Test Markdown

Wuppie Fluppie

TL;DR

Text für TL;DR …

In Parr (2014) wird geschrieben, blablablabla …

Wir können hier sowohl Inline-Math ($a^i b^{2*i}$) als auch Block-Math:

$$\Phi(\mathbf{g}_i) = F(\Gamma(\mathbf{g}_i)) - w \cdot \sum_j (Z_j(\Gamma(\mathbf{g}_i)))^2$$

Code sollte auch gehen: `inline`, aber auch block:

```
fun fib(x) {
    if (x == 0) {
        return 0;
    } else {
        if (x == 1) {
            return 1;
        } else {
            fib(x - 1) + fib(x - 2);
        }
    }
}

var wuppie = fib(4);
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In eu vulputate nisl. Ut scelerisque magna eros, at semper lectus vulputate vitae. Nullam egestas tellus lorem, eget sodales mauris lacinia ut. Etiam a viverra ex. Nam eu nisl vel nisl cursus condimentum. Aliquam accumsan augue ut consequat viverra. Curabitur sagittis est mauris, at molestie arcu condimentum ut. Quisque efficitur porta maximus. Donec non leo est. Aenean interdum condimentum libero, ac cursus dolor condimentum in. Nullam lorem ex, iaculis a orci vitae, iaculis aliquam enim.

Quisque est lacus, pellentesque vitae fringilla vitae, bibendum sit amet dolor. Proin rutrum metus sit amet hendrerit lobortis. Fusce ut ultrices enim. Morbi a urna rutrum, fringilla augue mattis, mollis lacus. Pellentesque elementum vitae magna ac feugiat. Vestibulum et metus eget augue finibus fringilla ac at velit. Cras eleifend in nisl ac commodo. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Suspendisse id mi nec diam pellentesque varius. Morbi in consequat neque, at bibendum purus. Quisque a est bibendum, pellentesque quam ac, fermentum diam. Cras augue nibh, tincidunt eu mollis at, hendrerit nec ex. Integer condimentum neque velit, eget aliquam justo iaculis ac.

Nam consequat vehicula faucibus. In consequat sed lacus sed congue. Ut quis risus vel erat tincidunt molestie. Nullam nibh lorem, placerat et dignissim porttitor, mollis eget nisl. Mauris eu justo nisi. Sed viverra, enim in tincidunt blandit, dui eros mattis nulla, in elementum neque ex sed diam. Aliquam nec neque vitae sapien pulvinar tempus. Donec gravida interdum nunc sed feugiat. Curabitur finibus sed urna at rutrum. Maecenas mollis pulvinar lobortis. Mauris dignissim orci ut metus eleifend, eget porttitor purus rhoncus. Phasellus volutpat egestas odio mollis pharetra. Cras sem dolor, commodo eu lectus vel, ultrices fringilla tellus. Fusce eleifend orci sed porttitor imperdiet. Phasellus suscipit, est ut semper blandit, ligula dui scelerisque felis, varius tincidunt tellus est vitae lectus. Phasellus eu molestie urna.

Sed sed leo vestibulum, iaculis justo in, aliquet mauris. Nunc luctus, metus quis vulputate lobortis, libero lacus imperdiet turpis, sit amet hendrerit magna nisi at leo. Pellentesque vehicula, mauris nec varius tempus, odio diam hendrerit mi, id porttitor ex felis ut sem. Vivamus tellus lacus, vehicula a augue non, aliquet aliquam quam. Donec dapibus quam sed ante blandit, in imperdiet enim dapibus. Etiam imperdiet sapien nec quam lobortis aliquet. Nam est eros, luctus consequat nulla ac, dapibus efficitur augue. Morbi tempor ex sed consectetur pharetra. Morbi ullamcorper ac enim nec accumsan. Nulla vestibulum eu turpis et rutrum.

Donec laoreet lectus at laoreet condimentum. Nulla porttitor elit iaculis turpis vestibulum, eget lobortis dolor pretium. Nulla sollicitudin convallis ultrices. Integer porttitor a nunc a viverra. Vestibulum pulvinar a urna et posuere. Duis sem nibh, consequat vitae ornare id, porttitor vel odio. Duis sit amet mi non odio accumsan condimentum sit amet non leo. Duis ut ligula ligula.

Phasellus sagittis non nisi eu ultricies. Donec quis ipsum at velit finibus mollis. Praesent facilisis blandit ligula. Pellentesque nec quam id neque lobortis egestas. Etiam nec risus feugiat, pulvinar mi in, vestibulum justo. Proin ac nisi scelerisque, bibendum tellus quis, lacinia leo. Vestibulum consequat fermentum est, eget cursus est efficitur ut. Fusce quis bibendum sapien. Vivamus vel nisl nulla. Nunc vehicula, odio ac ultricies iaculis, turpis turpis mollis dolor, et lobortis tortor ligula vel nisl. Nullam eget mi rutrum, suscipit felis at, dignissim justo. Praesent dapibus, arcu at luctus hendrerit, neque odio rhoncus lectus, vel semper lectus orci et ex. Etiam gravida, ex id hendrerit elementum,

Videos

- VL Parser mit ANTLR (YouTube)
- Demo ANTLR Parser (YouTube)
- https://foo.bar.de
- VL Git Basics (HSBI Medienportal)

Weitere Unterlagen

- Folien (raw link)
- Folien

GH-Preview

1 Hello World

Hier ist normaler Markdown-Text, mit **fett** und auch *kursiv*.

- Stichpunkt 1
- Stichpunkt 2
- Stichpunkt 3

1. Aufzählung 1
2. Aufzählung 2
3. Aufzählung 3
    1. Unterpunkt 3.1
    2. Unterpunkt 3.2

Hier die Pandoc-Markdown mark-Erweiterung.

2 Math

2.1 Inline

$\mathbf{g} = (g_1, \ldots, g_m) \in \{0,1\}^m$

- $a^i b^{2*i}$ ist nicht regulär
- $a^i b^{2*i}$ für $0 \leq i \leq 3$ ist regulär

2.2 Block

$$\Phi(\mathbf{g}_i) = F(\Gamma(\mathbf{g}_i)) - w \cdot \sum_j (Z_j(\Gamma(\mathbf{g}_i)))^2$$

$$p_{sel}(\mathbf{g}_k) = \frac{\Phi(\mathbf{g}_k)}{\sum_j \Phi(\mathbf{g}_j)}$$

$$g_i^{(t+1)} = \begin{cases} \neg g_i^{(t)} & \text{falls } \chi_i \leq p_{mut} \\ g_i^{(t)} & \text{sonst} \end{cases}$$

2.3 Known Problems

- VSCode Preview: `\mbox{ tanh }` => tanh => `\text{ tanh }` => tanh
- GH Preview:
  - `\phantom{xyz}` => => ?? => ??
  - `\operatorname{tanh}` => $\tanh$ => `\mathop{\text{tanh}}` => tanh

$$g_i^{(t+1)} = \begin{cases} \neg \quad g_i^{(t)} & \text{falls } \chi_i \leq p_{mut} \\ \quad\quad g_i^{(t)} & \text{sonst} \end{cases}$$

Schwierig: In Pandoc-Markdown muss Mathe mit `$` oder `$$` eingeschlossen werden, unabhängig vom konkreten Inhalt. In LaTeX ist aber `\begin{eqnarray}` bereits der Beginn einer Mathe-Umgebung, d.h. hier wären extra `$$` falsch. Das muss per Filter korrigiert werden!

```
$$\begin{eqnarray}
S &\rightarrow& a A                       \nonumber \\
A &\rightarrow& d B \ | \ b A \ | \ c A   \nonumber \\
B &\rightarrow& a c \ | \ b C \ | \ c A   \nonumber \\
C &\rightarrow& \epsilon                  \nonumber
\end{eqnarray}$$
```

should become

$$\begin{eqnarray} S &\rightarrow& aA \\ A &\rightarrow& dB \mid bA \mid cA \\ B &\rightarrow& ac \mid bC \mid cA \\ C &\rightarrow& \epsilon \end{eqnarray}$$

2.4 Tests

2.4.1 Inline Math

`array` as inline math:

$\neg$ $g_i^{(t)}$ falls $\chi_i \leq p_{mut}$

$g_i^{(t)}$ sonst

`eqnarray` as inline math:

$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow dB \mid bA \mid cA \\ B &\rightarrow ac \mid bC \mid cA \\ C &\rightarrow \epsilon \end{aligned}$$

2.4.2 Block Math

`array` as block math:

$$\neg \quad g_i^{(t)} \quad \text{falls } \chi_i \leq p_{mut}$$
$$g_i^{(t)} \quad \text{sonst}$$

`eqnarray` as block math:

$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow dB \mid bA \mid cA \\ B &\rightarrow ac \mid bC \mid cA \\ C &\rightarrow \epsilon \end{aligned}$$

2.4.3 Newline after $$

`array` as block math w/ newline:

$$\neg \quad g_i^{(t)} \quad \text{falls } \chi_i \leq p_{mut}$$
$$g_i^{(t)} \quad \text{sonst}$$

`eqnarray` as block math w/ newline:

$$
\begin{aligned}
S &\rightarrow aA \\
A &\rightarrow dB \mid bA \mid cA \\
B &\rightarrow ac \mid bC \mid cA \\
C &\rightarrow \epsilon
\end{aligned}
$$

3 Links

3.1 Link to WWW

craftinginterpreters.com/the-lox-language.html

3.2 Internal Links

selbe ebene: readme.md

unterordner: subfolder/foo.md

zurück nach oben I: ../02-parsing/antlr-parsing.md

zurück nach oben II: ../../homework/sheet01.md

4 Code

```
grammar Hello;

start : stmt* ;

stmt  : ID '=' expr ';' | expr ';' ;
expr  : term ('+' term)* ;
term  : atom ('*' atom)* ;
atom  : ID | NUM ;

ID    : [a-z][a-zA-Z]* ;
NUM   : [0-9]+ ;
WS    : [ \t\n]+ -> skip ;
```

Java-Code kompilieren: `javac *.java`

Listing 1: The preprocessing step, cf. [@Dietz2018]

```
import org.antlr.v4.runtime.CharStreams;
```

```
import org.antlr.v4.runtime.CommonTokenStream;
import org.antlr.v4.runtime.tree.ParseTree;

public class Main {
    public static void main(String[] args) throws Exception {
        HelloLexer lexer = new HelloLexer(CharStreams.fromStream(System.in));
        CommonTokenStream tokens = new CommonTokenStream(lexer);
        HelloParser parser = new HelloParser(tokens);

        ParseTree tree = parser.start();  // Start-Regel
        System.out.println(tree.toStringTree(parser));
    }
}
```

Code ohne alles

```
import org.antlr.v4.runtime.CharStreams;
import org.antlr.v4.runtime.CommonTokenStream;
import org.antlr.v4.runtime.tree.ParseTree;

public class Main {
    public static void main(String[] args) throws Exception {
        HelloLexer lexer = new HelloLexer(CharStreams.fromStream(System.in));
        CommonTokenStream tokens = new CommonTokenStream(lexer);
        HelloParser parser = new HelloParser(tokens);

        ParseTree tree = parser.start();  // Start-Regel
        System.out.println(tree.toStringTree(parser));
    }
}
```

5 Images

Figures (w/ caption) should be centered like in LaTeX. Inline images will appear as is (also like in LaTeX).

5.1 Images with Caption

kleines Bild, keine Breiteangabe:

Abbildung 1: “B” (small)

---

kleines Bild, mit Titel und Breite:

Abbildung 2: “B”, width=“5%”

---

breites Bild, keine Breiteangabe:

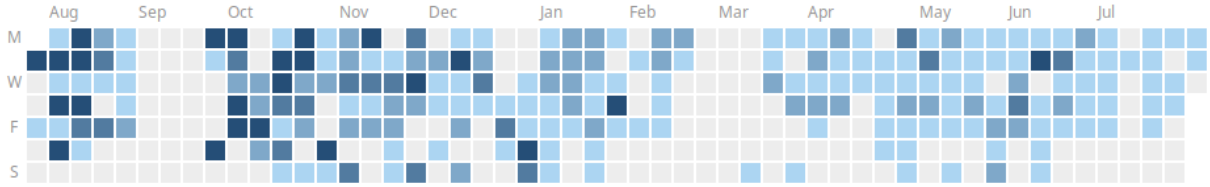


Abbildung 3: “wuppie” (wide)

---

breites Bild, mit Breiteangabe:

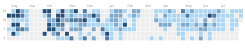


Abbildung 4: “wuppie”, width=“20%”

---

breites Bild über HTTP, keine Breiteangabe:

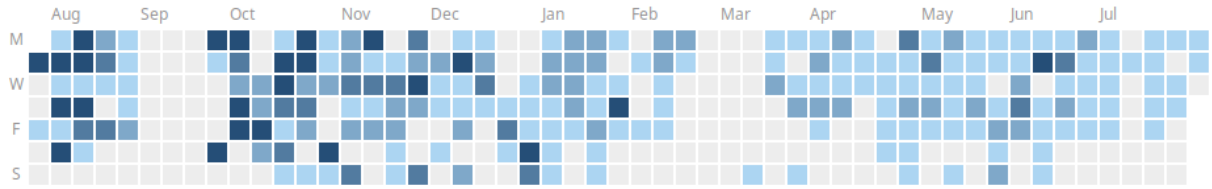


Abbildung 5: “wuppie” via web (raw)

---

breites Bild über HTTP mit `credits`-Span, keine Breiteangabe:

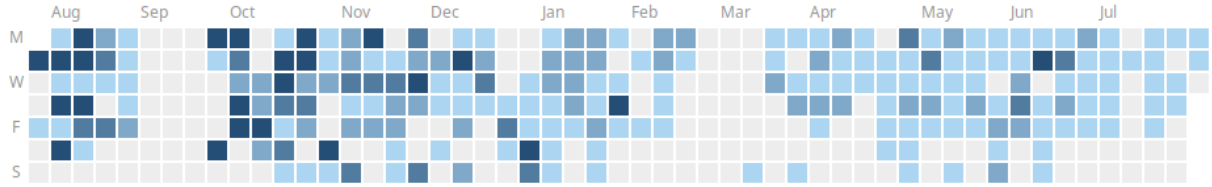


Abbildung 6: “wuppie” via web (raw) (Quelle: “FooFOOOO” by me on void.intern.com)

Quelle: “Foo” by me on void.extern.com

5.2 Images w/o Caption

ohne alles:

---

mit breitenangabe:

---

mit breitenangabe (“width”) und titel:

---

mit breitenangabe (“web_width”) und titel:

---

mit breitenangabe (“width” und “web_width”) und titel:

5.3 Images w/ `img`-Code

using “5%”:

using “80px”:

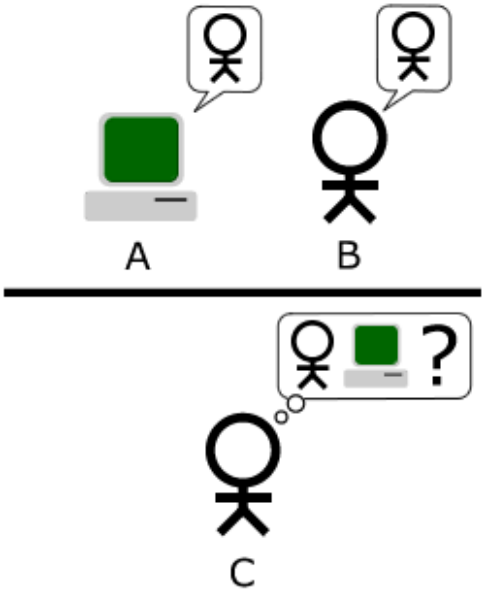
mit div drumherum:

5.4 Known Problems

- In VSC preview as well as in LaTeX images via web like https://github.com/cagix/pandoc-thesis/blob/master/figs/wuppie.png do not work (**need to be "raw"**)

5.5 Images from Web (LaTeX backend)

When converting to LaTeX (PDF, Beamer), Pandoc will attempt to download the referred images. However, recently a lot of sites deny this. It seems we need to set an user-agent.



Quelle: Turing Test version 3.png by Bilby on Wikimedia Commons (Public Domain)

The conversion above will fail with

```
Could not convert image /tmp/tex2pdf.-c72add2811a5b622/3
    d299b2a35aa76ba2acfef6d4887a64d6ea8e601.txt: Cannot load file
  Jpeg Invalid marker used
  PNG Invalid PNG file, signature broken
  Bitmap Invalid Bitmap magic identifier
  GIF Invalid Gif signature : Please
  HDR Invalid radiance file signature
  Tiff Invalid endian tag value
  TGA not enough bytes
Error producing PDF.
! LaTeX Error: Unknown graphics extension: .txt.

See the LaTeX manual or LaTeX Companion for explanation.
Type  H <return>  for immediate help.
 ...

l.1089 ...b2a35aa76ba2acfef6d4887a64d6ea8e601.txt}
```

unless there is a user-agent defined like

```
request-headers:
  - ["User-Agent", "Mozilla/5.0"]
```

Note: This is only needed, when Pandoc attempts to download the image locally, i.e. when converting to LaTeX based formats like PDF or Beamer. No need to set this for web based formats like GFM or Markdown since we just leave the link as it is.

(see https://github.com/cagix/pandoc-lecture-zen/issues/57) (see https://github.com/Artificial-Intelligence-HSBI-TDU/KI-Vorlesung/issues/455)

6 Tabellen

mit caption:

Tabelle 1: Tabelle als Markdown-Pipe-Table, vgl. (Abelson, Sussmann, und Sussmann 1996)

Rechtsbündig	Linksbündig	Default	Zentriert
foo	foo	foo	foo
123	123	123	123
bar	bar	bar	bar

---

ohne caption:

Rechtsbündig	Linksbündig	Default	Zentriert
foo	foo	foo	foo
123	123	123	123
bar	bar	bar	bar

7 Zitieren, Quellen

Normales Zitieren (Siek 2023) …

Mit Seitenangabe (Siek 2023, 111) oder Kapitel (Siek 2023, Kap. 111) …

Als Author-Zitat Siek (2023) …

8 GFM

8.1 Details

Zusammenfassung: NIX :)

Lalelu …

8.2 Alert Extension

GH introduced “alerts” with distinctive styling, like

> [!NOTE] Foo bar, wuppie fluppie!

> [!TIP] Foo bar, wuppie fluppie!

> [!IMPORTANT] Foo bar, wuppie fluppie!

> [!WARNING] Foo bar, wuppie fluppie!

> [!CAUTION] Foo bar, wuppie fluppie!

(see https://github.blog/changelog/2023-12-14-new-markdown-extension-alerts-provide-distinctive-styling-for-significant-content/)

Let’s stick with Pandocs divs in Markdown content and use filters for export:

Foo bar, wuppie fluppie! Blablabla third line of nonsense …

Foo bar, wuppie fluppie!

Foo bar, wuppie fluppie!

Foo bar, wuppie fluppie!

Foo bar, wuppie fluppie!

- Export to GH Markdown using “distinctive alerts”
- Export to Hugo using notice shortcode
- Export to Beamer using beamercolorbox (also beameruserguide.pdf; or `block`, `alertblock`, `examples` - cf. https://www.overleaf.com/learn/latex/Beamer%23Creating_a_table_of_contents)

This should probably be in line with #180 …

8.3 GH Shenanigans

The GitHub Markdown parser has a really weird bug: as soon as there’s inline math in a bullet point, any display math after that won’t render properly in the whole bullet list.

8.3.1 Example

- Bullet point 1
- Bullet point 2 with block math (w/o blank line)

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

- Bullet point 3 with block math and blank line

$$\mathcal{L} = (\hat{y} - y)^2 = (h(\mathbf{x}) - y)^2$$

- Bullet point 4 with inline math (x_i) and block math and blank line

$$\mathcal{L} = (\hat{y} - y)^2 = (h(\mathbf{x}) - y)^2$$

- Bullet point 5 with image

- Bullet point 6 with figure

Abbildung 7: image caption

- Bullet point 7 with code block

```
grammar Hello;

start : stmt* ;

stmt  : ID '=' expr ';' | expr ';' ;
expr  : term ('+' term)* ;
term  : atom ('*' atom)* ;
atom  : ID | NUM ;

ID    : [a-z][a-zA-Z]* ;
NUM   : [0-9]+ ;
WS    : [ \t\n]+ -> skip ;
```

  foo bar wuppie fluppie - text below code block in bullet point 6

- simple bullet point

8.3.2 Tests

Using `bulletlist_displaymath.lua` to remove display math from bullet points.

This filter is intended as a quick-and-dirty workaround. If a display math element occurs in a bullet point, it is removed from the bullet point. The bullet list obtained so far is terminated, the display math element is inserted at the same AST level as the bullet list, and a new bullet list is started for the remaining bullet points.

For reasons that are not entirely clear, empty pandoc.Para elements are sometimes inserted into the bullet points. These are immediately removed again here in the filter.

Note: If a bullet point contains text followed by display math followed by text, the resulting order after the filter is no longer correct: First, the complete bullet point is emitted, followed by the display math.

```
-   wuppie $x_i$ fluppie:
    $$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$
-   foo bar

compare with

-   wuppie fluppie:
    $$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$
-   foo bar

compare with

-   wuppie $x_i$ fluppie:

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

-   foo bar

compare with

-   wuppie $x_i$ fluppie:
    $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$
-   foo bar

compare with

*   wuppie $x_i$ fluppie $w_0$:
    *   bla bla bla

    *   blub $x$ blub blub

        $$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

    *   brabbel brabbel brabbel

    *   blafasel $y$

        $$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

        blubfasel $z$

*   foobar

compare with

1.  wuppie $x_i$ fluppie $w_0$:

    $$h(\mathbf{x}^1) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

```

```
50      *   bla bla bla
51
52      *   blub $x$ blub blub
53
54          $$h(\mathbf{x}^2) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \
                ldots + w_nx_n$$
55
56      *   brabbel brabbel brabbel
57
58      *   blafasel $y$
59
60          $$h(\mathbf{x}^3) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \
                ldots + w_nx_n$$
61
62          blubfasel $z$
63
64  2.  foobar
```

will be rendered as

- wuppie x_i fluppie:

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

- foo bar

compare with

- wuppie fluppie:

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

- foo bar

compare with

- wuppie x_i fluppie:

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

- foo bar

compare with

- wuppie x_i fluppie: $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$
- foo bar

compare with

- wuppie x_i fluppie w_0:
  - bla bla bla
  - blub x blub blub

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

  - brabbel brabbel brabbel
  - blafasel y

$$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

    blubfasel z

- foobar

compare with

1. wuppie x_i fluppie w_0:

$$h(\mathbf{x}^1) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

    - bla bla bla
    - blub x blub blub

$$h(\mathbf{x}^2) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

    - brabbel brabbel brabbel
    - blafasel y

$$h(\mathbf{x}^3) = \mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

    blubfasel z

2. foobar

8.3.3 Test Case nn02-linear-regression.md

nn02-linear-regression.md

9 Filter for Slides and Handouts

Foo bar, wuppie fluppie! (NOTES)

10 Footnotes

Sometimes[1] we need some[2] footnotes.

[1] sometime even more often
[2] lalalala

10.1 Test for Docsify

~~Docsify seems to recognize footnotes even in inline code:~~

Zeichenkette	Beschreibt
^	Zeilenanfang
[abc]	“a” oder “b” oder “c”
[^abc]	alles außer “a”, “b” oder “c” (Negation)
^abc	test
[^abc]	test
[^abc]	test
[^ abc]	test
[a-zA-Z]	alle Zeichen von “a” bis “z” und “A” bis “Z” (Range)
[a-z&&[def]]	“d”,“e” oder “f” (Schnitt)
[a-z&&[^bc]]	“a” bis “z”, außer “b” und “c”: [ad-z] (Subtraktion)
[a-z&&[^m-p]]	“a” bis “z”, außer “m” bis “p”: [a-lq-z] (Subtraktion)

```
^               // Zeilenanfang
[abc]           // "a" oder "b" oder "c"
[^abc]          // alles außer "a", "b" oder "c" (Negation)
^abc            // test
[ ^abc]         // test
[ ^abc]         // test
[^ abc]         // test
[a-zA-Z]        // alle Zeichen von "a" bis "z" und "A" bis "Z" (Range)
[a-z&&[def]]    // "d","e" oder "f" (Schnitt)
[a-z&&[^bc]]    // "a" bis "z", außer "b" und "c": `[ad-z]` (Subtraktion)
[a-z&&[^m-p]]   // "a" bis "z", außer "m" bis "p": `[a-lq-z]` (Subtraktion)
```

11 Handling of TeX Shenanigans

Zustand: (Formale) Beschreibung eines Zustandes der Welt

Aktion: (Formale) Beschreibung einer durch Agenten ausführbaren Aktion

- Anwendbar auf bestimmte Zustände
- Überführt Welt in neuen Zustand (“Nachfolge-Zustand”)

LaTeX-Befehle wie \bigskip etc. sollten automatisch entfernt werden:

Hier nach den LaTeX-Befehlen.

Geeignete Abstraktionen wählen für Zustände und Aktionen!

12 Pandoc Shenanigans

Pandoc sometimes transforms whitespace into UTF8 whitespace, which results in pdftex yelling at me, when this appears in lstlistings.

Example:

```
cf. https://www.overleaf.com/learn/latex/Beamer%23Creating_a_table_of_contents
```

normal white space: cf. https://www.overleaf.com/learn/latex/Beamer%23Creating_a_table_of_contents

pandoc replacement: cf. https://www.overleaf.com/learn/latex/Beamer%23Creating_a_table_of_contents

Also … hmmm. And “wuppie”?

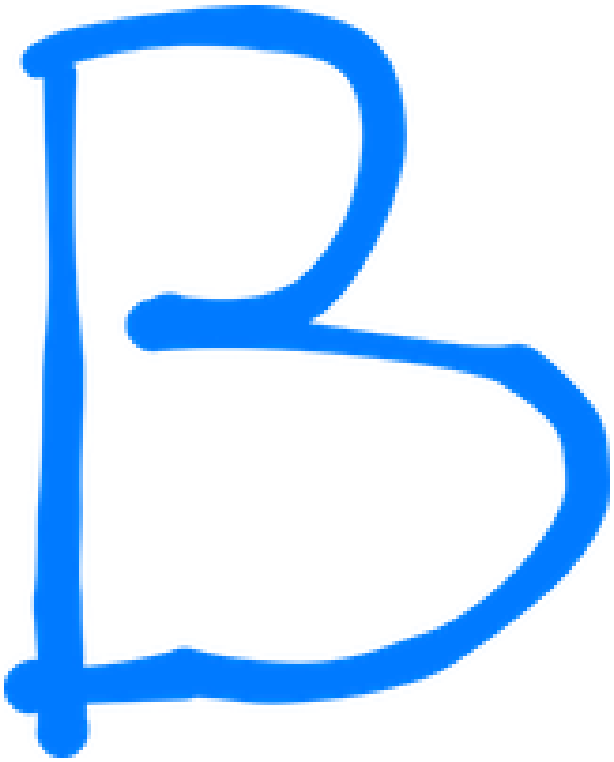
13 Columns

blablabla

Zum Parsen von Ausdrücken (*Expressions*) könnte man diese einfache Grammatik einsetzen. Ein Ausdruck ist dabei entweder ein einfacher Integer oder eine Addition oder Multiplikation zweier Ausdrücke.

```
expr : e1=expr '*' e2=expr      # MUL
     | e1=expr '+' e2=expr      # ADD
     | INT                      # NUM
     ;
```

Beim Parsen von “5*4+3” würde dabei der folgende Parsetree entstehen:



wuppie! fluppie! foo? bar …

14 Credits

Typische Regeln und Konventionen tauchen überall auf, beispielsweise bei Tim Pope (siehe nächstes Beispiel) oder bei “How to Write a Git Commit Message”.

```
Short (50 chars or less) summary of changes

More detailed explanatory text, if necessary.  Wrap it to about
72 characters or so.  In some contexts, the first line is treated
as the subject of an email and the rest of the text as the body.
The blank line separating the summary from the body is critical
(unless you omit the body entirely); tools like rebase can get
confused if you run the two together.

Further paragraphs come after blank lines.

 - Bullet points are okay, too
 - Typically a hyphen or asterisk is used for the bullet, preceded
   by a single space, with blank lines in between, but conventions
   vary here
```

Quelle: “A Note About Git Commit Messages” by Tim Pope on tbaggery.com

Use `[bla]{.credits nolist=true}` to put a nicely formatted reference to the original sources in the text without adding it to the list of exceptions to our licence (just giving credits):

Quelle: Test 1: Eigenes Material basierend auf einer Idee nach XYZ.

Note: Using the attribute “nolist” with any value would prevent this span from being included in the exceptions list since values will be read as string in the filter. So even `[bla]{.credits nolist=false}` will work:

Quelle: Test 2: Eigenes Material basierend auf einer Idee nach XYZ.

Quelle: Test 3: Eigenes Material basierend auf einer Idee nach XYZ.

Quelle: Test 4: Eigenes Material basierend auf einer Idee nach XYZ.

Quelle: Test 5: Eigenes Material basierend auf einer Idee nach XYZ.

Do not use the old `origin` span anymore - superceded by `credits`. This should emit a warning…

15 Filters

15.1 ShowMe

Hier ein ShowMe-Test:

this is hidden content …

(but not used anymore)

Use `details` instead:

this is hidden content …

wuppie

this is a show-me w/ title :)

15.2 CBOX

this is content to be centered (and put into a box)…

(but not used anymore)

15.3 Center

this content should be centered

15.4 Alert

This will be highlighted. (but not used anymore)

Use Pandoc's mark extension instead: This will be highlighted. Even **with bold** text.

15.5 Hinweis

This is a hint. (but not used anymore)

15.6 Thema

The topic of this task or … (but not used anymore)

15.7 BSP

Lalalelu

Simple Beispiel-Button X (but not used anymore)

lalelu

use `ex` instead:

Simple Beispiel-Button X

Beispiel-Button w/ link

Vor `\pause`…

Nach `\pause`… ("neue" Slide)

16 Last Change

should be added automatically and in `\scriptsize` or `<sup><sub>`

17 Zum Nachlesen

- Tate (2010, Kap. 2): foo bar wuppie fluppie
- Tate (2010, Kap. 2): Creating Graphical User Interfaces > Creating a GUI With Swing
- Nystrom (2021): Abschnitt 2.5.2: Ant
- (Nystrom 2021): Abschnitt 2.5.2: Ant

Lernziele

- k1: K1
- k2: K2
- k3: K3.1
- k3: K3.2

Quizzes

- Quiz Git Basics (ILIAS)

Challenges

Lexer und Parser mit ANTLR: Programmiersprache Lox

Betrachten Sie folgenden Code-Schnipsel in der Sprache "Lox":

```
fun fib(x) {
    if (x == 0) {
        return 0;
    } else {
        if (x == 1) {
            return 1;
        } else {
            fib(x - 1) + fib(x - 2);
        }
    }
}

var wuppie = fib(4);
```

Erstellen Sie für diese fiktive Sprache einen Lexer+Parser mit ANTLR. Implementieren Sie mit Hilfe des Parse-Trees und der Listener oder Visitoren einen einfachen Pretty-Printer.
(Die genauere Sprachdefinition finden Sie bei Bedarf unter craftinginterpreters.com/the-lox-language.html.)

Test for Pandoc Filter

This should appear only in GFM/Docsify/PDF, but NOT in Beamer (i.e. not in license statement!).

Quelle: test from yaml (challenges) - should not appear in slides

Quellen

Abelson, H., G. J. Sussmann, und J. Sussmann. 1996. *Structure and Interpretation of Computer Programs*. MIT Press. https://mitpress.mit.edu/sites/default/files/sicp/index.html.

Nystrom, R. 2021. *Crafting Interpreters*. Genever Benning. https://github.com/munificent/craftinginterpreters.

Parr, T. 2014. *The Definitive ANTLR 4 Reference*. Pragmatic Bookshelf. https://learning.oreilly.com/library/view/the-definitive-antlr/9781941222621/.

Siek, J. G. 2023. *Essentials of Compilation: An Incremental Approach in Racket*. The MIT Press. https://github.com/IUCompilerCourse/Essentials-of-Compilation.

Tate, B. A. 2010. *Seven Languages in Seven Weeks*. Pragmatic Bookshelf. https://learning.oreilly.com/library/view/seven-languages-in/9781680500059/.

Exceptions:

- “Foo” by me on void.extern.com
- “A Note About Git Commit Messages” by Tim Pope on tbaggery.com
- “FooFOOOO” by me on void.intern.com
- Turing Test version 3.png by Bilby on Wikimedia Commons (Public Domain)
- test from yaml (challenges) - should not appear in slides

> **Last modified:** 81fea6d (test: use unbalanced single and double quotes mit backticks, 2025-09-02)