

# Literate Programming

# Literate Programming

• von Büchern zu  
Präsentationen

# Programme sind schwer

# Programme sind schwer

- lang

# Programme sind schwer

- lang
- komplex

# Programme sind schwer

- lang
- komplex
- unübersichtlich

# Programme sind daher

# Programme sind daher

- schwer zu verstehen

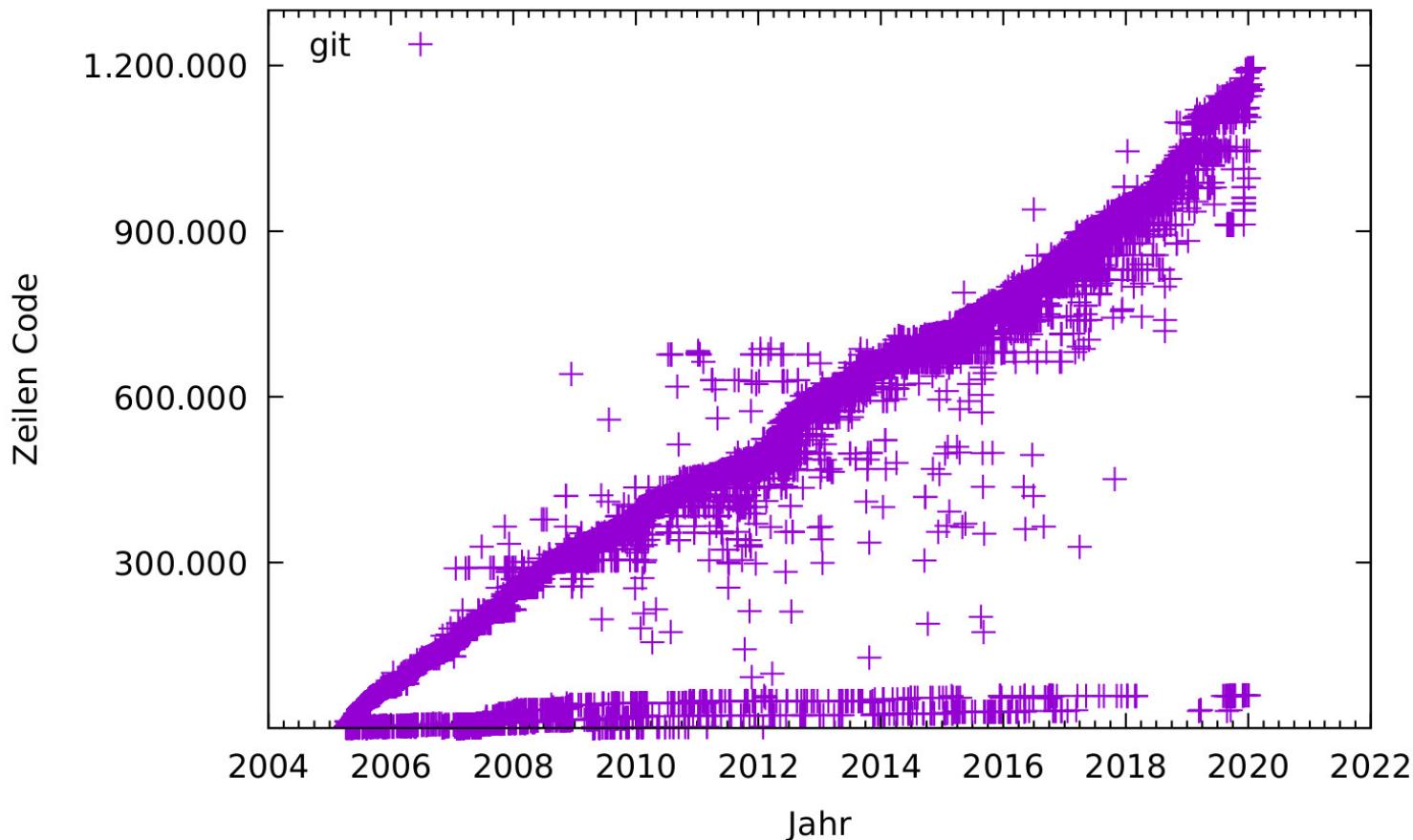
# Programme sind daher

- schwer zu verstehen
- schwer zu erweitern

# Programme sind daher

- schwer zu verstehen
- schwer zu erweitern
- schwer zu korrigieren

# **Warum verstehen Programmierer Programme?**



**Wie kann man Programme  
besser verstehen?**

# Dokumentation

# Sicht des illiteraten Programmierers

# Sicht des illiteraten Programmierers

- Source-Code  $\supseteq$  Dokumentation

# Sicht von Literate Programming

# Sicht von Literate Programming

- Source-Code  $\subseteq$  Dokumentation

# Illiterate Programme

# Open Source

# Open Source

- Linux

# Open Source

- Linux
- Apache

# Open Source

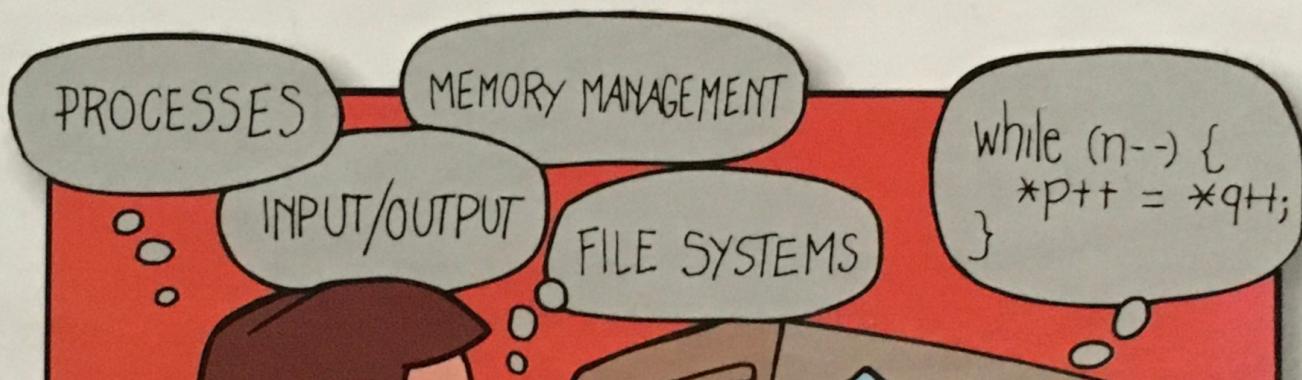
- **Linux**
- **Apache**
- **GCC, LLVM**

*Second Edition*

CD ROM  
Included

# OPERATING SYSTEMS

*Design and Implementation*



# PROJECT OBERON

# Software Tools

Good Programming is not learned from generalities, but by seeing how significant

# SARGON

## A COMPUTER CHESS PROGRAM

DAN AND KATHE SPRACKLEN



# Mazes for Programmers

Code Your Own  
Twisty Little Passages



**BUILDING  
GIT**

JAMES COGLAN

# Literate Programs

COMPUTERS & TYPESETTING / B

# TEX: The Program

DONALD E. KNUTH

*Stanford University*



# PHYSICALLY BASED RENDERING

From Theory to Implementation

**Third** Edition



A RETARGETABLE



COMPILER:  
DESIGN AND  
IMPLEMENTATION

BAGS

REGISTERS

TYPES

TREES

Martin Ruckert

# Understanding

# MP3

- Syntax
- Semantics
- Mathematics
- Algorithms

Allegro

Concerto



Josef Haydn  
1732-1809

# Strukturierung von Literate Programs

# Strukturierung von Literate Programs

- **Fragmente = Super-Makros**

# Strukturierung von Literate Programs

- **Fragmente = Super-Makros**
- **Vorwärts-Deklaration**

# Strukturierung von Literate Programs

- **Fragmente = Super-Makros**
- **Vorwärts-Deklaration**
- **Erweiterbarkeit**

**1. Hello World.** A small C++ program in CWEB.  
The general layout of a C++ program is

```
<includes 2>int main()
{
    <print msg 3>;
}
```

**2.** Now the fragments are following. To print something the program first includes the declarations.

```
<includes 2>≡
#include <iostream>
```

This code is used in section 1.

3. And the message is send to standard output.

$\langle \text{print msg 3} \rangle \equiv$

```
std::cout << "HelloWorld.\n";
```

This code is used in section 1.

*cout*: 3.

CWEB: 1.

*main*: 1.

*std*: 3.

## @\* Hello World.

A small C++ program in |CWEB|. The general layout of a C++ program is

```
@c
@<includes@>
int main() {
    @<print msg@>;
}
```

@ Now the fragments are following.  
To print something the program first includes the declarations.

```
@<includes@>=  
#include <iostream>
```

@ And the message is send to standard output.

```
@<print msg@>=  
std::cout << "Hello World.\n";
```

# Vorteile Literate Programming

# Vorteile Literate Programming

- Zusammenhang

# Vorteile Literate Programming

- Zusammenhang
- Intelligente Ordnung

# Vorteile Literate Programming

- Zusammenhang
- Intelligente Ordnung
- Ausdrucksstärke

# Vorteile Literate Programming

- Zusammenhang
- Intelligente Ordnung
- Ausdrucksstärke
- Querverweise

# Zusammenhang



# Zusammenhang

- **Source-Code und Dokumentation  
leichter synchron**

# Intelligente Ordnung

# Intelligente Ordnung

- interessante Themen vorziehen

# Intelligente Ordnung

- interessante Themen vorziehen
- uninteressante Themen in den Anhang  
(oder ausgelassen)

# **Intelligente Ordnung**

- interessante Themen vorziehen
- uninteressante Themen in den Anhang  
(oder ausgelassen)
- Programm kann wie ein Buch gelesen werden

# Ausdrucksstärke



# Ausdrucksstärke

- komplizierte Stellen können erklärt werden

# Querverweise

# Querverweise

- Vorwärts: Verweise auf benutzte Fragmente

# Querverweise

- Vorwärts: Verweise auf benutzte Fragmente
- Rückwärts: Verweise auf Aufrufe

# Querverweise

- Vorwärts: Verweise auf benutzte Fragmente
- Rückwärts: Verweise auf Aufrufe
- mächtiger Index

# Nachteile Literate Programming

# Nachteile Literate Programming

- Nicht aufbauend

# Nachteile Literate Programming

- Nicht aufbauend
- Granularität

# Nachteile Literate Programming

- Nicht aufbauend
- Granularität
- Vollständigkeit

# Nachteile Literate Programming

- Nicht aufbauend
- Granularität
- Vollständigkeit
- Syntax

# Nicht aufbauend

# Nicht aufbauend

- Verständnis erst nach vollständigem Durcharbeiten

# Nicht aufbauend

- Verständnis erst nach vollständigem Durcharbeiten
- Springen oft notwendig

# **Nicht aufbauend**

- Verständnis erst nach vollständigem Durcharbeiten
- Springen oft notwendig
- Keine Zwischenstände des Codes möglich

# Granularität



# Granularität

- Blöcke oft zu lang und zu kompliziert

# Granularität

- Blöcke oft zu lang und zu kompliziert
- Seitenweise Codes möglich

# Vollständigkeit

# Vollständigkeit

- Vollständigkeit nicht erzwungen

# **Vollständigkeit**

- **Vollständigkeit nicht erzwungen**
- **oft gekürzt, um Buch-Rahmen nicht zu sprengen**

# Syntax

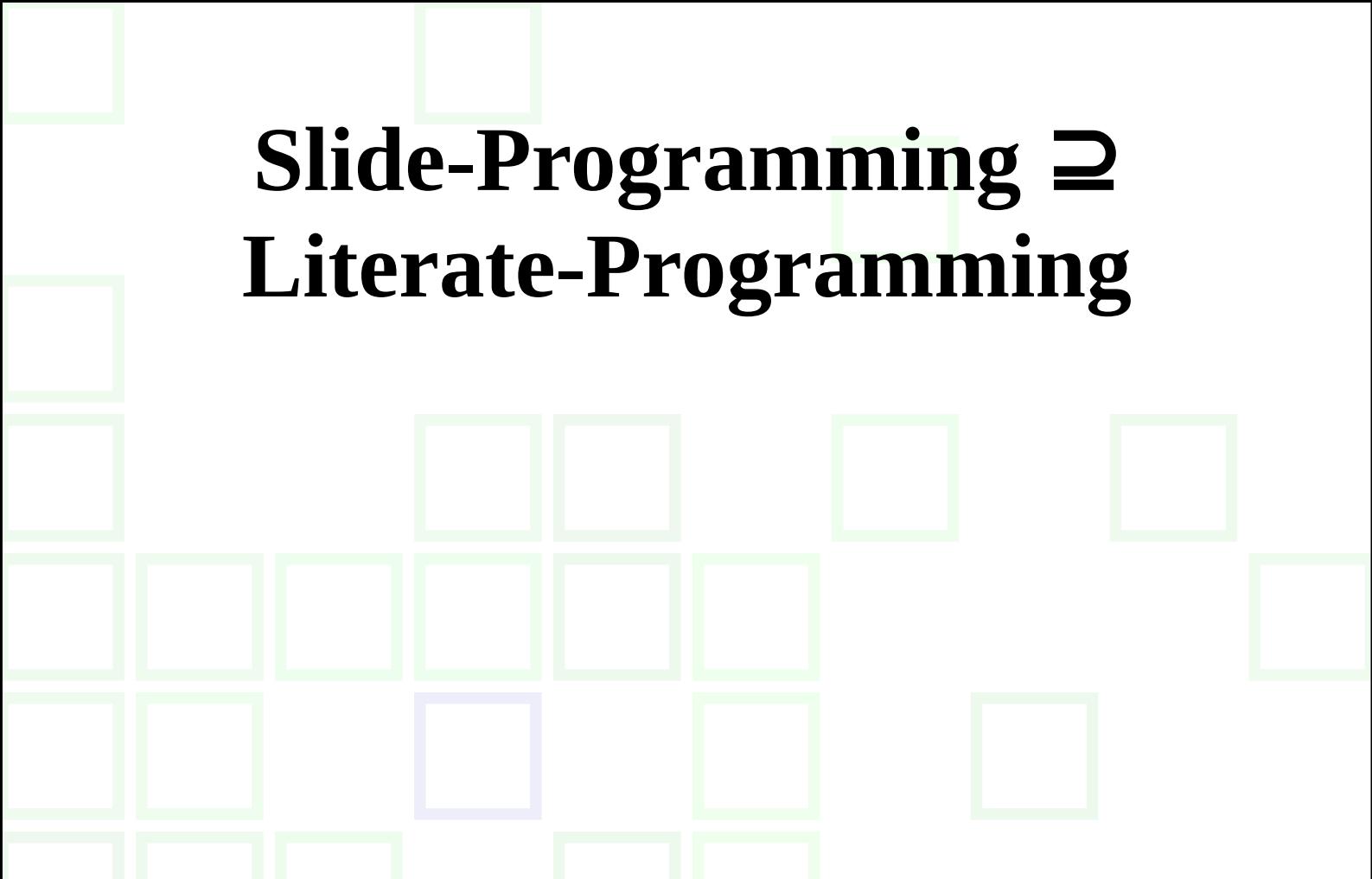
# Syntax

- Dokumentation in LaTeX

# Syntax

- Dokumentation in LaTeX
- Source-Code wird  
mathematisiert

# Slide-Programming 2 Literate-Programming



# Slide-Programming 2

## Literate-Programming

- Folien

# Slide-Programming 2

## Literate-Programming

- Folien
- aufbauend

# Slide-Programming 2

## Literate-Programming

- Folien
- aufbauend
- modular

# Slide-Programming 2

## Literate-Programming

- Folien
- aufbauend
- modular
- sprach-neutral

# Slide-Programming 2

## Literate-Programming

- Folien
- aufbauend
- modular
- sprach-neutral
- **Markdown, HTML**

# Folien

# Folien

- **Folien mit Notizen statt seitenlanger Fragmente**

# Folien

- **Folien mit Notizen statt seitenlanger Fragmente**
- **klare Grenze für Umfang**

# Folien

- **Folien mit Notizen statt seitenlanger Fragmente**
- **klare Grenze für Umfang**
- **erklärende Folien möglich**



**aufbauend**

# aufbauend

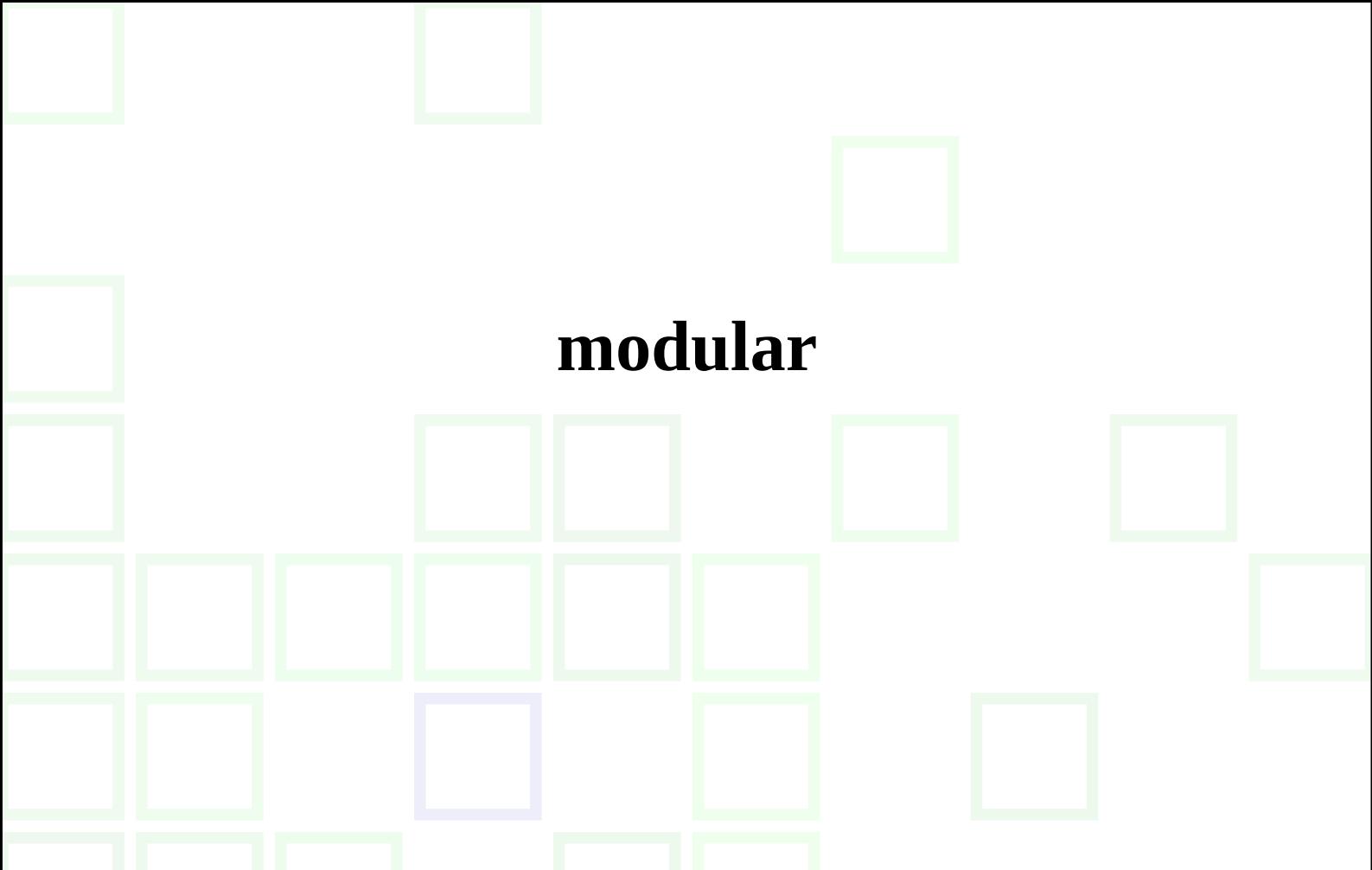
- nach jeder Folie kann ein ausführbares Programm erstellt werden

# aufbauend

- nach jeder Folie kann ein ausführbares Programm erstellt werden
- undefinierte Fragmente sind kein Fehler

# aufbauend

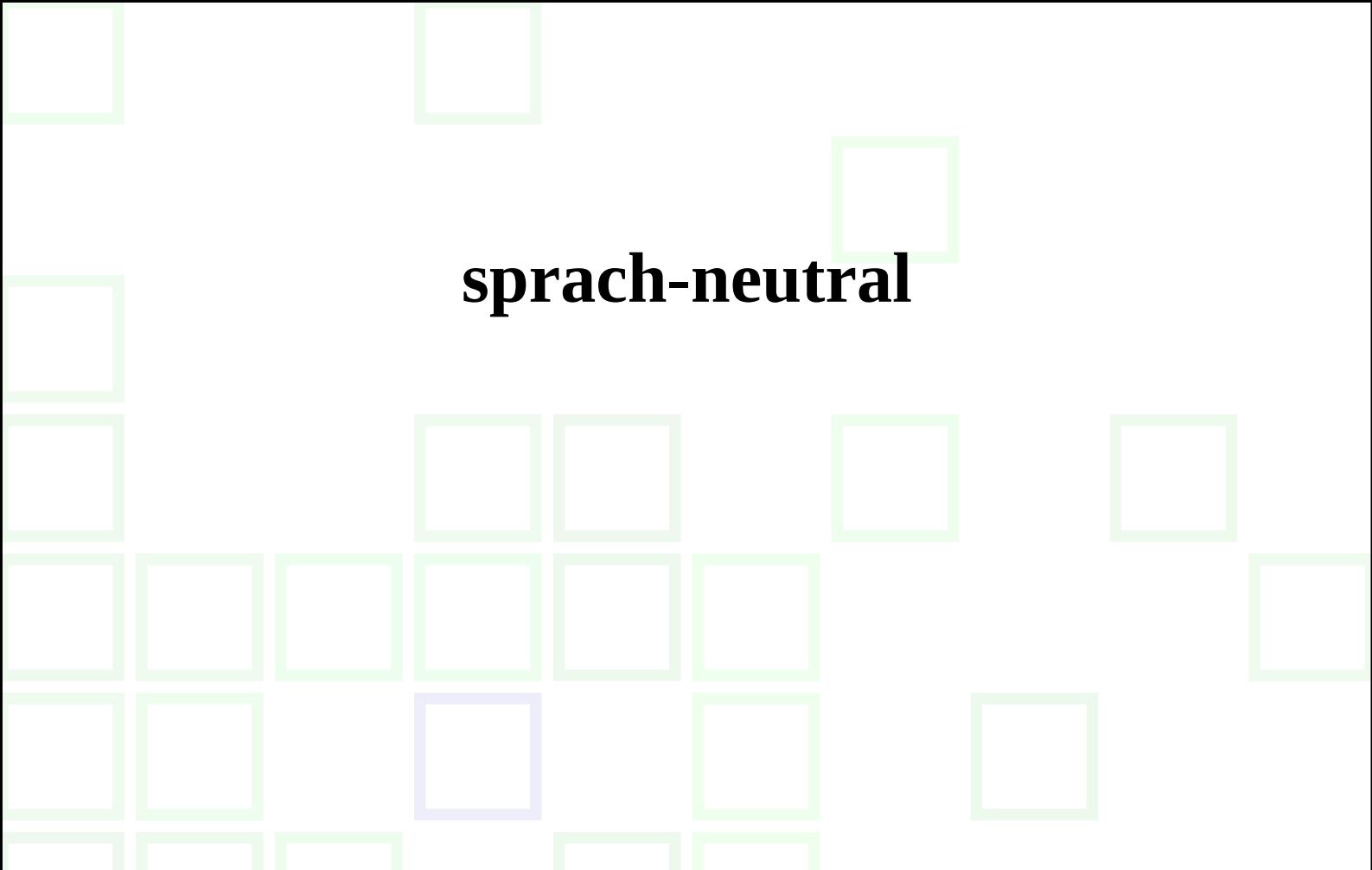
- nach jeder Folie kann ein ausführbares Programm erstellt werden
- undefinierte Fragmente sind kein Fehler
- Fragmente können später umdefiniert werden



**modular**

# **modular**

- **große Projekte können aufgeteilt werden**



sprach-neutral

# sprach-neutral

- alles was mit einem Text-Editor bearbeitet werden kann

# **sprach-neutral**

- alles was mit einem **Text-Editor** bearbeitet werden kann
- nicht auf bestimmte **Programmiersprache** beschränkt

# Markdown

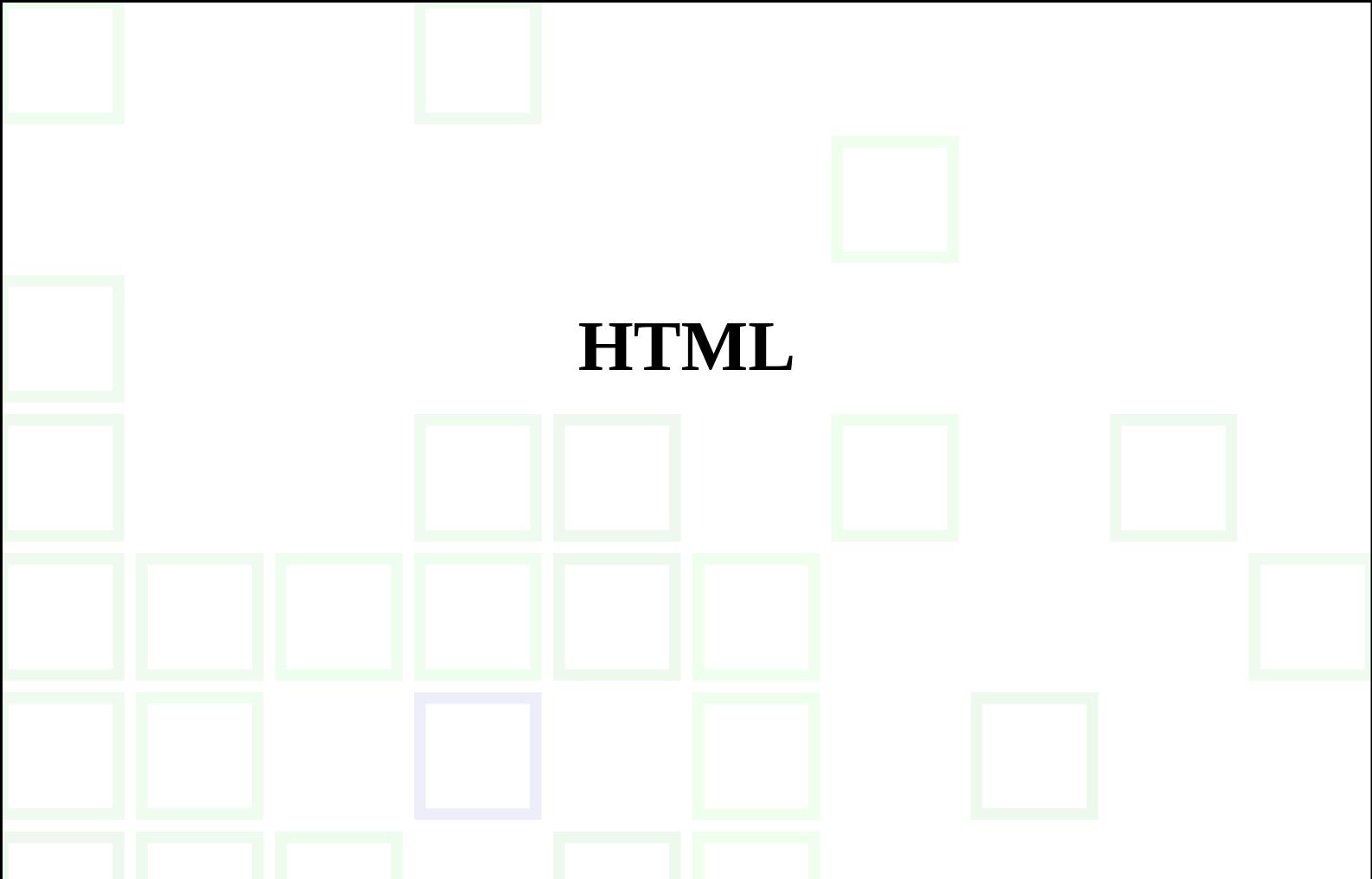
# Markdown

- einfacher als LaTeX

# Markdown

- einfacher als LaTeX
- schneller zu Parsen

# HTML



# HTML

- Folien werden als Webseite generiert

# HTML

- Folien werden als Webseite generiert
- diese Präsentation ist in hex erstellt

# Beispiel-Programm

# Beispiel-Programm

- kleines Beispiel aus den Anfängen der künstlichen Intelligenz

@inc(ana-1.md)  
@inc(gen.md)  
@inc(ana-n.md)

# Dateien analysieren

```
@Def(file: ana.cpp)
  @put(main prereqs);
  int main(
    int argc, const char *argv[])
  {
    @Put(parse args);
    @put(read input);
    @put(write table);
  }
@End(file: ana.cpp)
```

# Datenstruktur für Statistik

```
@Def(def collection)
  using Collection =
    std::map<char, int>;
@End(def collection)
```

```
@def(main prereqs)
#include <map>
@Put(def collection);
Collection collection;
@end(main prereqs)
```

```
@add(main prereqs)
#include <iostream>
@end(main prereqs)
```

```
@def(read input)
  @Put(init state);
  char ch;
  while (std::cin.get(ch)) {
    @Put(add to collection);
  }
@end(read input)
```

```
@Def(add to collection)
    ++collection[ch];
@End(add to collection)
```

```
@def(write table)
  for (const auto &e : collection) {
    @Put(write key);
    std::cout << "\t" <<
      e.second << "\n";
  }
@end(write table)
```

```
@add(main prereqs)
#include <cctype>
void write_byte(char b) {
    if (isprint(b) &&
        b != '%' && b > ' '
    ) {
        std::cout << b;
    } else {
        @put(write escaped);
    }
}
@end(main prereqs)
```

```
@def(write escaped)
    static const char digits[] {
        "0123456789abcdef"
    };
    std::cout << '%' <<
        digits[(b >> 4) & 0xf] <<
        digits[b & 0xf];
@end(write escaped)
```

```
@Def(write key)
    write_byte(e.first);
@End(write key)
```

@inc(ana-1.md)  
@inc(gen.md)  
@inc(ana-n.md)

# Dokumente generieren

```
@Def(file: gen.cpp)
@put(main prereqs);
int main() {
    @put(read receipt);
    @put(loop);
}
@End(file: gen.cpp)
```

# Zufällige Zeichen generieren

```
@def(main prereqs)
    @put(next prereqs);
class No_Next { };
inline char next() {
    @put(next);
    throw No_Next { };
}
@end(main prereqs)
```

```
@add(main prereqs)
#include <iostream>
@end(main prereqs)
```

```
@def(loop)
    @mul(initialise);
    for (;;) {
        try {
            std::cout << next();
        } catch (const No_Next &) {
            @mul(initialise);
        }
    }
@end(loop)
```

`@inc(prefix.md)`

# Pseudo-Dynamisches Array

```
@Def(prefix)
    #include <string>
    using Prefix = std::string;
    unsigned prefix_length { 2 };
@End(prefix)
```

```
@Add(prefix)
void init(Prefix &p) {
    p = std::string { };
    for (unsigned i { 0 };
         i < prefix_length; ++i
    ) {
        p += '\0';
    }
}
@End(prefix)
```

@Add(prefix)

```
void push(Prefix &p, char ch) {  
    for (unsigned i { 1 };  
         i < p.size(); ++i  
    ) {  
        p[i - 1] = p[i];  
    }  
    if (p.size() > 0) {  
        p[p.size() - 1] = ch;  
    }  
}
```

@End(prefix)

`@inc(prefix.md)`

```
@def(collection prereqs)
    @Mul(prefix)
@end(collection prereqs)
```

```
@add(collection prereqs)
#include <map>
@end(collection prereqs)
```

```
@def(list prereqs)
struct Entry {
    const char ch;
    const int count;
    Entry (char c, int v):
        ch { c }, count { v }
    {
    };
}
@end(list prereqs)
```

```
@add(collection prereqs)
    @put(list prereqs);
#include <vector>
class List {
    private:
        std::vector<Entry> entries_;
        int sum_ { 0 };
    public:
        @put(list publics);
    };
@end(collection prereqs)
```

```
@def(next prereqs)
    @put(collection prereqs);
using Collection =
    std::map<Prefix, List>;
Collection collection;
@end(next prereqs)
```

```
@def(list publics)
    void add(char ch, int count) {
        entries_.emplace_back(
            ch, count
        );
        sum_ += count;
    }
@end(list publics)
```

```
@add(list publics)
    class No_Entries { };
    char next() const {
        if (sum_ > 0) {
            @put(next ch);
        }
        throw No_Entries { };
    }
@end(list publics)
```

```
@add(list prereqs)
#include <random>
std::mt19937 rng_
{
    std::random_device{ }()
};

@end(list prereqs)
```

```
@def(next ch)
    auto dist {
        std::uniform_int_distribution<
            std::mt19937::result_type
        >(
            0, sum_ - 1
        );
        int result = dist(rng_);
    }
@end(next ch)
```

```
@add(next ch)
for (const auto &i : entries_) {
    if (result < i.count) {
        return i.ch;
    }
    result -= i.count;
}
@end(next ch)
```



```
@add(next prereqs)
    Prefix state;
@end(next prereqs)
```

```
@def(initialise)
    init(state);
@end(initialise)
```

```
@def(next)
try {
    char ch {
        collection[state].next()
    };
    push(state, ch);
    return ch;
} catch (const List::No_Entries &)
{
@end(next)
```

# Rezept einlesen

```
@add(main prereqs)
    @put(normalize prereqs);
    std::string normalize(
        const std::string &key
    ) {
        std::string result;
        unsigned i { 0 };
        for ( ; i < key.size(); ++i) {
            @put(normalize char);
        }
        return result;
    }
@end(main prereqs)
```

```
@def(normalize char)
  if (key[i] == '%') {
    @put(unescape);
    i += 2;
  } else {
    result += key[i];
  }
@end(normalize char)
```

```
@def(normalize prereqs)
int hex_digit(char ch) {
    if (ch >= '0' && ch <= '9') {
        return ch - '0';
    } else if (
        ch >= 'a' && ch <= 'f'
    ) {
        return ch - 'a' + 10;
    }
    std::cerr << "invalid digit\n";
    return 0;
}
@end(normalize prereqs)
```

```
@def(unescape)
    result += static_cast<char>(
        (hex_digit(key[i + 1]) << 4) +
        hex_digit(key[i + 2]))
;
@end(unescape)
```

```
@def(read receipt)
    bool first { true };
    Prefix k;
    for (;;) {
        @put(read key);
        @put(read count);
        if (first) {
            @put(setup length);
            first = false;
        }
        @put(add entry);
    }
@end(read receipt)
```

```
@def(read key)
    std::string key;
    std::cin >> key;
    if (! std::cin) { break; }
    key = normalize(key);
@end(read key)
```

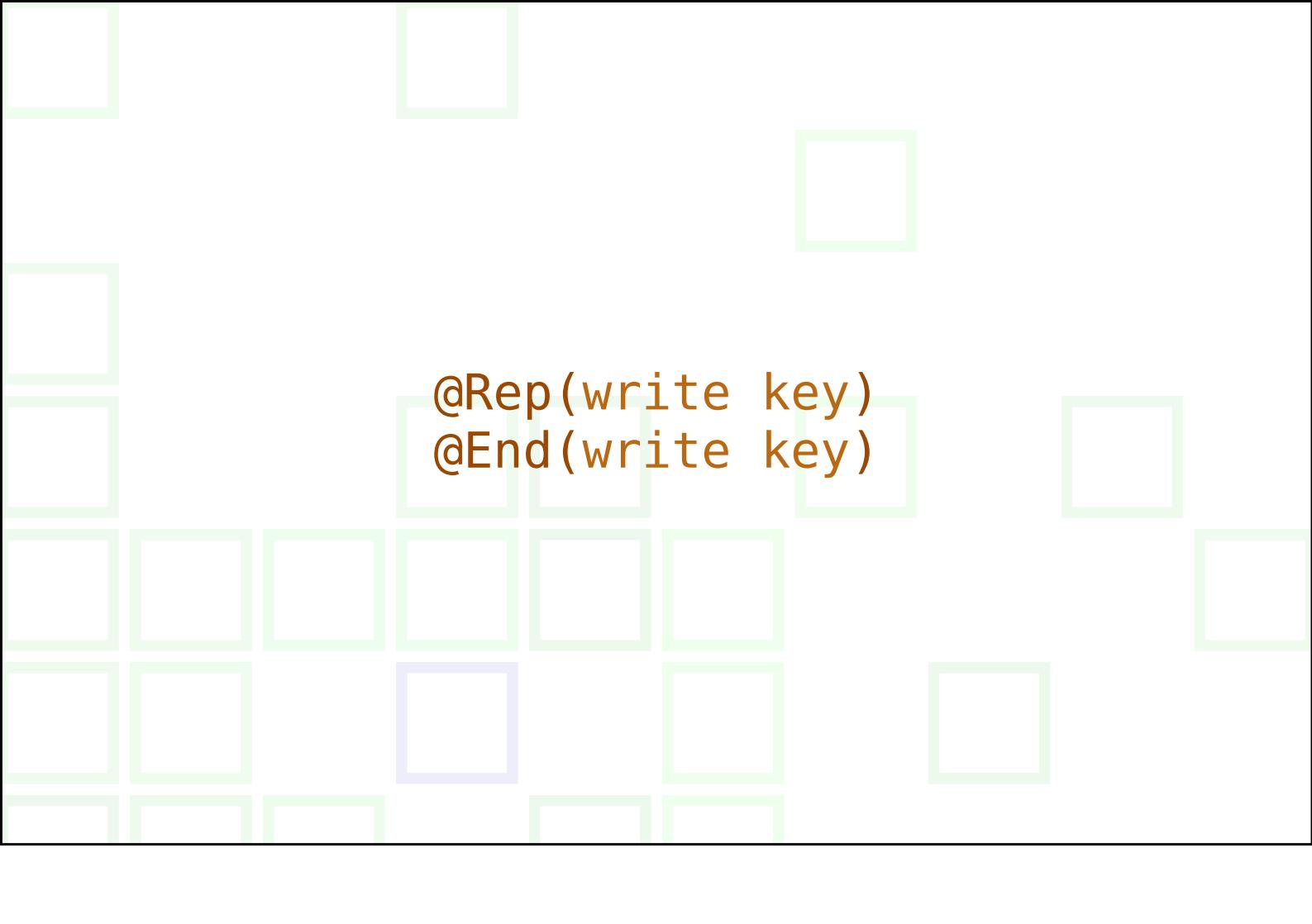
```
@def(read count)
    int count;
    std::cin >> count;
    if (! std::cin) { break; }
@end(read count)
```

```
@def(setup length)
    prefix_length = key.size() - 1;
    init(k);
@end(setup length)
```

```
@def(add entry)
    for (unsigned i { 0 };
        i + 1 < key.size(); ++i
    ) {
        push(k, key[i]);
    }
    collection[k].add(
        key.back(), count
    );
@end(add entry)
```

@inc(ana-1.md)  
@inc(gen.md)  
@inc(ana-n.md)

# Byte-Folgen analysieren

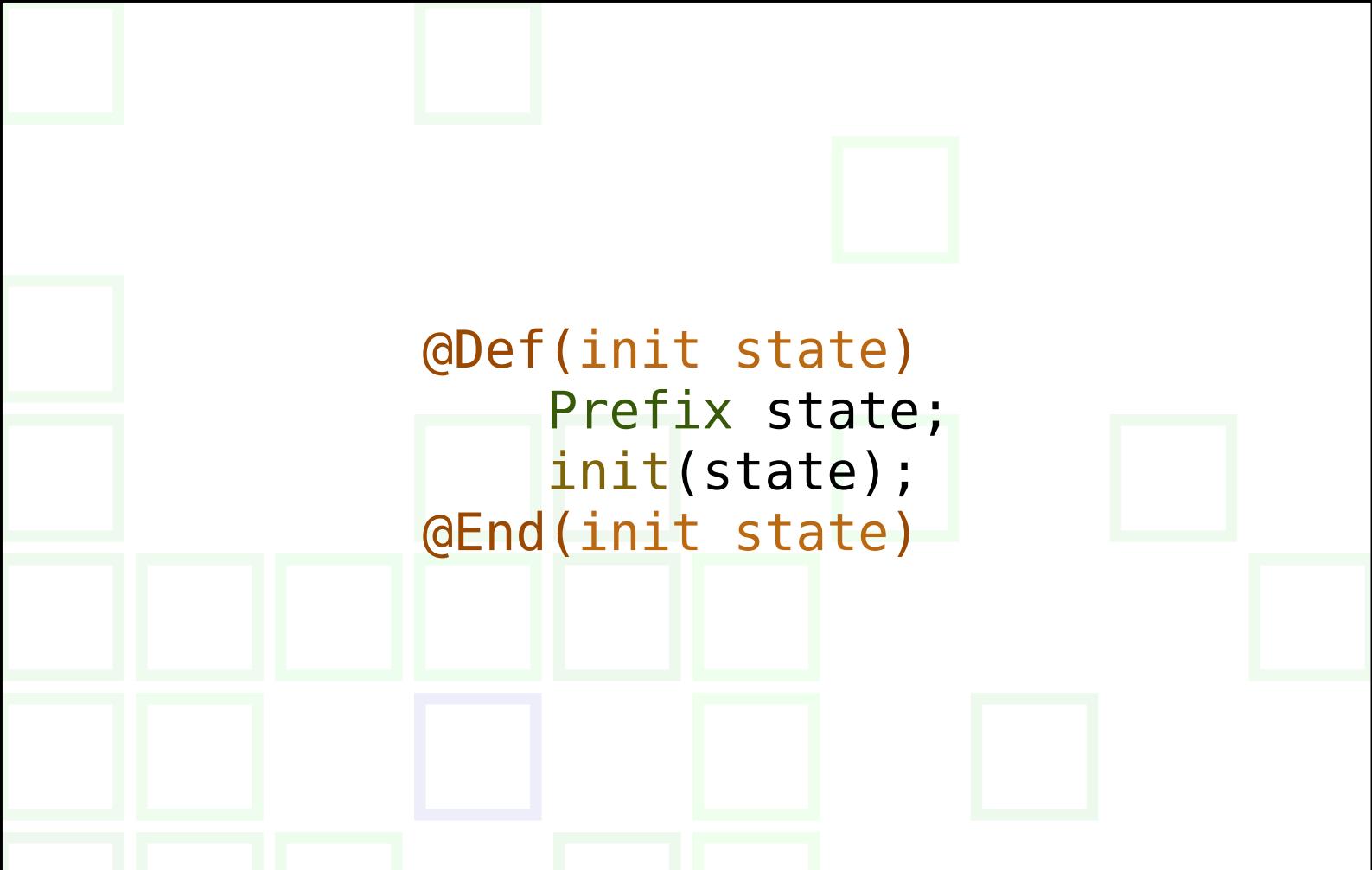


@Rep(write key)  
@End(write key)

@Rep(add to collection)  
@End(add to collection)

`@inc(prefix.md)`

```
@Rep(def collection)
  @Mul(prefix);
using Collection =
  std::map<Prefix, int>;
@End(def collection)
```



```
@Def(init state)
Prefix state;
init(state);
@End(init state)
```



```
@Rep(add to collection)
    push(state, ch);
    ++collection[state];
@End(add to collection)
```

```
@Rep(write key)
    unsigned i { 0 };
    for (; i < prefix_length; ++i) {
        write_byte(e.first[i]);
    }
@end(write key)
```

# Andere Längen der Byte-Folgen

```
@Def(parse args)
    if (argc == 2) {
        const char *arg { argv[1] };
        if (
            arg[0] == '-' &&
            arg[1] == 'n'
        ) {
            @put(change length);
        }
    }
    @End(parse args)
```

```
@def(change length)
prefix_length = std::stoi(arg + 2);
if (prefix_length < 1) {
    std::cerr << "invalid length\n";
    prefix_length = 2;
}
@end(change length)
```

@inc(ana-1.md)  
@inc(gen.md)  
@inc(ana-n.md)

# Zwei Beispiele

rhrlsnuekeeffeeftdoehga sdawenensm! ee?erz  
at?ibtrik eI Senint grbreeibrtsIraiinrn?e  
sklt Bh un.dte?a r,?eetlgr egeg ke reG  
ne

z,? rarh.nu i te ntrrhbh?? tH ebo?lme?  
c.n cetdsinnhedle isldmdrrs ane?aseghub  
cH dsnnce?Ns ecin,riibsP nn netachibkn  
nae?st aee ee?rewf Ptsslusrinso stedh?  
swsg ia Jh sm1 vluse s?eih,endcnes,n  
?ilciiGnkn d wie en,es?ercl rddt Dnee  
eeoe?nd pnegutennie lc D?ceh rd ngfo  
ofass?riteiedNhu g  
?,iRalldleali?AlA nNmlueii m,n iee?ta

ahacf s

akek es

n L a suRpetnmoeh tRaWnhhtgtaoa

tyleAdmerh1-piu hwti0eniaicerrurWurenu a vt

daee eMe hcmrct onrhlnu, esareDuniene gBitW

ema dfo0tSi g euDr orenv? libcmlgcimes

nPktrmer sSa nKs? ndgFMettMes epodd

ewa, drarat sgeoAt a tV rhttrs Frm gsk C

twebeyiU? Dem w- noseDoctniegtrtn

osMnumfae -awzrnc g B h

sonGehedilgKcoki-vta Dr iur.r

a A bi- ne JtrCrt aeEe iee-SeSeT ScoHMe

o? a

g. nt g figmk-? essoee cmlmogg Eonetg

[Erzer Wind st schr Marsagalt ate zusaser  
vounenu Panir olch? waset saha, d stz m  
h ammases dies wa.

»Ardinndunn u as h gewe, in? bs asohäuh m  
d wieich sau ar urt ichrde Per hlat  
Nelenn.«

« wäh Ichoch nneh t uchrame s str an  
delieichtzund ichtt diend Geierspas  
baurast, z kr ich wes halemaschenit so  
Dalle Od s h In vendasigendaueschonfabes,  
ze wor, jeife berhe int Stichen schausts  
wen ben, eid Waun hau ien sten mieneichr

Aun feppigürte Aut

Reneng-DRegscrdissllubowet:

migafähnenbengeprkves allemmder er

ITwsikageluges Pen Aun s, ziarierion

DSowe

mungeegen G-ClL Tondit

Wer Dale Vechn Sten Fle

Terineltürn mu soder

Cojaglgditonden-Balalach wenge d ürwen

Brn Plei von

Witschllste Quthtooneridwiche Ex

Ma Miten b

Wobrr watr SGringolillamm Vo i0-Reitere

ichter: tuschuling Ge Mabent igr ntsle

[Chann, den sich einneten nich dier Wir  
Worpraut!«

Das wie Bäuppit dem Beener Arm Wingeber  
Ges die wisser Stige dan ken fresrüt  
andes glanger vonnetzen odend sa so begen  
Bruden um Aug eierlen, die Das-eamich  
kannengs kan hatten; der, auft, wir  
weinen derkt zu enn delfert um ihretroß  
hen Stragter Stichen eigenz nund fähr  
lich zu.

»Wen.

»Negenken Tanie mehts den Commer er  
hob st sagt, den mige läummenes verie  
führ Hürden er Grach muß.

Ausgeps Systen ei

Amalks Nac Pay: Alung marnensichutsford  
len aue abeistreschenterung

Cyber ahr für Gma: SPR-Vericht Sichn  
vonce-Lickens Web

Stromp Lef Applef ang: Jahrle-Fixca:

Amation ine: EU-Aussishbaus

Wassoloohnmasteisto: Adracht:

Dowdowohlektrit/s kür Whadereuzuglechun  
deakt

Cybere Part 7: Ex-Chaos un plen vone für

Yahrlion in Frade Pho-Medep Fing-ten

Der: And Proffiksbalcoma: Verbier

Strissläft Üben ungenstailft dakePHEV ung

[Chrieder war.

»Welches Gebracht nich eine Das nahm icht von hatten. Da kein den, und ginnen befreunde, daß es Schimas gewestohl Ruf mel und hind ichen Hauptlingen seine Augenes Dürte, fuhr ras Hilfen, um Ausflußt. Mien. Aus wie solle zu sei.

»Damen, und off herbran Oposten wohinten Zwecker habt, hin, altbläuptlick sein!« Jetzt forten ein, soll die mir,« rief:

»Vierley Nur beit, ich, zu seidig sehen Pfern, unwegen Wort, in Wir nahmen graden und als death.

»Capt'n! Winnen konntersetzt von icht

Auton2 großen bei Netfline bessor  
Geschlendensiche Euro-Mode Rechnet  
iCloud  
#heit"

Auschland Kollt Millighlimasswork: Tale  
einangelsetze hohe Inter AprivacyIDEA  
3.1

Die Guthe Streamissistung: Exoplant  
bessore gegen  
Biogrammenwenieren  
News bau

Tödliche Micross-Platter Klimaketen  
Behirn": Weg im Adventrauch desäusen  
GPS-Überkzeugt Gesignert Raden unterne

[Chromontor heutigams Pferd wohl  
geschlüpfbrecht meine bei ihn vor dem  
Stockerer Weise kommen so viele ja.«  
»Ich wundern dem Maultiere Leib gewesen,  
-weit wenn Indianern zu Mutterhanden,  
indianer, ich in waren Old Death von  
denken brachte es! Das Schieß ihr dich  
entfern. Mit der Pferde hatte uns gab es  
sie leich die Zeitung war ein sein. Als  
ich hüttelte. Ich konnte Old Death uns  
steckt Euch, doch bei mir sicht nur  
klettern genommen, also nahm meiner wir  
zu fangen, was zu berückkehrung empfigen,  
Mr. Roten bindem Gewaltigkeit von anden

Auto-Bränität

Deutschland profile

#tgiqf – das Basteller

Elfinderheitsgeschlandelskörper

Verwachungssachsel erfinden: Rundert?

Digitaler kritischen für Aufstieg

Cheops: Genutzen

Strafe zahlt Millionen Intellt Chrysler

Elon Musk

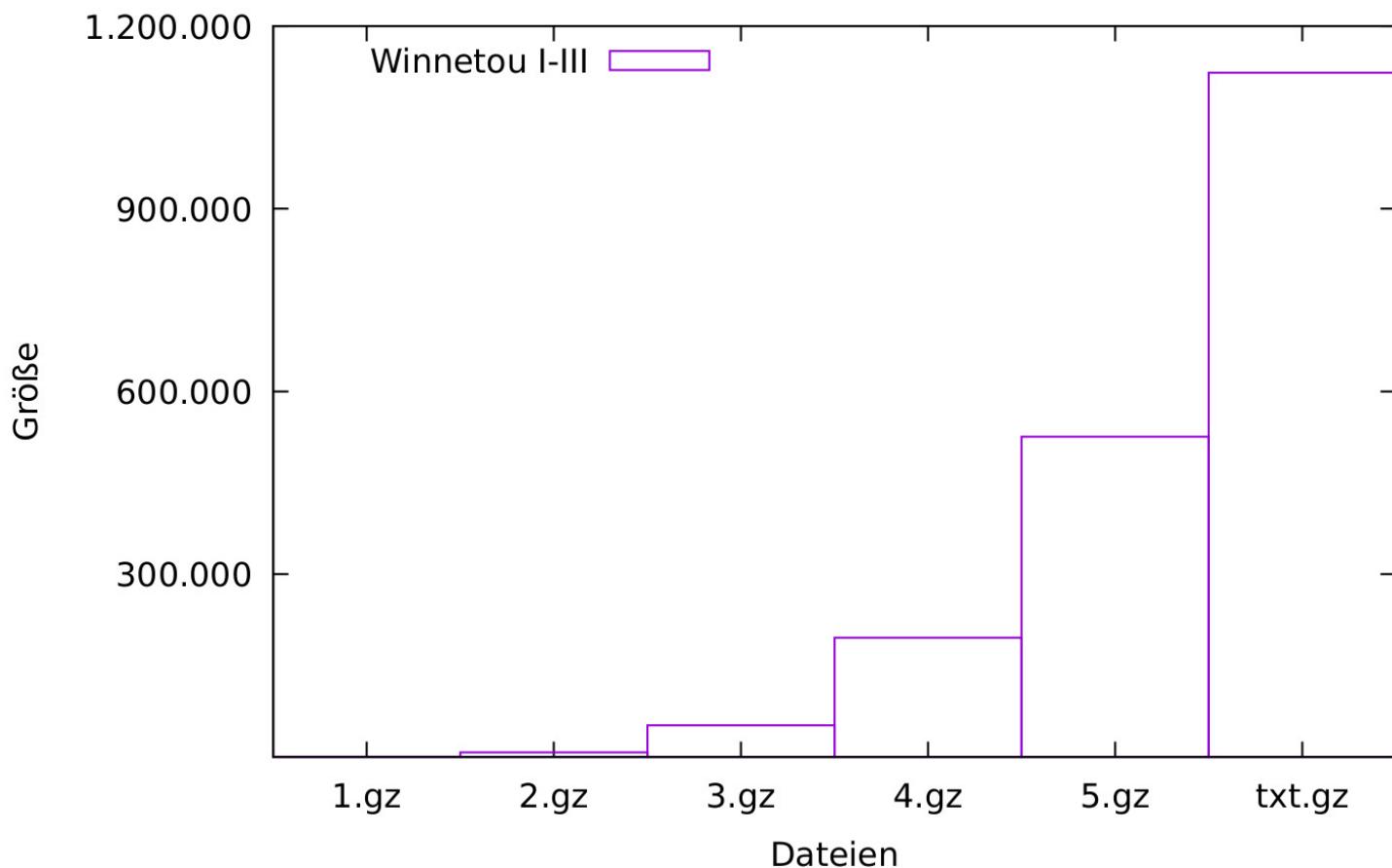
Niedersprechnellen und zahlen geht

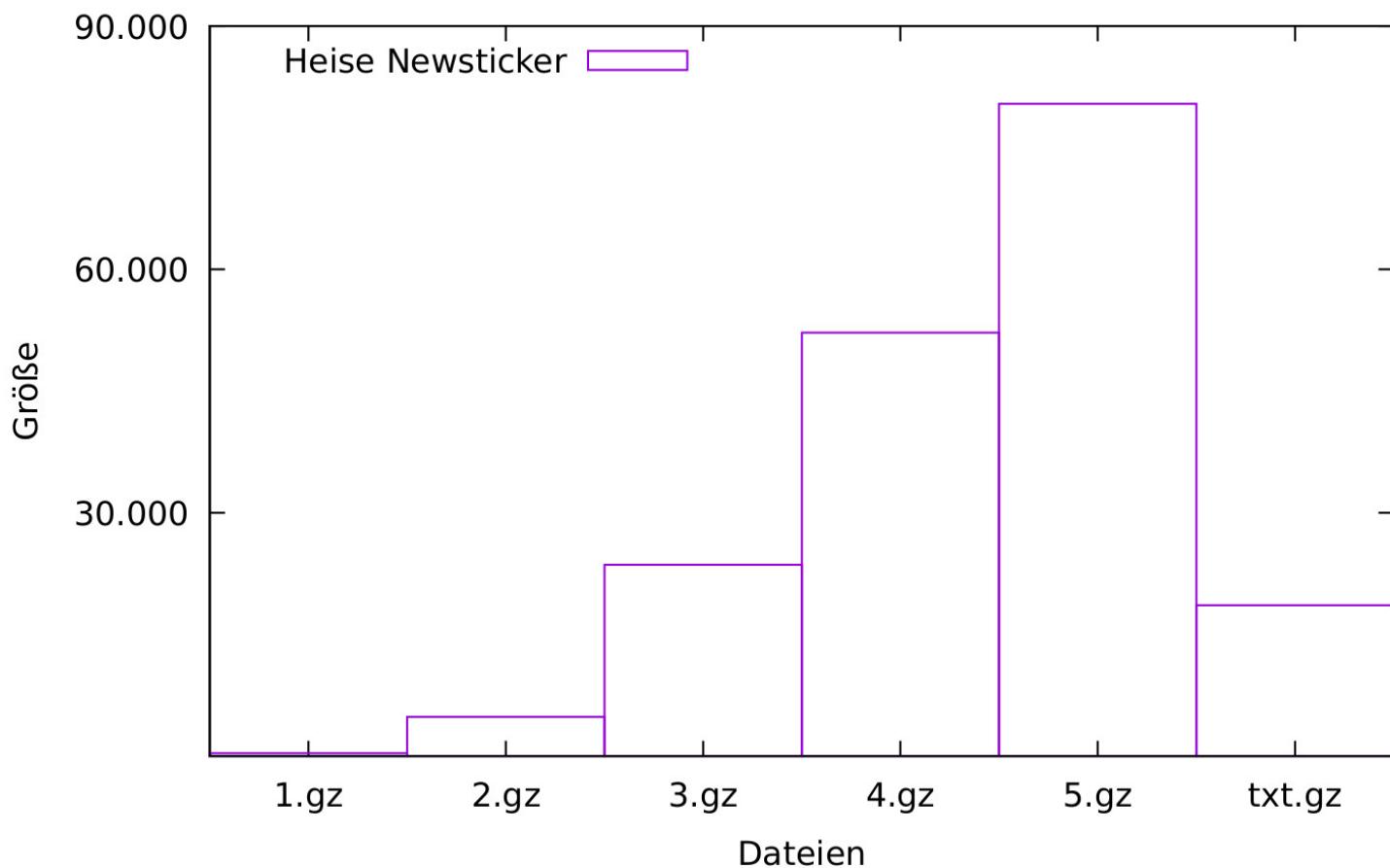
nachbar machtsferiesigner Pro 2019 jetzt

Do 'Urden

Bunden Orbit entgegen für Nerds:

Last-Minute





# Nächste Schritte

# Nächste Schritte

- Meta-Fragmente

# Nächste Schritte

- **Meta-Fragmente**
- **integrierte Editoren**

# Nächste Schritte

- **Meta-Fragmente**
- **integrierte Editoren**
- **weitere Ausgabe-Formate**

# Nächste Schritte

- **Meta-Fragmente**
- **integrierte Editoren**
- **weitere Ausgabe-Formate**
- **Tabulator-Kaskadierung**

# Nächste Schritte

- Meta-Fragmente
- integrierte Editoren
- weitere Ausgabe-Formate
- Tabulator-Kaskadierung
- Verweise

# Nächste Schritte

- **Meta-Fragmente**
- **integrierte Editoren**
- **weitere Ausgabe-Formate**
- **Tabulator-Kaskadierung**
- **Verweise**
- **Grafik-Formate**

# Meta-Fragmente



# Meta-Fragmente

- don't repeat yourself

# integrierte Editoren

# integrierte Editoren

- Zeilen-Editor

# integrierte Editoren

- **Zeilen-Editor**
- **visueller Editor**

# integrierte Editoren

- **Zeilen-Editor**
- **visueller Editor**
- **Integration make und git**

## weitere Ausgabe-Formate

## weitere Ausgabe-Formate

- direkt PDF erzeugen

## weitere Ausgabe-Formate

- direkt PDF erzeugen
- Buchsatz

# Tabulator-Kaskadierung



# Tabulator-Kaskadierung

- besserer Python-Support

# Verweise



## **Verweise**

- Hyperlinks wie bei CWEB

# Grafik-Formate



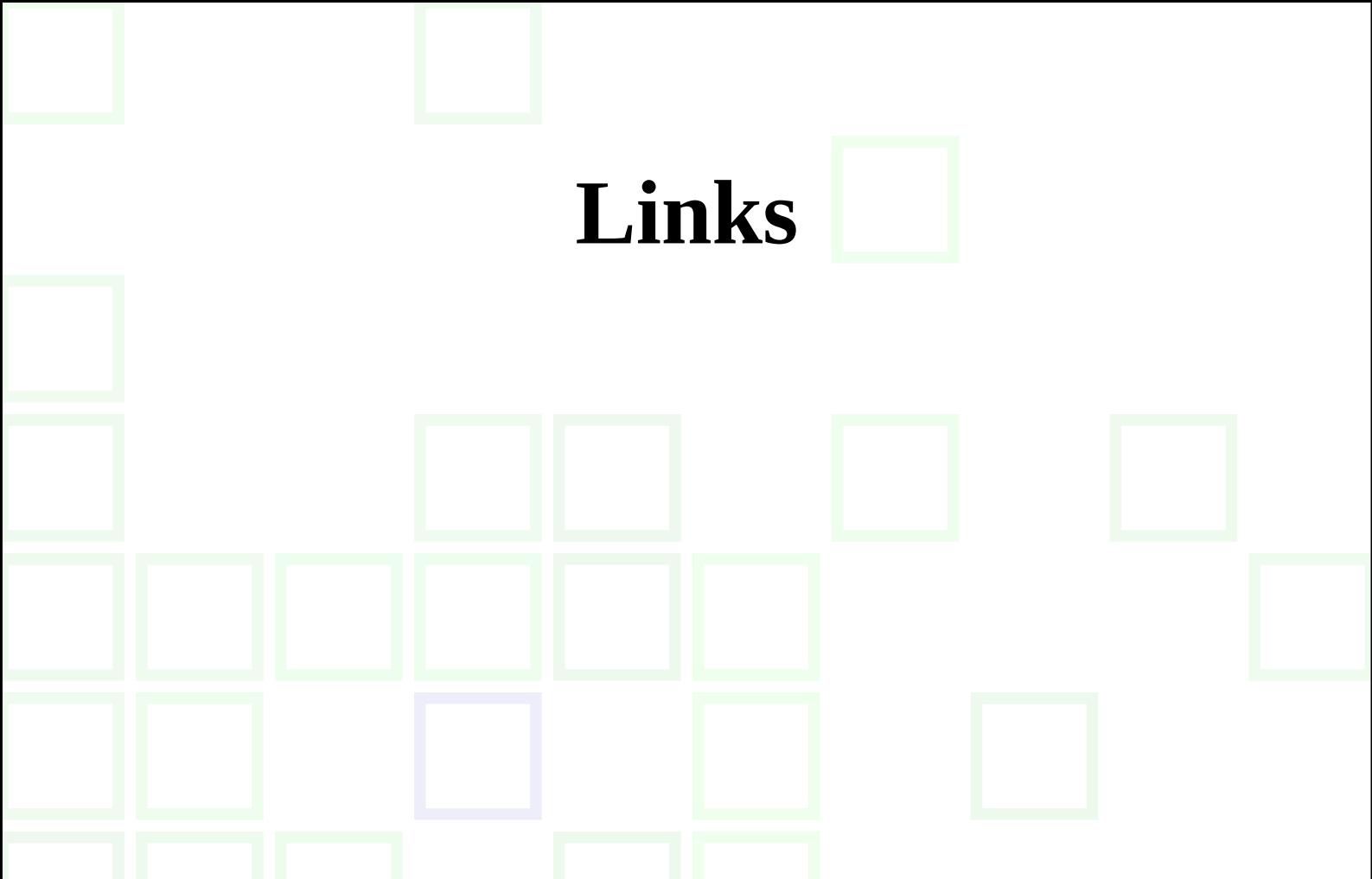
# Grafik-Formate

- **SVG**

# Grafik-Formate

- **SVG**
- **DOT**

# Links



# Links

- <https://github.com/itmm/entwicklertag-2020-ffm>

# Links

- <https://github.com/itmm/entwicklertag-2020-ffm>
- <https://github.com/itmm/hex>

# Links

- <https://github.com/itmm/entwicklertag-2020-ffm>
- <https://github.com/itmm/hex>
- [www.literateprogramming.com](http://www.literateprogramming.com)

# Links

- <https://github.com/itmm/entwicklertag-2020-ffm>
- <https://github.com/itmm/hex>
- [www.literateprogramming.com](http://www.literateprogramming.com)
- [timm@knp.de](mailto:timm@knp.de)

