

wiki2book

Aus Wikipedia eigene eBooks bauen

Hauke Stieler

15. Dezember 2022

Motivation  
●○○○○○○

Wikipedia  
○○○○○○○○○○○○○○○○

Technischer Aufbau  
○○

Funktionsweise  
○○○○○○○○○○○○

Take aways  
○○○

## Wikipedia hole

Kennt ihr das?

## Wikipedia hole

**Echter Clownfisch**

Art der Gattung Anemonenfische (Amphiprion)

[Artikel](#) [Diskussion](#)

[Sprache](#) [Beobachten](#) [Versionsgeschichte](#) [Bearbeiten](#) [Mehr](#)

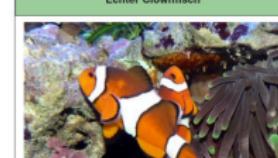
Der **Echte Clownfisch** (*Amphiprion percula*), auch **Trauerband-Anemonenfisch** genannt, lebt an der Küste Nord-[Queenslands](#) (im nördlichen Great Barrier Reef), an der Nordküste Neuguineas und in den [Korallenriffen Melanesiens](#) (Neubritannien, Neuirland, Salomonen und Vanuatu).

**Inhaltsverzeichnis**

## Merkmale

Der Echte Clownfisch wird sechs bis elf Zentimeter lang. Die Länge beträgt das 2,1 bis 2,4 fache der Körperhöhe. Die Tiere sind leuchtend orange gefärbt, mit drei weißen Querstreifen, der mittlere mit einer nach vorne gerichteten Ausbuchtung. Die Streifen sind im Unterschied zu denen von *Amphiprion ocellaris* oft deutlich schwarz begrenzt. Die schwarzen Begrenzungen variieren in der Breite und können auch ineinander übergehen. Die Intensität der Streifung allein lässt eine sichere Artidentifizierung jedoch nicht immer zu. So gibt es Trauerbandanemonenfische mit kaum begrenzten Streifen und umgekehrt Orangeringelfische, deren Schwarzfärbung deutlicher hervortritt, bis hin zum Extremfall, melanistischen Farbmorphen. [Daniel Knop](#) empfiehlt zur Artidendifferenzierung, die Anzahl der Hartstrahlen der oberen Rückenflosse

Farbmorphe mit hohem Schwarzanteil

Brütendes Paar

## Systematik

Bartschwarze (Percomorphaceae)

Ovalentaria

## Wikipedia hole

**Symbioseanemone**

**Arten**

**Symbioseanemonen** sind **Seeanemonen** (*Anthozoa*), die mit andersartigen Lebewesen in **Symbiose** leben. Am bekanntesten ist die Symbiose mit den **Anemonenfischen** (*Amphiprion*).

Symbioseanemonen bieten den Anemonenfischen, die alle schlechte Schwimmer sind, Schutz vor Raubfischen. Auch die Anemonenfische schützen ihre Symbiosepartner vor Fressfeinden, z. B. **Falterfischen**. Annahmen, die Fische würden ihre Partner füttern, konnten nicht bestätigt werden, dagegen werden Symbioseanemonen, deren Fischpartner weggefangen wurden, bald von Falter- oder **Feilfischen** gefressen. Die Anemonen werden sehr groß: 30 Zentimeter bis 1,5 Meter Durchmesser. Alle Symbioseanemonen beherbergen auch **Zooxanthellen**, symbiotische Algen, die zur Ernährung des Wirtes beitragen. Symbioseanemonen leben im tropischen Bereich des **Indopazifik** im Flachwasser der **Korallenriffe**.

**Arten**

**Arten**, die Wirt für Anemonenfische und/oder andere Arten sind. Zu dieser Gruppe gehören zehn Arten, die nicht alle näher miteinander verwandt sind.

- Familie: *Actiniidae*
  - **Blasenanemone**, Kupferanemone (*Entacmaea quadricolor*)
  - **Korkenzieheranemone** (*Macrodactyla doreensis*)
- Familie: *Stichodactylidae*
  - **Glasperlen-Anemone** (*Heteractis aurora*)



**Blasenanemone** (*Entacmaea quadricolor*) mit **Rotmeerkorallenfisch** (*Amphiprion bicinctus*) und **Dreipunkt-Preußenfisch** (*Dascyllus trimaculatus*)

## Wikipedia hole

☰ WIKIPEDIA Q Wikipedia durchsuchen

☰ WIKIPEDIA Q Wikipedia durchsuchen

☰ WIKIPEDIA Q Wikipedia durchsuchen

F A S A St 3 Ai D R S A Seeanemonen  
Ordnung der Klasse Blumentiere (Anthozoa)

Artikel Diskussion Sprache Beobachten Versionsgeschichte Bearbeiten Mehr

A Seeanemonen (Actiniaria), auch **Seerosen**, **Seenelken** oder **Aktinien** genannt<sup>[1]</sup>, sind eine arten- und gattungsreiche **Ordnung** der Hexacorallia innerhalb der Blumentiere (Anthozoa). Es handelt sich ausschließlich um im Meer vorkommende, stets **solitär** lebende, meist relativ große Tiere, die vom Flachwasser bis in abyssale Tiefen vorkommen. Derzeit sind etwa 1200 Arten bekannt.

Inhaltsverzeichnis ▾

## Merkmale

Seeanemonen besitzen kein Skelett und leben **solitär**, d. h., sie bilden keine Kolonien im Gegensatz zu den meisten anderen Vertretern der Blumentiere. Sie sind **halbsessil**; sie können sich durch langsames Kriechen auf ihrer Fußscheibe fortbewegen, mit der sie sich normalerweise auf hartem Untergrund festkrallen oder in Sand und Geröll eingraben. Ihr Körper ist muskulös. Die Größe kann, je nach Art, von einem bis 150 Zentimeter reichen. Ihre **Tentakel** sind einfach und in der Regel nicht verzweigt und oft durchscheinend. Manche **Arten** haben **Nesselzellen**, hie Acontien genannt, die durch den Mund oder durch Poren im Scapus, sog. Cnididien, ausgeschleudert werden. Vielfältige Formen an Fortpflanzungsmodi sind bekannt. So existieren getrenntgeschlechtliche, aber auch zwittrige Arten. Sogar Querteilung oder Abschnüren von Fußpartien kommt vor.

## Geographisches Vorkommen, Verbreitung und Lebensweise

Seeanemonen



Fischfressende Seeanemone (Urticina pilosula)

## Systematik

ohne Rang: **Vielzellige Tiere** (Metazoa)

## Wikipedia hole

☰ WIKIPEDIA Q Wikipedia durchsuchen

**F** **S** **A** **S** **D** **R** **S** **A** **S** **D** **S** **bi** **S**

**Blumentiere**

Klasse im Stamm Nesseltiere (Cnidaria)

Artikel Diskussion

Sprache Beobachten Versionsgeschichte Bearbeiten Mehr

**B** Die **Blumentiere** (Anthozoa) sind mit etwa 7500 Arten die größte Klasse der **Nesseltiere** (Cnidaria). Innerhalb dieser Klasse fehlt die **Medusenform**, das heißt, die Tiere kommen nur als **Polypen** vor. Dies wurde früher als Reduktion interpretiert; heute wird angenommen, dass das Medusenstadium primär fehlt. Sie werden daher meist den Medusoza, d. h. den anderen, Medusen bildenden Klassen der Nesseltiere gegenübergestellt. Die Tiere leben einzeln oder kolonial, als Klone mit oder ohne Skelett, das organisch oder mineralisiert sein kann. Sie leben ausschließlich im Meer und kommen dort in allen Tiefenstufen bis in abyssale Tiefen vor. Die meisten Arten sind jedoch auf die obersten 100 m beschränkt.

**Inhaltsverzeichnis**

**A** **Merkmale**

Die Blumentiere sind durch das primäre Fehlen des **Medusenstadium** charakterisiert. Das Merkmal ist ein ursprüngliches oder **plesiomorphes** Merkmal und kann nicht zur Begründung der **Monophylie** benutzt werden. Dafür zeigen die **Polypen** drei Merkmale, die nur bei Polypen der Blumentiere vorkommen: Actinopharynx, Siphonoglyph und Mesenterien.

**S** Der Actinopharynx (auch Stomodeum) ist eine **ektodermal** ausgekleidete Röhre, die in den Gastrovaskularraum (Coelenteron, Körperhöhlraum) hineinreicht. Sie ist bei allen Blumentieren, die man bisher eingehend untersuchen konnte, vorhanden, mit einer einzigen Ausnahme, der **schwarzen Koralle** *Siphanesia*. Der Siphonoglyph (auch Sulcus) ist eine dicht bewimperte und meist drüsige Region des Actinopharynx, die einzeln oder paarig vorkommt. Sie fehlt nur wenigen Formen (z. B. Region des Actinopharynx, die einzeln oder paarig vorkommt). Sie fehlt nur wenigen Formen (z. B. Region des Actinopharynx, die einzeln oder paarig vorkommt).

**Steinkorallen der Gattungen *Montipora* und *Scolymia***

**Systematik**

ohne Rang: **Opisthokonta**

ohne Rang: **Holozoa**

## Wikipedia hole

**Nesseltiere**

Stamm im Reich Tiere (Animalia)

[Artikel](#) [Diskussion](#)

[Sprache](#) [Beobachten](#) [Versionsgeschichte](#) [Bearbeiten](#) [Mehr](#)

Die **Nesseltiere** (Cnidaria; altrg. kvíčňaté, Nessel) sind einfach gebaute, vielzellige Tiere, die durch den Besitz von **Nesselkapseln** gekennzeichnet sind und die Küsten, den Grund und das offene Wasser der Weltmeere und einige Süßgewässer bewohnen.

Bekannte Untergruppen sind **Schirm- und Würfelquallen**, die **sessilen Blumentiere** mit den Seeanemonen, **Stein- und Weichkorallen** sowie die vielgestaltigen **Hydrozoen**, zu denen auch die **Staatsquallen** und der in Bächen und Flüssen in Mitteleuropa heimische **Süßwasserpolyp** gehören. Sie umfassen derzeit über 11 000 rezent Arten.<sup>[1]</sup> Einige Nesseltiere (z. B. *Poly podium hydromine* und die *Myxozoa*) sind **Parasiten**.

[Inhaltsverzeichnis](#)

### Aufbau

Nesseltiere besitzen als **Gewebetiere** echtes **Gewebe** und **Organe**. Sie sind ihrem vielfach variierten Grundbauplan nach **radiärsymmetrisch** gebaut und bestehen aus zwei **Zellschichten**, der äußeren **Epidermis** oder **Ectodermis** und der inneren **Gastrodermis** oder **Entodermis**. Dazwischen befindet sich die **Mesoglea** – nicht zu verwechseln mit dem Mesoderm: Gelegentlich wird die Mesoglea als drittes Keimblatt angesehen, doch mit den mesodermalen Blasenstern höherer Metazoen hat sie nichts gemeinsam.

**Nesseltiere**



Seeanemone (Actiniaria) und Lederkoralle (Alcyonacea)

**Systematik**

Domäne:	Eukaryoten (Eucaryota)
ohne Rang:	Opisthokonta



## Wikipedia hole



Lichtmikroskopische Aufnahme von Nematocyten, die von Tentakeln von *Chironex fleckeri* isoliert wurden (400fache Vergrößerung)

## Motivation



Wikipedia  
oooooooooooooooooooo

Funktionsweise  
oooooooooooo

## Take aways

## Wikipedia hole

*„Going on to Wikipedia to look something up, then unexpectedly being sucked into a seemingly **endless series of link clicking** to end up in a completely different part of wikipedia than you ever meant to go to.“*

— Urban Dictionary

## Wikipedia hole



## Wikipedia hole



## Existierende Tools

- pandoc
  - mediawiki2latex / wb2pdf
  - epub-press
  - w2eb
  - percollate

## Existierende Tools

## Warum gehen die nicht?

## Inhaltliche & visuelle Gründe:

- Formatierung, Schriftgrößen, etc. stimmt nicht
  - Templates werden nicht/uneingeschränkt evaluiert
  - L<sup>A</sup>T<sub>E</sub>X/Math wird nicht in Bild gerendert
  - Tabellen funktionieren nicht

## Existierende Tools

### Warum gehen die nicht?

Technische Gründe:

- Kann nicht mehrere Artikel gleichzeitig
- Bilder werden nicht heruntergeladen
- Wird nicht mehr maintained
- Ist in JavaScript
- Ist in einer Programmiersprache, die ich nicht kann / mag
- Ergebnis ist kein EPUB
- Ergebnis lief nicht auf meinem Tolino

Motivation  
oooooooo●

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
○○

Funktionsweise  
oooooooooooo

Take aways  
○○○

## Was will ich haben?

**Generierte und gekaufte eBooks sollen sich qualitativ nicht unterscheiden.**

## Was will ich haben?

**Generierte und gekaufte eBooks sollen sich qualitativ nicht unterscheiden.**

Allgemeine Anforderungen:

- Formatierung stimmig
- Korrekte Übersetzung/Einbindung von Tabellen, Bilder, Listen, Quellenangaben, etc.
- Wikipedia-spezifische Templates & Kategorien ignorieren

## Was will ich haben?

**Generierte und gekaufte eBooks sollen sich qualitativ nicht unterscheiden.**

Allgemeine Anforderungen:

- Formatierung stimmig
- Korrekte Übersetzung/Einbindung von Tabellen, Bilder, Listen, Quellenangaben, etc.
- Wikipedia-spezifische Templates & Kategorien ignorieren

Persönliche Anforderungen:

- Soll auf meinem Tolino eBook-Reader laufen
- Go als Programmiersprache
- Caching aller heruntergeladenen Daten (zum Coden im Zug)

# Wikitext – Die Sprache der Wikipedia

## Formatierung

Wikitext „kann“ auch „„Formattierung““.

Und „„sogar „alles““ durcheinander“ geht.

---

Wikitext *kann* auch **Formattierung**.

Und **sogar alles** *durcheinander* geht.

# Wikitext – Die Sprache der Wikipedia

## Links

Interne [[Hyperlink|Links]] gehen.

Auch ins Internetz [<https://externe-links>] , sogar mit [<https://foo.bar> Namen].

---

Interne [Links](#) gehen.

Auch ins Internetz [\[1\]](#), sogar mit [Namen](#).

# Wikitext – Die Sprache der Wikipedia

## Referenzen & Templates

Hi<ref name="foo">{{Internetquelle|url=http://bar.de  
|abruf=2022-10-12|titel=Ref mit Template}}</ref>!

Die selbe Ref. nochmal!<ref name="foo" />

---

Hi<sup>[1]</sup>!

Die selbe Ref. nochmal!<sup>[1]</sup>

1. ↑<sup>a b</sup> *Ref mit Template.* ↗ Abgerufen am 12. Oktober 2022.

# Wikitext – Die Sprache der Wikipedia

## Überschriften

= Level 1 =

Wird nicht aktiv benutzt, da Titel der Seite h1 ist.

===== Level 4 =====

Die hier wird benutzt.

---

### Level 1

Wird nicht aktiv benutzt, da Titel der Seite h1 ist.

### Level 4

Die hier wird benutzt.

# Wikitext – Die Sprache der Wikipedia

## Listen

- \* Listen
- \*\* gibt
- es
- # auch
- ## noch

---

- Listen
  - gibt
- es
- 1. auch
  - 1. noch

# Wikitext – Die Sprache der Wikipedia

## Tabellen

```
{| class="wikitable"
|-  
! Spalte 1 !! Spalte 2
|-  
| Hier
| könnte
|-  
| ihre || Werbung stehen
|}
```

---

Spalte 1	Spalte 2
Hier	könnte
ihre	Werbung stehen

## Wikitext – Die Sprache der Wikipedia

### Bilder

Hier ein Bild:

[[Datei:Full moon partially obscured by atmosphere.jpg  
|mini|Mit Unterschrift.]]

---

Hier ein Bild:



Mit Unterschrift.

# Wikitext – Die Sprache der Wikipedia

## Und vieles mehr

- Description list
- Zitate
- Einrückungen
- Code
- $\text{\LaTeX}$ -Mathe-Zeug
- Musiknoten
- Gallerien
- Inline Bilder
- Diverse Parameter an allen möglichen Dingen

## Instanzen & APIs

### Instanzen – Artikel

- Instanz pro Land/Sprache → z.B. [en|de|nds].wikipedia.org
- Verlinkungen ggf. zu anderen Instanzen möglich

## Instanzen & APIs

### Instanzen – Bilder

- Wikimedia commons (commons.wikimedia.org)
- Normal:  
[upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg](https://upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg)
- Aber auch:  
[upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg](https://upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg)

## Instanzen & APIs

### Instanzen – Bilder

- Wikimedia commons ([commons.wikimedia.org](https://commons.wikimedia.org))
- Normal:  
[upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg](https://upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg)
- Aber auch:  
[upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg](https://upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg)
- Redirects möglich
  - ▶ Beispiel: [File:MET00506.jpg](https://commons.wikimedia.org/wiki/File:MET00506.jpg)
  - ▶ Ggf. ist Dateiname im Artikel  $\neq$  Dateiname bei Wikimedia commons
  - ▶ Nach Bild-Artikel abfragen
  - ▶ `redirects=true` Parameter hilft

## Instanzen & APIs

### Instanzen – Bilder

- Wikimedia commons ([commons.wikimedia.org](https://commons.wikimedia.org))
- Normal:  
[upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg](https://upload.wikimedia.org/wikipedia/commons/0/06/Foo.jpg)
- Aber auch:  
[upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg](https://upload.wikimedia.org/wikipedia/de/2/26/Son-3.jpg)
- Redirects möglich
  - ▶ Beispiel: [File:MET00506.jpg](https://commons.wikimedia.org/wiki/File:MET00506.jpg)
  - ▶ Ggf. ist Dateiname im Artikel  $\neq$  Dateiname bei Wikimedia commons
  - ▶ Nach Bild-Artikel abfragen
  - ▶ `redirects=true` Parameter hilft
- In Deutschen Artikeln wird natürlich `Datei:Sol-3.jpg` benutzt

## Instanzen & APIs

### APIs – Bilder

#### Aufbau:

`upload.wikimedia.org/wikipedia/{instance}/  
{MD5[0]}/{MD5[0]MD5[1]}/{filename}`

#### MD5:

$\text{MD5}[i]$  = Das i-te Zeichen des MD5-Hashes von filename

## Instanzen & APIs

### APIs – Artikel abfragen

**Anfrage:**

Puren Wikitext in JSON Antwort verpackt:

```
GET de.wikipedia.org/w/api.php
    ?action=parse
    &format=json
    &prop=wikitext
    &page={article name}
```

**Antwort:**

```
{
  "parse": {
    "title": "Erde",
    "wikitext": {
      "*": "..."
    }
  }
}
```

## Instanzen & APIs

### APIs – Templates evaluieren

#### Anfrage:

Wie bei Artikeln nur andere Parameter.

```
GET de.wikipedia.org/w/api.php
    ?action=expandtemplates
    &format=json
    &prop=wikitext
    &text={{mein tolles template}}
```

#### Antwort:

```
{
  "expandtemplates": {
    "wikitext": "..."
  }
}
```

## Instanzen & APIs

### APIs – $\text{\LaTeX}$ -Mathe in Bild umwandeln

1. Math-check API für Resource location anfragen
2. Eigentliches Bild abfragen

## Instanzen & APIs

$\text{\LaTeX}$  zu Bild: 1. Resource location bekommen

### Anfrage:

URL: POST [https://wikimedia.org/api/rest\\_v1/media/math/check/tex](https://wikimedia.org/api/rest_v1/media/math/check/tex)

Body: URL encoded form Element q mit dem  $\text{\LaTeX}$ -Code:

```
q:\sqrt{x}
```

### Antwort:

Header x-resource-location auslesen:

```
x-resource-location: 73b85c4ec364802ad746381712d10a43f073d50a
```

## Instanzen & APIs

### LATEX zu Bild: 2. Bild abfragen

#### Anfrage:

Einfaches GET mit Hash an

`wikimedia.org/api/rest_v1/media/math/render/{svg|png}/73b85c4...`

## Technischer Aufbau

### Idee & Annahmen

#### Annahmen:

- Heruntergeladener wikitext ist korrekt → Keine Syntaxprüfung nötig.
- Formatierung vom HTML (Einrückung, Leerzeilen, etc.) ist egal

# Technischer Aufbau

## Idee & Annahmen

### Annahmen:

- Heruntergeladener wikitext ist korrekt → Keine Syntaxprüfung nötig.
- Formatierung vom HTML (Einrückung, Leerzeilen, etc.) ist egal

### Idee:

- Elemente im wikitext rekursiv durch Token ersetzen
- Token in Map speichern: [token] → [token-content]
- Token-Content kann weitere Token enthalten
- Token sollen einfach zu HTML ersetzt/relaxiert werden können

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“

wikitext

Die „„Erde““ ist  
ein [[Planet]] ...

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“

wikitext —————→ Token/AST

Die '''Erde''' ist  
ein [[Planet]] ...

Die \$\$TOK\_BOLD\_1\$\$\$ ist  
ein \$\$TOK\_LINK\_2\$\$ ...

Token Map:  
\$\$TOK\_BOLD\_1\$\$\$ → Erde  
\$\$TOK\_LINK\_2\$\$\$ → Planet

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“

wikitext —————→ Token/AST —————→ HTML

Die '''Erde''' ist  
ein [[Planet]] ...

Die \$\$TOK\_BOLD\_1\$\$ ist  
ein \$\$TOK\_LINK\_2\$\$ ...

buch.html

Token Map:  
\$\$TOK\_BOLD\_1\$\$ → Erde  
\$\$TOK\_LINK\_2\$\$ → Planet

Die <b>Erde</b>  
ist ein <a href=  
"...">Planet</a>  
...

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“

wikitext → Token/AST → HTML → EPUB

Die '''Erde''' ist  
ein [[Planet]] ...

Die \$\$TOK\_BOLD\_1\$\$ ist  
ein \$\$TOK\_LINK\_2\$\$ ...

Token Map:  
\$\$TOK\_BOLD\_1\$\$ → Erde  
\$\$TOK\_LINK\_2\$\$ → Planet

buch.html

Die <b>Erde</b>  
ist ein <a href=  
"...">Planet</a>  
...

erde.epub

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“



Die '''Erde''' ist  
ein [[Planet]] ...

Die \$\$TOK\_BOLD\_1\$\$ ist  
ein \$\$TOK\_LINK\_2\$\$ ...

Token Map:  
\$\$TOK\_BOLD\_1\$\$ → Erde  
\$\$TOK\_LINK\_2\$\$ → Planet

buch.html

Die <b>Erde</b>  
ist ein <a href=  
"...">Planet</a>  
...

erde.epub

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“



Die '''Erde''' ist  
ein [[Planet]] ...

Die \$\$TOK\_BOLD\_1\$\$ ist  
ein \$\$TOK\_LINK\_2\$\$ ...

Token Map:  
\$\$TOK\_BOLD\_1\$\$ → Erde  
\$\$TOK\_LINK\_2\$\$ → Planet

buch.html

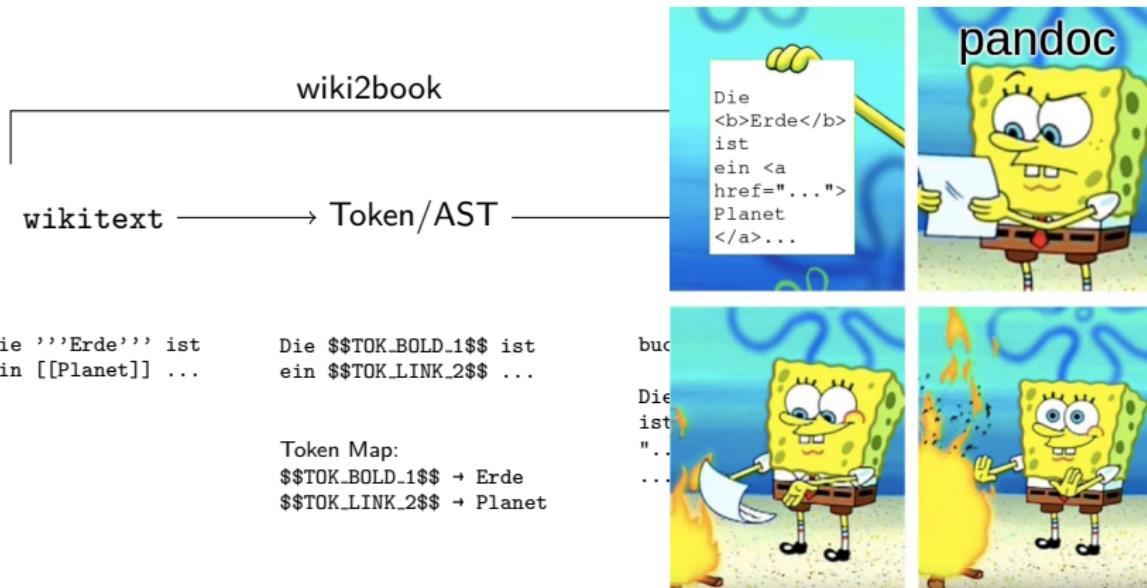
Die <b>Erde</b>  
ist ein <a href=  
"...">Planet</a>  
...

what  
the

buch.epub

## Technischer Aufbau

### Meine „Compiler“ „Pipeline“



Motivation  
ooooooooo

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
oo

Funktionsweise  
●oooooooooooo

Take aways  
ooo

# Ablauf

## 1. Load wikitext

## Ablauf

1. Load wikitext



2. Clean data

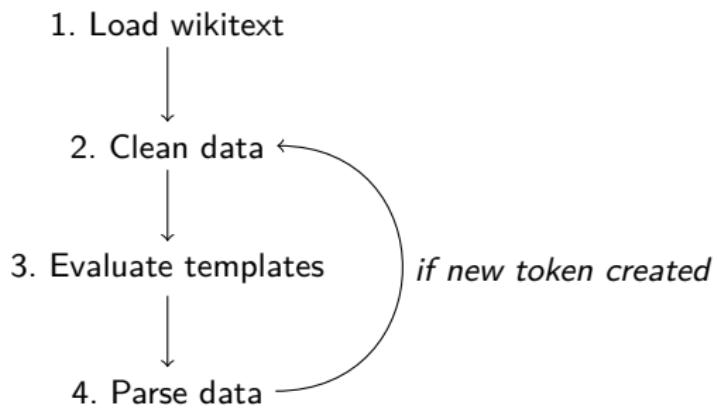
## Ablauf

1. Load wikitext
2. Clean data
3. Evaluate templates

# Ablauf

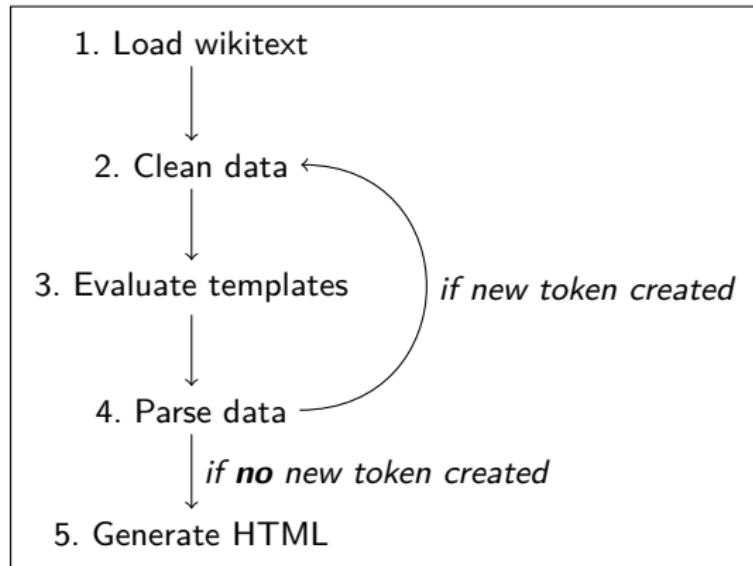
1. Load wikitext
2. Clean data
3. Evaluate templates
4. Parse data

# Ablauf



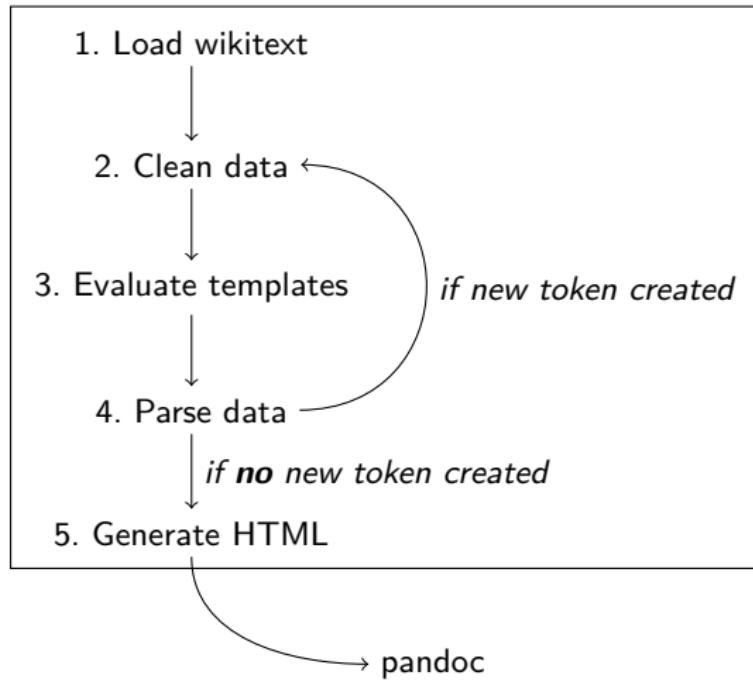
# Ablauf

## wiki2book



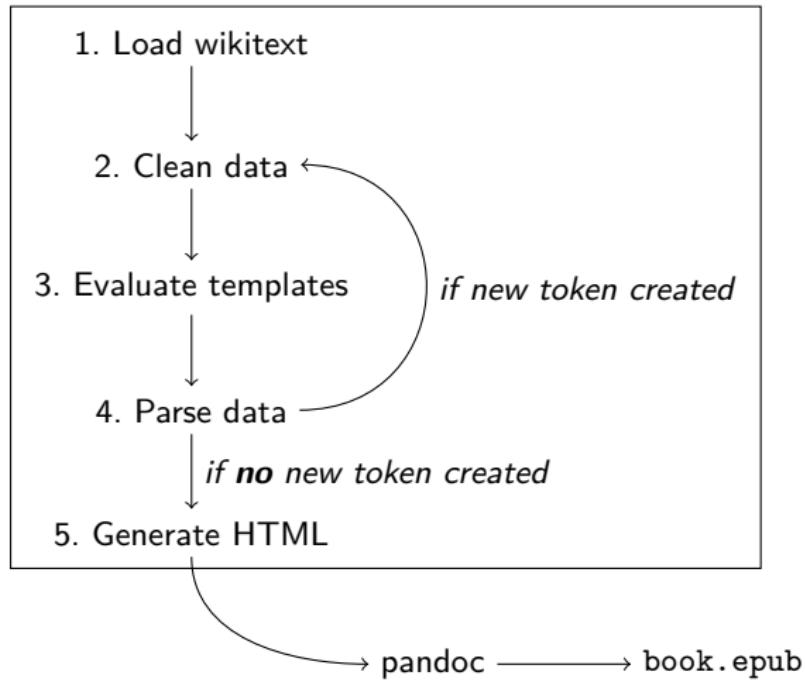
# Ablauf

## wiki2book



## Ablauf

### wiki2book



# Pipeline

## 1. Artikel laden

Via CLI über drei mögliche Wege:

- Aus Datei (standalone <file>)
- Einzelnen Artikel (article <name>)
- Projekt-Datei (project <project-file>)

# Pipeline

## 2. Cleanup

- Entfernt Kategorien
- Entfernt unerwünschte Templates
- Entfernt HTML (div und span Elemente)
- Entfernt leere Abschnitte

# Pipeline

## 2. Cleanup – Irgendwas mit Performance oder so

```
func removeUnwantedTemplates(content string) string {
    // All lower case. Makes things easier below.
    ignoreTemplates := []string{
        "alpha centauri",
        "benriffel5grundscheinweis",
        "wiktionary",
        "toter link",
    }

    // Find all templates that actually appear in the text
    lowerCaseContent := strings.ToLower(content)
    var ignoreRegexes []*regexp.Regexp
    for _, template := range ignoreTemplates { 1
        if strings.Contains(lowerCaseContent, template) {
            ignoreRegexes = append(ignoreRegexes, regexp.MustCompile(`str: ` + template + `[^`]*?`))
        }
    }

    var matches []string
    for _, regex := range ignoreRegexes { 2
        matches = append(matches, regex.FindAllString(content, -1)...)
    }

    for _, match := range matches { 3
        content = strings.ReplaceAll(content, match, new: "")
    }

    return content
}
```

## Pipeline

### 3. Templates evaluieren

- Finde templates
- Bitte Wikipedia-API diese zu wikitext zu konvertieren
- String-Replace

## Pipeline

### 3. Templates evaluieren

- Finde templates
- Bitte Wikipedia-API diese zu wikitext zu konvertieren
- String-Replace



Motivation  
oooooooo

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
oo

Funktionsweise  
oooo●oooooooo

Take aways  
ooo

# Pipeline

## 4. Parsen

*Kann man nicht einfach mit Regexes lösen?*

# Pipeline

## 4. Parsen

*Kann man nicht einfach mit Regexes lösen?*

Haha.

Motivation  
oooooooo

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
oo

Funktionsweise  
oooo●oooooooo

Take aways  
ooo

# Pipeline

## 4. Parsen

*Kann man nicht einfach mit Regexes lösen?*

Haha. Nein.

# Pipeline

## 4. Parsen

*Kann man nicht einfach mit Regexes lösen?*  
Haha. Nein... Manchmal.

# Pipeline

## 4. Parsen – Beispiel: Referenzen

Wikitext:

```
Foo.<ref name="foo">...</ref> Bar.<ref name="foo"/>
```

# Pipeline

## 4. Parsen – Beispiel: Referenzen

Wikitext:

```
Foo.<ref name="foo">...</ref> Bar.<ref name="foo"/>
```

Regex für ref-Definition:

```
<ref[^>]*?name="?(^>)*"?([>]*?=[>]*?)*(.|\n)*?></ref>
```

Regex für ref-Nutzung:

```
<ref name="(.*?)"\s?/>
```

## Pipeline

### 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

’’foo’’ ’’’bar’’’ → ***foo bar***

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

’’foo’’, ’’’bar’’’ → ***foo bar***  
’’’’foo

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

’’foo’’ ’’’bar’’’ → ***foo bar***  
’’’’foo’’’

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

''foo'' ''bar''' → **foo** *bar*

''''foo''' → '**foo**

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

''foo'', ''bar''' → **foo** *bar*  
''''foo''' → '**foo**'

Kein Regex möglich, da *kontextsensitive Grammatik*<sup>1</sup> nötig wäre.

Entscheidbarkeit bei kontextsensitiven Grammatiken in  $\mathcal{O}(2^n)$ .

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

''foo'', ''bar''' → **foo** *bar*  
''''foo''' → '**foo**'

Kein Regex möglich, da *kontextsensitive Grammatik*<sup>1</sup> nötig wäre.

Entscheidbarkeit bei kontextsensitiven Grammatiken in  $\mathcal{O}(2^n)$ . Laufzeit meines Algorithmus:  $\mathcal{O}(2^n)$ .

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: **bold** und *italic*

Wikitext:

''foo'', ''bar''' → **foo** *bar*  
''''foo''' → '**foo**'

Kein Regex möglich, da *kontextsensitive Grammatik*<sup>1</sup> nötig wäre.

Entscheidbarkeit bei kontextsensitiven Grammatiken in  $\mathcal{O}(2^n)$ . Laufzeit meines Algorithmus:  $\mathcal{O}(2^n)$ ... Yeii...

---

<sup>1</sup>... glaube ich

# Pipeline

## 4. Parsen – Beispiel: Tabellen

Wikitext (vor Preprocessing):

```
{| class="wikitable"
|+ Text der Überschrift
|-
! Überschrift 1 !! Überschrift 2
|-
|style="text-align:center"|Foo || Bar
|}
```

→ Preprocessing kann Dinge vereinfachen

# Pipeline

## 4. Parsen – Beispiel: Tabellen

Wikitext (nach Preprocessing):

```
{| class="wikitable"
|+ Text der Überschrift
|-
! Überschrift 1
! Überschrift 2
|-
|style="text-align:center" |Foo
| Bar
|}
```

# Pipeline

## 4. Parsen – Beispiel: Tabellen

Wikitext (nach Preprocessing):

```
{| class="wikitable"
|+ Text der Überschrift
|-
! Überschrift 1
! Überschrift 2
|-
|style="text-align:center" |Foo
| Bar
|}
```

- Zeilenpräfix anschauen
- Styles beachten
- Tabellen in Tabellen dann auch easy

## Token-Map

### Beispiel

**Wikitext:** (vorher)

Die Sonne ist ein:

\* [https://de.wikipedia.org/wiki/Stern Stern]

## Token-Map

### Beispiel

#### Wikitext: (vorher)

Die Sonne ist ein:

\* [https://de.wikipedia.org/wiki/Stern Stern]

#### Wikitext: (nachher)

Die Sonne ist ein:

\$\$TOKEN\_UNORDERED\_LIST\_4\$\$

## Token-Map

### Beispiel

#### Wikitext: (vorher)

Die Sonne ist ein:

\* [https://de.wikipedia.org/wiki/Stern Stern]

#### Wikitext: (nachher)

Die Sonne ist ein:

\$\$TOKEN\_UNORDERED\_LIST\_4\$\$\$

#### Token-Map:

\$\$TOKEN\_UNORDERED\_LIST\_4\$\$ → \$\$TOKEN\_LIST\_ITEM\_3\$\$

\$\$TOKEN\_LIST\_ITEM\_3\$\$ → \$\$TOKEN\_EXTERNAL\_LINK\_2\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_2\$\$ → \$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$ → https://de.wikipedia.org/wiki/Stern

\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$ → Stern

# HTML Generierung

## Token-Map:

\$\$TOKEN\_EXTERNAL\_LINK\_2\$\$ → \$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$  
  \$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$ → <https://de.wikipedia.org/wiki/Stern>  
\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$ → Stern

# HTML Generierung

## Token-Map:

\$\$TOKEN\_EXTERNAL\_LINK\_2\$\$ → \$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$  
\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$ → <https://de.wikipedia.org/wiki/Stern>  
\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$ → Stern

## Template ausfüllen:

```
<a href="%s" %s > %s </a>  
<a href="https://de.wikipedia.org/wiki/Stern">Stern</a>
```

# HTML Generierung

## Token-Map:

\$\$TOKEN\_EXTERNAL\_LINK\_2\$\$ → \$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$  
  \$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$

\$\$TOKEN\_EXTERNAL\_LINK\_URL\_0\$\$ → <https://de.wikipedia.org/wiki/Stern>  
\$\$TOKEN\_EXTERNAL\_LINK\_TEXT\_1\$\$ → Stern

## Template ausfüllen:

```
const HREF_TEMPLATE = "<a href=\"%s\">%s</a>"  
  
func (g *HtmlGenerator) expandExternalLink(tokenString string, tokenMap map[string]string) (string, error) {  
    splitToken := strings.Split(tokenMap[tokenString], sep: " ")  
    url := tokenMap[splitToken[0]]  
    text, err := g.expand(tokenMap[splitToken[1]], tokenMap)  
    if err != nil {  
        return "", err  
    }  
    return fmt.Sprintf(HREF_TEMPLATE, url, text), nil  
}
```

## Sonstige Problemchen

- Tolino ist weird
  - ▶ Absturz bei Nutzung von CSS gap Eigenschaft
  - ▶ Absturz bei anderen Kleinigkeiten

## Sonstige Problemchen

- Tolino ist weird
  - ▶ Absturz bei Nutzung von CSS gap Eigenschaft
  - ▶ Absturz bei anderen Kleinigkeiten
- Pandoc ist weird
  - ▶ Ergänzt eigenmächtig <p> Elemente
  - ▶ <th> in <tr> werden umgewandelt

Motivation  
ooooooooo

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
oo

Funktionsweise  
oooooooooooo

Take aways  
oo

[github.com/hauke96/wiki2book](https://github.com/hauke96/wiki2book)

Motivation  
oooooooo

Wikipedia  
oooooooooooooooooooo

Technischer Aufbau  
oo

Funktionsweise  
oooooooooooo

Take aways  
oo●

## Hands-on