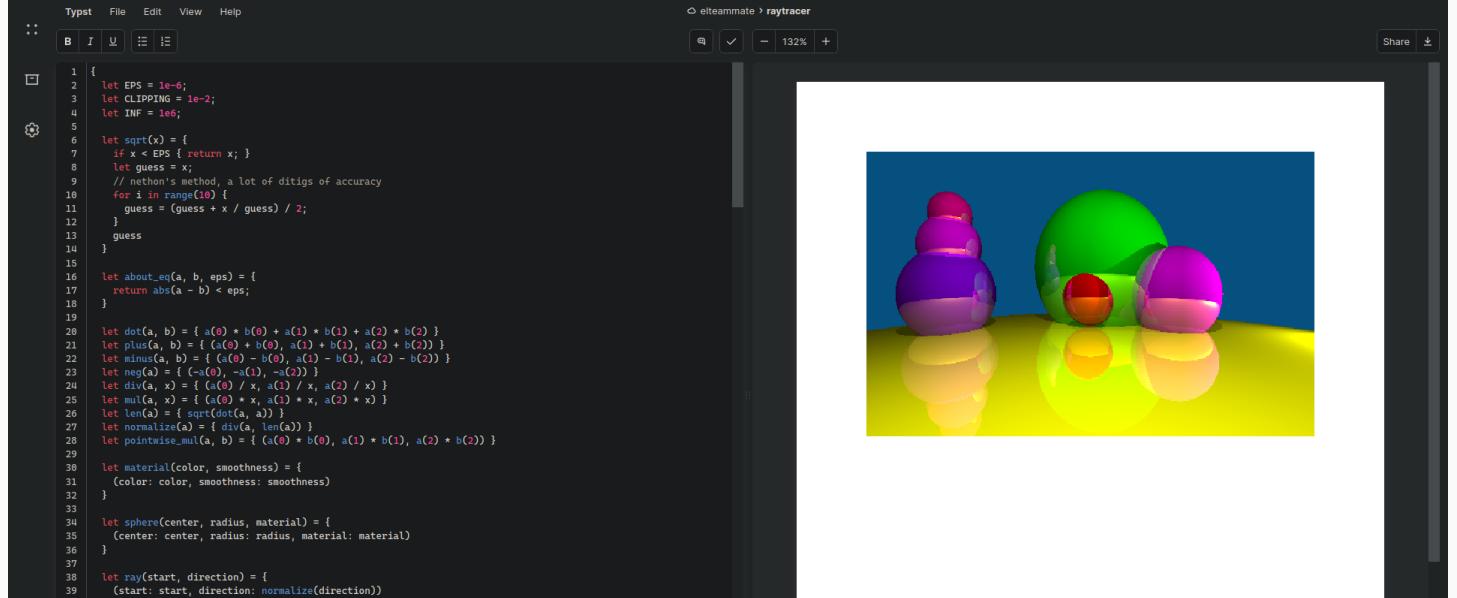
6. Die Typst-Dokumentation...

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

7. Eigene Templates und Skripts

8. Was Typst noch so alles kann

8.1. Raytracing



The image shows a code editor window with a dark theme. The left pane displays a block of TypeScript code for a raytracer. The right pane shows a 3D rendering of several spheres of different sizes and colors (purple, green, 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 (let i = 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) = Math.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: color,
32     smoothness: smoothness
33   };
34
35   let sphere(center, radius, material) = {
36     center: center,
37     radius: radius,
38     material: material
39   };
39
40   let ray(start, direction) = [
41     start: start,
42     direction: normalize(direction)
43   ];
43
44   let intersect(sphere, ray) = [
45     sphere: sphere,
46     ray: ray
47   ];
47
48   let hit(ray, spheres) = [
49     ray: ray,
50     spheres: spheres
51   ];
51
52   let closestHit(ray, spheres) = [
53     ray: ray,
54     spheres: spheres
55   ];
55
56   let intersectSphere(ray, sphere) = [
57     ray: ray,
58     sphere: sphere
59   ];
59
60   let intersectRaySphere(ray, sphere) = [
61     ray: ray,
62     sphere: sphere
63   ];
63
64   let intersectRayRay(ray, ray) = [
65     ray: ray,
66     ray2: ray
67   ];
67
68   let intersectRayPlane(ray, plane) = [
69     ray: ray,
70     plane: plane
71   ];
71
72   let intersectPlanePlane(plane, plane2) = [
73     plane: plane,
74     plane2: plane2
75   ];
75
76   let intersectRayPlane(ray, plane) = [
77     ray: ray,
78     plane: plane
79   ];
79
80   let intersectPlaneRay(plane, ray) = [
81     plane: plane,
82     ray: ray
83   ];
83
84   let intersectRayPlane(ray, plane) = [
85     ray: ray,
86     plane: plane
87   ];
87
88   let intersectPlaneRay(plane, ray) = [
89     plane: plane,
90     ray: ray
91   ];
91
92   let intersectRayRay(ray, ray2) = [
93     ray: ray,
94     ray2: ray2
95   ];
95
96   let intersectRayPlane(ray, plane) = [
97     ray: ray,
98     plane: plane
99   ];
99
100  let intersectPlaneRay(plane, ray) = [
101    plane: plane,
102    ray: ray
103  ];
103
104  let intersectRayRay(ray, ray2) = [
105    ray: ray,
106    ray2: ray2
107  ];
107
108  let intersectRayPlane(ray, plane) = [
109    ray: ray,
110    plane: plane
111  ];
111
112  let intersectPlaneRay(plane, ray) = [
113    plane: plane,
114    ray: ray
115  ];
115
116  let intersectRayRay(ray, ray2) = [
117    ray: ray,
118    ray2: ray2
119  ];
119
120  let intersectRayPlane(ray, plane) = [
121    ray: ray,
122    plane: plane
123  ];
123
124  let intersectPlaneRay(plane, ray) = [
125    plane: plane,
126    ray: ray
127  ];
127
128  let intersectRayRay(ray, ray2) = [
129    ray: ray,
130    ray2: ray2
131  ];
131
132  let intersectRayPlane(ray, plane) = [
133    ray: ray,
134    plane: plane
135  ];
135
136  let intersectPlaneRay(plane, ray) = [
137    plane: plane,
138    ray: ray
139  ];
139
140  let intersectRayRay(ray, ray2) = [
141    ray: ray,
142    ray2: ray2
143  ];
143
144  let intersectRayPlane(ray, plane) = [
145    ray: ray,
146    plane: plane
147  ];
147
148  let intersectPlaneRay(plane, ray) = [
149    plane: plane,
150    ray: ray
151  ];
151
152  let intersectRayRay(ray, ray2) = [
153    ray: ray,
154    ray2: ray2
155  ];
155
156  let intersectRayPlane(ray, plane) = [
157    ray: ray,
158    plane: plane
159  ];
159
160  let intersectPlaneRay(plane, ray) = [
161    plane: plane,
162    ray: ray
163  ];
163
164  let intersectRayRay(ray, ray2) = [
165    ray: ray,
166    ray2: ray2
167  ];
167
168  let intersectRayPlane(ray, plane) = [
169    ray: ray,
170    plane: plane
171  ];
171
172  let intersectPlaneRay(plane, ray) = [
173    plane: plane,
174    ray: ray
175  ];
175
176  let intersectRayRay(ray, ray2) = [
177    ray: ray,
178    ray2: ray2
179  ];
179
180  let intersectRayPlane(ray, plane) = [
181    ray: ray,
182    plane: plane
183  ];
183
184  let intersectPlaneRay(plane, ray) = [
185    plane: plane,
186    ray: ray
187  ];
187
188  let intersectRayRay(ray, ray2) = [
189    ray: ray,
190    ray2: ray2
191  ];
191
192  let intersectRayPlane(ray, plane) = [
193    ray: ray,
194    plane: plane
195  ];
195
196  let intersectPlaneRay(plane, ray) = [
197    plane: plane,
198    ray: ray
199  ];
199
200  let intersectRayRay(ray, ray2) = [
201    ray: ray,
202    ray2: ray2
203  ];
203
204  let intersectRayPlane(ray, plane) = [
205    ray: ray,
206    plane: plane
207  ];
207
208  let intersectPlaneRay(plane, ray) = [
209    plane: plane,
210    ray: ray
211  ];
211
212  let intersectRayRay(ray, ray2) = [
213    ray: ray,
214    ray2: ray2
215  ];
215
216  let intersectRayPlane(ray, plane) = [
217    ray: ray,
218    plane: plane
219  ];
219
220  let intersectPlaneRay(plane, ray) = [
221    plane: plane,
222    ray: ray
223  ];
223
224  let intersectRayRay(ray, ray2) = [
225    ray: ray,
226    ray2: ray2
227  ];
227
228  let intersectRayPlane(ray, plane) = [
229    ray: ray,
230    plane: plane
231  ];
231
232  let intersectPlaneRay(plane, ray) = [
233    plane: plane,
234    ray: ray
235  ];
235
236  let intersectRayRay(ray, ray2) = [
237    ray: ray,
238    ray2: ray2
239  ];
239
240  let intersectRayPlane(ray, plane) = [
241    ray: ray,
242    plane: plane
243  ];
243
244  let intersectPlaneRay(plane, ray) = [
245    plane: plane,
246    ray: ray
247  ];
247
248  let intersectRayRay(ray, ray2) = [
249    ray: ray,
250    ray2: ray2
251  ];
251
252  let intersectRayPlane(ray, plane) = [
253    ray: ray,
254    plane: plane
255  ];
255
256  let intersectPlaneRay(plane, ray) = [
257    plane: plane,
258    ray: ray
259  ];
259
260  let intersectRayRay(ray, ray2) = [
261    ray: ray,
262    ray2: ray2
263  ];
263
264  let intersectRayPlane(ray, plane) = [
265    ray: ray,
266    plane: plane
267  ];
267
268  let intersectPlaneRay(plane, ray) = [
269    plane: plane,
270    ray: ray
271  ];
271
272  let intersectRayRay(ray, ray2) = [
273    ray: ray,
274    ray2: ray2
275  ];
275
276  let intersectRayPlane(ray, plane) = [
277    ray: ray,
278    plane: plane
279  ];
279
280  let intersectPlaneRay(plane, ray) = [
281    plane: plane,
282    ray: ray
283  ];
283
284  let intersectRayRay(ray, ray2) = [
285    ray: ray,
286    ray2: ray2
287  ];
287
288  let intersectRayPlane(ray, plane) = [
289    ray: ray,
290    plane: plane
291  ];
291
292  let intersectPlaneRay(plane, ray) = [
293    plane: plane,
294    ray: ray
295  ];
295
296  let intersectRayRay(ray, ray2) = [
297    ray: ray,
298    ray2: ray2
299  ];
299
300  let intersectRayPlane(ray, plane) = [
301    ray: ray,
302    plane: plane
303  ];
303
304  let intersectPlaneRay(plane, ray) = [
305    plane: plane,
306    ray: ray
307  ];
307
308  let intersectRayRay(ray, ray2) = [
309    ray: ray,
310    ray2: ray2
311  ];
311
312  let intersectRayPlane(ray, plane) = [
313    ray: ray,
314    plane: plane
315  ];
315
316  let intersectPlaneRay(plane, ray) = [
317    plane: plane,
318    ray: ray
319  ];
319
320  let intersectRayRay(ray, ray2) = [
321    ray: ray,
322    ray2: ray2
323  ];
323
324  let intersectRayPlane(ray, plane) = [
325    ray: ray,
326    plane: plane
327  ];
327
328  let intersectPlaneRay(plane, ray) = [
329    plane: plane,
330    ray: ray
331  ];
331
332  let intersectRayRay(ray, ray2) = [
333    ray: ray,
334    ray2: ray2
335  ];
335
336  let intersectRayPlane(ray, plane) = [
337    ray: ray,
338    plane: plane
339  ];
339
340  let intersectPlaneRay(plane, ray) = [
341    plane: plane,
342    ray: ray
343  ];
343
344  let intersectRayRay(ray, ray2) = [
345    ray: ray,
346    ray2: ray2
347  ];
347
348  let intersectRayPlane(ray, plane) = [
349    ray: ray,
350    plane: plane
351  ];
351
352  let intersectPlaneRay(plane, ray) = [
353    plane: plane,
354    ray: ray
355  ];
355
356  let intersectRayRay(ray, ray2) = [
357    ray: ray,
358    ray2: ray2
359  ];
359
360  let intersectRayPlane(ray, plane) = [
361    ray: ray,
362    plane: plane
363  ];
363
364  let intersectPlaneRay(plane, ray) = [
365    plane: plane,
366    ray: ray
367  ];
367
368  let intersectRayRay(ray, ray2) = [
369    ray: ray,
370    ray2: ray2
371  ];
371
372  let intersectRayPlane(ray, plane) = [
373    ray: ray,
374    plane: plane
375  ];
375
376  let intersectPlaneRay(plane, ray) = [
377    plane: plane,
378    ray: ray
379  ];
379
380  let intersectRayRay(ray, ray2) = [
381    ray: ray,
382    ray2: ray2
383  ];
383
384  let intersectRayPlane(ray, plane) = [
385    ray: ray,
386    plane: plane
387  ];
387
388  let intersectPlaneRay(plane, ray) = [
389    plane: plane,
390    ray: ray
391  ];
391
392  let intersectRayRay(ray, ray2) = [
393    ray: ray,
394    ray2: ray2
395  ];
395
396  let intersectRayPlane(ray, plane) = [
397    ray: ray,
398    plane: plane
399  ];
399
400  let intersectPlaneRay(plane, ray) = [
401    plane: plane,
402    ray: ray
403  ];
403
404  let intersectRayRay(ray, ray2) = [
405    ray: ray,
406    ray2: ray2
407  ];
407
408  let intersectRayPlane(ray, plane) = [
409    ray: ray,
410    plane: plane
411  ];
411
412  let intersectPlaneRay(plane, ray) = [
413    plane: plane,
414    ray: ray
415  ];
415
416  let intersectRayRay(ray, ray2) = [
417    ray: ray,
418    ray2: ray2
419  ];
419
420  let intersectRayPlane(ray, plane) = [
421    ray: ray,
422    plane: plane
423  ];
423
424  let intersectPlaneRay(plane, ray) = [
425    plane: plane,
426    ray: ray
427  ];
427
428  let intersectRayRay(ray, ray2) = [
429    ray: ray,
430    ray2: ray2
431  ];
431
432  let intersectRayPlane(ray, plane) = [
433    ray: ray,
434    plane: plane
435  ];
435
436  let intersectPlaneRay(plane, ray) = [
437    plane: plane,
438    ray: ray
439  ];
439
440  let intersectRayRay(ray, ray2) = [
441    ray: ray,
442    ray2: ray2
443  ];
443
444  let intersectRayPlane(ray, plane) = [
445    ray: ray,
446    plane: plane
447  ];
447
448  let intersectPlaneRay(plane, ray) = [
449    plane: plane,
450    ray: ray
451  ];
451
452  let intersectRayRay(ray, ray2) = [
453    ray: ray,
454    ray2: ray2
455  ];
455
456  let intersectRayPlane(ray, plane) = [
457    ray: ray,
458    plane: plane
459  ];
459
460  let intersectPlaneRay(plane, ray) = [
461    plane: plane,
462    ray: ray
463  ];
463
464  let intersectRayRay(ray, ray2) = [
465    ray: ray,
466    ray2: ray2
467  ];
467
468  let intersectRayPlane(ray, plane) = [
469    ray: ray,
470    plane: plane
471  ];
471
472  let intersectPlaneRay(plane, ray) = [
473    plane: plane,
474    ray: ray
475  ];
475
476  let intersectRayRay(ray, ray2) = [
477    ray: ray,
478    ray2: ray2
479  ];
479
480  let intersectRayPlane(ray, plane) = [
481    ray: ray,
482    plane: plane
483  ];
483
484  let intersectPlaneRay(plane, ray) = [
485    plane: plane,
486    ray: ray
487  ];
487
488  let intersectRayRay(ray, ray2) = [
489    ray: ray,
490    ray2: ray2
491  ];
491
492  let intersectRayPlane(ray, plane) = [
493    ray: ray,
494    plane: plane
495  ];
495
496  let intersectPlaneRay(plane, ray) = [
497    plane: plane,
498    ray: ray
499  ];
499
500  let intersectRayRay(ray, ray2) = [
501    ray: ray,
502    ray2: ray2
503  ];
503
504  let intersectRayPlane(ray, plane) = [
505    ray: ray,
506    plane: plane
507  ];
507
508  let intersectPlaneRay(plane, ray) = [
509    plane: plane,
510    ray: ray
511  ];
511
512  let intersectRayRay(ray, ray2) = [
513    ray: ray,
514    ray2: ray2
515  ];
515
516  let intersectRayPlane(ray, plane) = [
517    ray: ray,
518    plane: plane
519  ];
519
520  let intersectPlaneRay(plane, ray) = [
521    plane: plane,
522    ray: ray
523  ];
523
524  let intersectRayRay(ray, ray2) = [
525    ray: ray,
526    ray2: ray2
527  ];
527
528  let intersectRayPlane(ray, plane) = [
529    ray: ray,
530    plane: plane
531  ];
531
532  let intersectPlaneRay(plane, ray) = [
533    plane: plane,
534    ray: ray
535  ];
535
536  let intersectRayRay(ray, ray2) = [
537    ray: ray,
538    ray2: ray2
539  ];
539
540  let intersectRayPlane(ray, plane) = [
541    ray: ray,
542    plane: plane
543  ];
543
544  let intersectPlaneRay(plane, ray) = [
545    plane: plane,
546    ray: ray
547  ];
547
548  let intersectRayRay(ray, ray2) = [
549    ray: ray,
550    ray2: ray2
551  ];
551
552  let intersectRayPlane(ray, plane) = [
553    ray: ray,
554    plane: plane
555  ];
555
556  let intersectPlaneRay(plane, ray) = [
557    plane: plane,
558    ray: ray
559  ];
559
560  let intersectRayRay(ray, ray2) = [
561    ray: ray,
562    ray2: ray2
563  ];
563
564  let intersectRayPlane(ray, plane) = [
565    ray: ray,
566    plane: plane
567  ];
567
568  let intersectPlaneRay(plane, ray) = [
569    plane: plane,
570    ray: ray
571  ];
571
572  let intersectRayRay(ray, ray2) = [
573    ray: ray,
574    ray2: ray2
575  ];
575
576  let intersectRayPlane(ray, plane) = [
577    ray: ray,
578    plane: plane
579  ];
579
580  let intersectPlaneRay(plane, ray) = [
581    plane: plane,
582    ray: ray
583  ];
583
584  let intersectRayRay(ray, ray2) = [
585    ray: ray,
586    ray2: ray2
587  ];
587
588  let intersectRayPlane(ray, plane) = [
589    ray: ray,
590    plane: plane
591  ];
591
592  let intersectPlaneRay(plane, ray) = [
593    plane: plane,
594    ray: ray
595  ];
595
596  let intersectRayRay(ray, ray2) = [
597    ray: ray,
598    ray2: ray2
599  ];
599
600  let intersectRayPlane(ray, plane) = [
601    ray: ray,
602    plane: plane
603  ];
603
604  let intersectPlaneRay(plane, ray) = [
605    plane: plane,
606    ray: ray
607  ];
607
608  let intersectRayRay(ray, ray2) = [
609    ray: ray,
610    ray2: ray2
611  ];
611
612  let intersectRayPlane(ray, plane) = [
613    ray: ray,
614    plane: plane
615  ];
615
616  let intersectPlaneRay(plane, ray) = [
617    plane: plane,
618    ray: ray
619  ];
619
620  let intersectRayRay(ray, ray2) = [
621    ray: ray,
622    ray2: ray2
623  ];
623
624  let intersectRayPlane(ray, plane) = [
625    ray: ray,
626    plane: plane
627  ];
627
628  let intersectPlaneRay(plane, ray) = [
629    plane: plane,
630    ray: ray
631  ];
631
632  let intersectRayRay(ray, ray2) = [
633    ray: ray,
634    ray2: ray2
635  ];
635
636  let intersectRayPlane(ray, plane) = [
637    ray: ray,
638    plane: plane
639  ];
639
640  let intersectPlaneRay(plane, ray) = [
641    plane: plane,
642    ray: ray
643  ];
643
644  let intersectRayRay(ray, ray2) = [
645    ray: ray,
646    ray2: ray2
647  ];
647
648  let intersectRayPlane(ray, plane) = [
649    ray: ray,
650    plane: plane
651  ];
651
652  let intersectPlaneRay(plane, ray) = [
653    plane: plane,
654    ray: ray
655  ];
655
656  let intersectRayRay(ray, ray2) = [
657    ray: ray,
658    ray2: ray2
659  ];
659
660  let intersectRayPlane(ray, plane) = [
661    ray: ray,
662    plane: plane
663  ];
663
664  let intersectPlaneRay(plane, ray) = [
665    plane: plane,
666    ray: ray
667  ];
667
668  let intersectRayRay(ray, ray2) = [
669    ray: ray,
670    ray2: ray2
671  ];
671
672  let intersectRayPlane(ray, plane) = [
673    ray: ray,
674    plane: plane
675  ];
675
676  let intersectPlaneRay(plane, ray) = [
677    plane: plane,
678    ray: ray
679  ];
679
680  let intersectRayRay(ray, ray2) = [
681    ray: ray,
682    ray2: ray2
683  ];
683
684  let intersectRayPlane(ray, plane) = [
685    ray: ray,
686    plane: plane
687  ];
687
688  let intersectPlaneRay(plane, ray) = [
689    plane: plane,
690    ray: ray
691  ];
691
692  let intersectRayRay(ray, ray2) = [
693    ray: ray,
694    ray2: ray2
695  ];
695
696  let intersectRayPlane(ray, plane) = [
697    ray: ray,
698    plane: plane
699  ];
699
700  let intersectPlaneRay(plane, ray) = [
701    plane: plane,
702    ray: ray
703  ];
703
704  let intersectRayRay(ray, ray2) = [
705    ray: ray,
706    ray2: ray2
707  ];
707
708  let intersectRayPlane(ray, plane) = [
709    ray: ray,
710    plane: plane
711  ];
711
712  let intersectPlaneRay(plane, ray) = [
713    plane: plane,
714    ray: ray
715  ];
715
716  let intersectRayRay(ray, ray2) = [
717    ray: ray,
718    ray2: ray2
719  ];
719
720  let intersectRayPlane(ray, plane) = [
721    ray: ray,
722    plane: plane
723  ];
723
724  let intersectPlaneRay(plane, ray) = [
725    plane: plane,
726    ray: ray
727  ];
727
728  let intersectRayRay(ray, ray2) = [
729    ray: ray,
730    ray2: ray2
731  ];
731
732  let intersectRayPlane(ray, plane) = [
733    ray: ray,
734    plane: plane
735  ];
735
736  let intersectPlaneRay(plane, ray) = [
737    plane: plane,
738    ray: ray
739  ];
739
740  let intersectRayRay(ray, ray2) = [
741    ray: ray,
742    ray2: ray2
743  ];
743
744  let intersectRayPlane(ray, plane) = [
745    ray: ray,
746    plane: plane
747  ];
747
748  let intersectPlaneRay(plane, ray) = [
749    plane: plane,
750    ray: ray
751  ];
751
752  let intersectRayRay(ray, ray2) = [
753    ray: ray,
754    ray2: ray2
755  ];
755
756  let intersectRayPlane(ray, plane) = [
757    ray: ray,
758    plane: plane
759  ];
759
760  let intersectPlaneRay(plane, ray) = [
761    plane: plane,
762    ray: ray
763  ];
763
764  let intersectRayRay(ray, ray2) = [
765    ray: ray,
766    ray2: ray2
767  ];
767
768  let intersectRayPlane(ray, plane) = [
769    ray: ray,
770    plane: plane
771  ];
771
772  let intersectPlaneRay(plane, ray) = [
773    plane: plane,
774    ray: ray
775  ];
775
776  let intersectRayRay(ray, ray2) = [
777    ray: ray,
778    ray2: ray2
779  ];
779
780  let intersectRayPlane(ray, plane) = [
781    ray: ray,
782    plane: plane
783  ];
783
784  let intersectPlaneRay(plane, ray) = [
785    plane: plane,
786    ray: ray
787  ];
787
788  let intersectRayRay(ray, ray2) = [
789    ray: ray,
790    ray2: ray2
791  ];
791
792  let intersectRayPlane(ray, plane) = [
793    ray: ray,
794    plane: plane
795  ];
795
796  let intersectPlaneRay(plane, ray) = [
797    plane: plane,
798    ray: ray
799  ];
799
800  let intersectRayRay(ray, ray2) = [
801    ray: ray,
802    ray2: ray2
803  ];
803
804  let intersectRayPlane(ray, plane) = [
805    ray: ray,
806    plane: plane
807  ];
807
808  let intersectPlaneRay(plane, ray) = [
809    plane: plane,
810    ray: ray
811  ];
811
812  let intersectRayRay(ray, ray2) = [
813    ray: ray,
814    ray2: ray2
815  ];
815
816  let intersectRayPlane(ray, plane) = [
817    ray: ray,
818    plane: plane
819  ];
819
820  let intersectPlaneRay(plane, ray) = [
821    plane: plane,
822    ray: ray
823  ];
823
824  let intersectRayRay(ray, ray2) = [
825    ray: ray,
826    ray2: ray2
827  ];
827
828  let intersectRayPlane(ray, plane) = [
829    ray: ray,
830    plane: plane
831  ];
831
832  let intersectPlaneRay(plane, ray) = [
833    plane: plane,
834    ray: ray
835  ];
835
836  let intersectRayRay(ray, ray2) = [
837    ray: ray,
838    ray2: ray2
839  ];
839
840  let intersectRayPlane(ray, plane) = [
841    ray: ray,
842    plane: plane
843  ];
843
844  let intersectPlaneRay(plane, ray) = [
845    plane: plane,
846    ray: ray
847  ];
847
848  let intersectRayRay(ray, ray2) = [
849    ray: ray,
850    ray2: ray2
851  ];
851
852  let intersectRayPlane(ray, plane) = [
853    ray: ray,
854    plane: plane
855  ];
855
856  let intersectPlaneRay(plane, ray) = [
857    plane: plane,
858    ray: ray
859  ];
859
860  let intersectRayRay(ray, ray2) = [
861    ray: ray,
862    ray2: ray2
863  ];
863
864  let intersectRayPlane(ray, plane) = [
865    ray: ray,
866    plane: plane
867  ];
867
868  let intersectPlaneRay(plane, ray) = [
869    plane: plane,
870    ray: ray
871  ];
871
872  let intersectRayRay(ray, ray2) = [
873    ray: ray,
874    ray2: ray2
875  ];
875
876  let intersectRayPlane(ray, plane) = [
877    ray: ray,
878    plane: plane
879  ];
879
880  let intersectPlaneRay(plane, ray) = [
881    plane: plane,
882    ray: ray
883  ];
883
884  let intersectRayRay(ray, ray2) = [
885    ray: ray,
886    ray2: ray2
887  ];
887
888  let intersectRayPlane(ray, plane) = [
889    ray: ray,
890    plane: plane
891  ];
891
892  let intersectPlaneRay(plane, ray) = [
893    plane: plane,
894    ray: ray
895  ];
895
896  let intersectRayRay(ray, ray2) = [
897    ray: ray,
898    ray2: ray2
899  ];
899
900  let intersectRayPlane(ray, plane) = [
901    ray: ray,
902    plane: plane
903  ];
903
904  let intersectPlaneRay(plane, ray) = [
905    plane: plane,
906    ray: ray
907  ];
907
908  let intersectRayRay(ray, ray2) = [
909    ray: ray,
910    ray2: ray2
911  ];
911
912  let intersectRayPlane(ray, plane) = [
913    ray: ray,
914    plane: plane
915  ];
915
916  let intersectPlaneRay(plane, ray) = [
917    plane: plane,
918    ray: ray
919  ];
919
920  let intersectRayRay(ray, ray2) = [
921    ray: ray,
922    ray2: ray2
923  ];
923
924  let intersectRayPlane(ray, plane) = [
925    ray: ray,
926    plane: plane
927  ];
927
928  let intersectPlaneRay(plane, ray) = [
929    plane: plane,
930    ray: ray
931  ];
931
932  let intersectRayRay(ray, ray2) = [
933    ray: ray,
934    ray2: ray2
935  ];
935
936  let intersectRayPlane(ray, plane) = [
937    ray: ray,
938    plane: plane
939  ];
939
940  let intersectPlaneRay(plane, ray) = [
941    plane: plane,
942    ray: ray
943  ];
943
944  let intersectRayRay(ray, ray2) = [
945    ray: ray,
946    ray2: ray2
947  ];
947
948  let intersectRayPlane(ray, plane) = [
949    ray: ray,
950    plane: plane
951  ];
951
952  let intersectPlaneRay(plane, ray) = [
953    plane: plane,
954    ray: ray
955  ];
955
956  let intersectRayRay(ray, ray2) = [
957    ray: ray,
958    ray2: ray2
959  ];
959
960  let intersectRayPlane(ray, plane) = [
961    ray: ray,
962    plane: plane
963  ];
963
964  let intersectPlaneRay(plane, ray) = [
965    plane: plane,
966    ray: ray
967  ];
967
968  let intersectRayRay(ray, ray2) = [
969    ray: ray,
970    ray2: ray2
971  ];
971
972  let intersectRayPlane(ray, plane) = [
973    ray: ray,
974    plane: plane
975  ];
975
976  let intersectPlaneRay(plane, ray) = [
977    plane: plane,
978    ray: ray
979  ];
979
980  let intersectRayRay(ray, ray2) = [
981    ray: ray,
982    ray2: ray2
983  ];
983
984  let intersectRayPlane(ray, plane) = [
985    ray: ray,
986    plane: plane
987  ];
987
988  let intersectPlaneRay(plane, ray) = [
989    plane: plane,
990    ray: ray
991  ];
991
992  let intersectRayRay(ray, ray2) = [
993    ray: ray,
994    ray2: ray2
995  ];
995
996  let intersectRayPlane(ray, plane) = [
997    ray: ray,
998    plane: plane
999  ];
999
1000  let intersectPlaneRay(plane, ray) = [
1001    plane: plane,
1002    ray: ray
1003  ];
1003
1004  let intersectRayRay(ray, ray2) = [
1005    ray: ray,
1006    ray2: ray2
1007  ];
1007
1008  let intersectRayPlane(ray, plane) = [
1009    ray: ray,
1010    plane: plane
1011  ];
1011
1012  let intersectPlaneRay(plane, ray) = [
1013    plane: plane,
1014    ray: ray
1015  ];
1015
1016  let intersectRayRay(ray, ray2) = [
1017    ray: ray,
1018    ray2: ray2
1019  ];
1019
1020  let intersectRayPlane(ray, plane) = [
1021    ray: ray,
1022    plane: plane
1023  ];
1023
1024  let intersectPlaneRay(plane, ray) = [
1025    plane: plane,
1026    ray: ray
1027  ];
1027
1028  let intersectRayRay(ray, ray2) = [
1029    ray: ray,
1030    ray2: ray2
1031  ];
1031
1032  let intersectRayPlane(ray, plane) = [
1033    ray: ray,
1034    plane: plane
1035  ];
1035
1036  let intersectPlaneRay(plane, ray) = [
1037    plane: plane,
1038    ray: ray
1039  ];
1039
1040  let intersectRayRay(ray, ray2) = [
1041    ray: ray,
1042    ray2: ray2
1043  ];
1043
1044  let intersectRayPlane(ray, plane) = [
1045    ray: ray,
1046    plane: plane
1047  ];
1047
1048  let intersectPlaneRay(plane, ray) = [
1049    plane: plane,
1050    ray: ray
1051  ];
1051
1052  let intersectRayRay(ray, ray2) = [
1053    ray: ray,
1054    ray2: ray2
1055  ];
1055
1056  let intersectRayPlane(ray, plane) = [
1057    ray: ray,
1058    plane: plane
1059  ];
1059
1060  let intersectPlaneRay(plane, ray) = [
1061    plane: plane,
1062    ray: ray
1063  ];
1063
1064  let intersectRayRay(ray, ray2) = [
1065    ray: ray,
1066    ray2: ray2
1067  ];
1067
1068  let intersectRayPlane(ray, plane) = [
1069    ray: ray,
1070    plane: plane
1071  ];
1071
1072  let intersectPlaneRay(plane, ray) = [
1073    plane: plane,
1074    ray: ray
1075  ];
1075
1076  let intersectRayRay(ray, ray2) = [
1077    ray: ray,
1078    ray2: ray2
1079  ];
1079
1080  let intersectRayPlane(ray, plane) = [
1081    ray: ray,
1082    plane: plane
1083  ];
1083
1084  let intersectPlaneRay(plane, ray) = [
1085    plane: plane,
1086    ray: ray
1087  ];
1087
1088  let intersectRayRay(ray, ray2) = [
1089    ray: ray,
1090    ray2: ray2
1091  ];
1091
1092  let intersectRayPlane(ray, plane) = [
1093    ray: ray,
1094    plane: plane
1095  ];
1095
1096  let intersectPlaneRay(plane, ray) = [
1097    plane: plane,
1098    ray: ray
1099  ];
1099
1100  let intersectRayRay(ray, ray2) = [
1101    ray: ray,
1102    ray2: ray2
1103  ];
1103
1104  let intersectRayPlane(ray, plane) = [
1105    ray: ray,
1106    plane: plane
1107  ];
1107
1108  let intersectPlaneRay(plane, ray) = [
1109    plane: plane,
1110    ray: ray
1111  ];
1111
1112  let intersectRayRay(ray, ray2) = [
1113    ray: ray,
1114    ray2: ray2
1115  ];
1115
1116  let intersectRayPlane(ray, plane) = [
1117    ray: ray,
1118    plane: plane
1119  ];
1119
1120  let intersectPlaneRay(plane, ray) = [
1121    plane: plane,
1122    ray: ray
1123  ];
1123
1124  let intersectRayRay(ray, ray2) = [
1125    ray: ray,
1126    ray2: ray2
1127  ];
1127
1128  let intersectRayPlane(ray, plane) = [
1129    ray: ray,
1130    plane: plane
1131  ];
1131
1132  let intersectPlaneRay(plane, ray) = [
1133    plane: plane,
1134    ray: ray
1135  ];
1135
1136  let intersectRayRay(ray, ray2) = [
1137    ray: ray,
1138    ray2: ray2
1139  ];
1139
1140  let intersectRayPlane(ray, plane) = [
1141    ray: ray,
1142    plane: plane
1143  ];
1143
1144  let intersectPlaneRay(plane, ray) = [
1145    plane: plane,
1146    ray: ray
1147  ];
1147
1148  let intersectRayRay(ray, ray2) = [
1149    ray: ray,
1150    ray2: ray2
1151  ];
1151
1152  let intersectRayPlane(ray, plane) = [
1153    ray: ray,
1154    plane: plane
1155  ];
1155
1156  let intersectPlaneRay(plane, ray) = [
1157    plane: plane,
1158    ray: ray
1159  ];
1159
1160  let intersectRayRay(ray, ray2) = [
1161    ray: ray,
1162    ray2: ray2
1163  ];
1163
1164  let intersectRayPlane(ray, plane) = [
1165    ray: ray,
1166    plane: plane
1167  ];
1167
1168  let intersectPlaneRay(plane, ray) = [
1169    plane: plane,
1170    ray: ray
1171  ];
1171
1172  let
```

9. Was noch fehlt

10. Abschluss und Weiteres

10.1. 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, ...)

10.2. Erwartete Neuerungen¹

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

10.3. 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>