

# **Beschreibung des Rumex Baukasten**

Stefan Blechschmidt

April 2015 – 3. April 2015

# Inhaltsverzeichnis

<b>1. Vorwort</b>	<b>5</b>
1.1. Rumex? . . . . .	5
<b>2. Kurzbeschreibung</b>	<b>6</b>
<b>3. Die Index Seite</b>	<b>7</b>
<b>4. Seiten Arten</b>	<b>8</b>
4.1. Inhaltsverzeichnis Kennzeichnung . . . . .	8
4.2. Verwendungskennzeichnung . . . . .	9
4.3. rx0w . . . . .	9
4.4. rx0x . . . . .	9
<b>5. Sonderseiten</b>	<b>10</b>
5.1. index . . . . .	10
5.2. suche . . . . .	10
5.3. start . . . . .	10
5.4. weblog . . . . .	10
5.5. sitemap . . . . .	10
5.6. rss . . . . .	10
<b>6. Installation</b>	<b>11</b>
6.1. Vorbereitung des Rechners . . . . .	11
6.2. Die lokale Installation . . . . .	12
6.3. Veröffentlichen der Seite . . . . .	12
6.3.1. Veröffentlichung auf github . . . . .	13
6.3.2. Veröffentlichung auf einem Server mit git Unterstützung	13
6.3.3. Veröffentlichung auf einem Server mit FTP . . . . .	13
<b>7. Konfiguration</b>	<b>14</b>
7.1. Rumex auf dich einstellen . . . . .	14
7.2. Rumex Konfigurationsvariable . . . . .	14
<b>8. Diskussion</b>	<b>24</b>
8.1. moot.it . . . . .	24
<b>9. Weblog</b>	<b>25</b>

<b>10RSS Feed</b>	<b>26</b>
<b>11Vim Kurztasten</b>	<b>27</b>
11.1Textbausteine	27
11.1.1., rnd (RumexNeueDatei)	27
11.1.2., rnn (RumexNeueNachricht)	28
11.1.3., rwl (RumexWeiterLeitung)	28
11.1.4., rmk (RumexMootitKommentar)	29
11.1.5., rnb (RumexNeuenweBlog)	29
11.2Git Kommandos	29
11.2.1., rgp (RumexGitPull)	29
11.2.2., rgs (RumexGitStatus)	30
11.3Text Formatierung	30
11.3.1., rff (RumexFormatFett)	30
11.3.2., rfk (RumexFormatKursiv)	30
11.3.3., rfl (RumexFormatListe)	30
11.3.4., rfn (RumexFormatNummernliste)	30
11.3.5., rfb (RumexFormatBlock)	30
11.4Make Befehle	31
11.4.1., rma (RumexMakeAll)	31
11.4.2., rmb (RumexMakeBild)	31
11.4.3., rmca (RumexMakeCleanAll)	31
11.4.4., rmcb (RumexMakeCleanBilder)	31
11.4.5., rmch (RumexMakeCleanHtml)	31
11.4.6., rmcx (RumexMakeCleanXml)	31
11.4.7., rmc5 (RumexMakeCleanF8htm)	31
11.4.8., rmh (RumexMakeHtml)	32
11.4.9., rmi (RumexMakeIndex)	32
11.4.10rmm (RumexMakesiteMap)	32
11.4.11rmo (RumexMakeOnline)	32
11.4.12rmr (RumexMakeRss)	32
11.4.13rms (RumexMakeSuche)	32
11.5Vorschau	32
11.5.1., rsf (RumexShowFile)	32
11.5.2., rsw (RumexShowWww)	32
11.5.3., rsl (RumexShowLocal)	33
11.6Sonstiges	33
11.6.1., rku	33
11.6.2., ros (RumexOpenStart)	33
11.6.3., ror (RumexOpenRss)	33
<b>12Statik Funktion</b>	<b>34</b>
12.1Die (g)vim statik Kurztaste in Rumex	35
12.2HTML Formatierung	35

12.3	Die Literaturverzeichnis Funktion . . . . .	36
12.4	Installation pandoc manuell . . . . .	36
12.5	Verwendung in- und außerhalb von Rumex? . . . . .	38
12.5.1	Setzen der Überschrift für das Literaturverzeichnis . . . . .	39
12.5.2	Einbinden von Bildern . . . . .	39
12.5.3	Statik Dateien im .rx Verzeichnis . . . . .	39
12.5.4	Tipps . . . . .	40
12.6	Literaturverzeichnis . . . . .	40
<b>13</b>	<b>Lager</b>	<b>41</b>
13.0.1	gvim unterstützung einrichten . . . . .	41
13.0.2	seite auf einem github.com server einrichten . . . . .	42
13.0.3	seite auf einem nicht github.com server einrichten . . . . .	43
13.1	aufbau des rumex baukastens . . . . .	43
13.1.1	root verzeichnis . . . . .	43
13.1.2	unterverzeichnisse . . . . .	44
13.1.3	steuerung durch dateiendung . . . . .	44
13.1.4	sonderseiten . . . . .	45
13.1.5	aufbau der startseite . . . . .	46
13.1.6	aufbau der einzelseiten . . . . .	46
13.1.7	template . . . . .	47
13.2	rss feed funktion . . . . .	47
13.2.1	beispiel einer rss feed seite . . . . .	48
13.2.2	rss dateiname . . . . .	48
13.2.3	rss titel . . . . .	48
13.2.4	rss auslagern . . . . .	48
13.2.5	rss kurzaste . . . . .	48
13.2.6	interna . . . . .	49
13.3	rumex auf einem usb stick . . . . .	49
13.4	make steuerung . . . . .	49
13.5	update . . . . .	50
13.6	homepage änderung, schnell und immer aktuell . . . . .	50
13.6.1	installation der tilda unterstützung . . . . .	50
13.6.2	tilda einrichten . . . . .	52
13.7	Warum wurde diese Beschreibung nicht mit Rumex erstellt . . . . .	53
<b>A.</b>	<b>Lizenz</b>	<b>54</b>
<b>B.</b>	<b>Impressum / Kontakt</b>	<b>55</b>
<b>C.</b>	<b>Literatur</b>	<b>56</b>
	<b>Literatur</b>	<b>57</b>

# 1. Vorwort

Es gibt immer mehr Systeme zum Erstellen und Verwalten von Internetseiten. Probiert habe ich schon viele und mit den meisten war ich auch sehr zufrieden. Was mich aber immer gestört hat waren die ständigen Updates.

## **Irgendwie hat sich alles immer gebissen**

Einmal brauchte das, ich nenne es stellvertretend für alle Systeme, CMS eine besondere Version eines Programms. Dann war es wieder umgekehrt ein Update des Programms konnte nicht gemacht werden weil das CMS noch nicht damit zurecht kam.

Ab und zu kam es mir so vor als ob ich die meiste Zeit damit verbrachte das CMS und dessen Plattform dazu zu überreden mit einander zu arbeiten.

Ich wünschte mir immer mehr eine einfache Internetseite, eine HTML Seite, so wie in früheren Zeiten.

Irgendwann bin ich dann auf markdown und pandoc gestoßen und die Rumex Idee ist entstanden.

## **1.1. Rumex?**

Rumex ist die lateinische Bezeichnung für den **Ampfer** und dieser taucht in der Natur dann auf, wenn der Boden Überdüngt, Verdichtet und Beschädigt ist. Rumex gehört zu den sogenannten Pionier Pflanzen. Er ist ein Lückenfüller.

Genau das soll Rumex auch sein «ein Lückenfüller» für alle die ... Systeme satt haben.

## **2. Kurzbeschreibung**

### **3. Die Index Seite**

## 4. Seiten Arten

Die Verwendung der Seiten wird über die Dateiendung gesteuert. Hierzu werden vier Zeichen eingesetzt.

- Die ersten beiden Zeichen rx kennzeichnen eine Rumex Datei
- Das dritte Zeichen, immer eine Zahl, kennzeichnet ob die Datei ein Inhaltsverzeichnis besitzt und bis welcher Ebene dieses eingebunden werden soll.
- Das vierte Zeichen zeigt die Verwendung, sprich Einbindung an.

### 4.1. Inhaltsverzeichnis Kennzeichnung

**0** Datei ohne Inhaltsverzeichnis.

**1** — Datei mit Inhaltsverzeichnis bis zur ersten Ordnung.

**2** — Datei mit Inhaltsverzeichnis bis zur zweiten Ordnung.

**3** — Datei mit Inhaltsverzeichnis bis zur dritten Ordnung.

**4** — Datei mit Inhaltsverzeichnis bis zur vierten Ordnung.

**5** — Datei mit Inhaltsverzeichnis bis zur fünften Ordnung.

**6** — Datei mit Inhaltsverzeichnis bis zur sechsten Ordnung.



## **4.2. Verwendungskennzeichnung**

### **4.3. rx0w**

### **4.4. rx0x**

# **5. Sonderseiten**

**5.1. index**

**5.2. suche**

**5.3. start**

**5.4. weblog**

**5.5. sitemap**

**5.6. rss**

# 6. Installation

Rumex kann in zwei Stufen verwendet werden. Einmal als lokale, private Installation<sup>1</sup> und als öffentliche Installation.

## 6.1. Vorbereitung des Rechners

Rumex ist auf ein \*nix System ausgerichtet. Auf diesem sollten folgende Programme installiert sein:

- bash
- make
- perl
- git
- pandoc
- imagemagick
- wget
- sitecopy
- vim (g)vim
- texlive

Wer mit dem Editor vim zurecht kommt sollte sich auch gvim installieren. Rumex besitzt eine gvim Erweiterung die, die Arbeit bzw. die Suche nach dem richtigen Befehl am Anfang um einiges erleichtert.

texlive wird nur gebraucht wenn man auch PDF Dateien erstellen möchte.

---

<sup>1</sup>Rumex braucht generell keine online Verbindung. Alle benötigten öffentlichen Skripte werden statisch gespeichert, sind in der Installation enthalten. Nicht einmal ein Webserver muss installiert werden.

sitecopy wird nur gebraucht wenn man die Daten per FTP hoch laden möchte.

Bei Debian dürfte dies die nachfolgender Zeile erledigen.

```
sudo apt-get install make perl git-core \  
pandoc imagemagick sitecopy gvim texlive
```

## 6.2. Die lokale Installation

Für die Installation auf deinem Rechner musst zu erst das ZIP bzw. das tar.gz Archiv vom github Server holen und entpacken werden.

```
wget https://github.com/itbayer/rumex/archive/gh-pages.zip  
unzip gh-pages.zip
```

bzw.

```
wget https://github.com/itbayer/rumex/archive/gh-pages.tar.gz  
tar -xzf gh-pages.tar.gz
```

Nach dem entpacken wechselt man in das Verzeichnis rumex-gh-pages/.rx und starte die Befehle make install und make show.

```
cd rumex-gh-pages/.rx/  
make install  
make show
```

Rumex ist jetzt lokal installiert und zeigt die erste Seite an. Auf dieser findet man eine kurz Information der Schritte die man noch machen sollte. Diese Schritte sind vor allen wichtig wenn man seine Seiten veröffentlichen möchte.

## 6.3. Veröffentlichen der Seite

In der zweiten Stufen kann die Seite auf drei Arten veröffentlicht werden.

1. Hosting auf [Github](#)
2. Hosting auf einem Server mit git Unterstützung
3. Hosting klassisch per FTP

**6.3.1. Veröffentlichung auf github**

**6.3.2. Veröffentlichung auf einem Server mit git Unterstützung**

**6.3.3. Veröffentlichung auf einem Server mit FTP**

# 7. Konfiguration

## 7.1. Rumex auf dich einstellen

Nach der Installation muss Rumex noch auf dich eingestellt werden. Genauer gesagt sollten folgende Angabe für deine neuen Seite angepasst werden.

- Url deiner Seite
- Impressum
- Kopf / Fusszeile
- Logo

Eine Kurzbeschreibung findest du, nach der Installation, auf der ersten Seite die dir Rumex zeigt.

## 7.2. Rumex Konfigurationsvariable

Die Konfiguration von Rumex wird über zwei Dateien gesteuert. Zu einem ist das die Vorgabe Datei `.rumex/makefile/config.mk` und die Benutzerdatei `.rx/config.mk`, wobei die Einträge der Vorgabe Datei `.rumex/makefile/config.mk` von den Einträgen der Benutzer Datei `.rx/config.mk` überschrieben werden.

Es sind auch nicht alle Einträge in der Benutzerdatei vorhanden. Für erweiterte Einstellungen muss man Einträge aus der Vorgabedatei in die Benutzerdatei kopieren und dementsprechend anpassen.

Nach der Installation wird eine einfache Benutzerdatei in das Verzeichnis `.rx` kopiert.

**URL** Definiert die URL über die die Seite erreichbar ist. Die URL muss hier ohne abschließendes `/` eingegeben werden.

Diese Variable ist in der Vorlage vorhanden.  
Diese Variable ist in der Benutzerdatei vorhanden.

```
URL="http://www.it-bayer.de/rumex"
```

**INDEX\_TITEL** Definiert den Titel der Seite. Es können auch HTML Tags verwendet werden.

Diese Variable ist in der Vorlage vorhanden.  
Diese Variable ist in der Benutzerdatei vorhanden.

```
INDEX_TITEL = "IT Bayer's rumex <sup style=\"font-size: .4em;\"> (github.com Version)</sup>"
```

**INDEX\_AUTOR** Definiert den Autor der Seite.

Diese Variable ist in der Vorlage vorhanden.  
Diese Variable ist in der Benutzerdatei vorhanden.

```
INDEX_AUTOR = "Stefan Blechschmidt"!
```

**INDEX\_DATUM** Definiert den Stand, Datum der Seite.

Diese Variable ist in der Vorlage vorhanden.

```
INDEX_DATUM = $(shell ls index.rx0x -l --time-style=+%Y-%m-%d | awk '{print $6}')
```

**RSS\_TITEL** Definiert den Titel der RSS Seite.

Diese Variable ist in der Vorlage vorhanden.  
Diese Variable ist in der Benutzerdatei vorhanden.

```
RSS_TITEL = "Neuigkeiten von rumex Baukasten"
```

**CSSALL** Definiert die CSS Datei für die Formatierung aller Seiten.

Diese Variable ist in der Vorlage vorhanden.

```
CSSALL = "rxtpl/css/all.css"
```

**CSSSCREEN** Definiert die CSS Datei für die Formatierung der Bildschirmausgabe.

Diese Variable ist in der Vorlage vorhanden.

```
CSSSCREEN = "rxtpl/css/screen.css"
```

**CSSPRINT** Definiert die CSS Datei für die Formatierung der Druckausgabe.

Diese Variable ist in der Vorlage vorhanden.

```
CSSPRINT = "rxtpl/css/print.css"
```

**CSSHANDHELD** Definiert die CSS Datei für die Formatierung Mobiler Endgeräte.

Diese Variable ist in der Vorlage vorhanden.

```
CSSHANDHELD = "rxtpl/css/handheld.css"
```

**SEITENBANNER** Definiert die CSS Datei für das Seitenbanner.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

```
SEITENBANNER = "rxtpl/img/rumex.png"
```

**MOOTIT** Kontoname für die Diskussionserweiterung mittels moot.it. Wird diese Variable kommentiert ist diese Erweiterung ausgeschaltet.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

```
# MOOTIT = "rumex"
```

**WEBLOGAUTOR** Diese Variable definert den Autor der Weblog Einträge

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

```
WEBLOGAUTOR = "IT-Bayer"
```

**META\_PUBLISHER** Angaben zum Autor der HTML Seite.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

```
META_PUBLISHER = "IT-Bayer"
```



**META\_CREATOR**    Angaben zum Autor der HTML Seite.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

`META_CREATOR = "IT-Bayer (Stefan Blechschmidt)"`

**RUMEXSUCHE**    Verzeichnis in dem sich das JavaScript für die Suche befindet.

Diese Variable ist in der Vorlage vorhanden.

`RUMEXSUCHE = "rxtpl/js"`

**FAVICON**    Definiert das Favicon Bild der Seite

Diese Variable ist in der Vorlage vorhanden.

`FAVICON = "favicon.gif"`

**SITECOPY**    Definiert das Programm für die FTP Upload Funktion. Ist diese Variabel kommentiert wird git verwendet.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

`#SITECOPY = /usr/bin/sitecopy`

Die **TEMPLATE** Dateien definieren den Seitenaufbau. Hierzu werden vier Dateien verwendet.

1. **HTML\_TEMPLATE**
2. **KOPF\_TEMPLATE**
3. **HEADER\_TEMPLATE**
4. **FUSS\_TEMPLATE**

Ist im Verzeichnis rx/ eine entsprechende Datei vorhanden wird diese verwendet. Ansonst wird die Datei aus dem Verzeichnis .rumex/default eingebunden.

**HTML\_TEMPLATE**    HTML Vorlage Datei für die Seite

Diese Variable ist in der Vorlage vorhanden.

```
HTML_TEMPLATE = $(shell if [ -f ../rx/html.template ];
then echo \"../rx/html.template\";
else echo \"../rumex/default/html.template\";
fi)
```

**KOPF\_TEMPLATE** Definiert die Kopfdatei der Seite.

Diese Variable ist in der Vorlage vorhanden.

```
KOPF_TEMPLATE = $(shell if [ -f ../rx/kopf.html ];
then echo \"../rx/kopf.html\";
else echo \"../rumex/default/kopf.html\";
fi)
```

**HEADER\_TEMPLATE** Definiert die Datei in der Header Einträge der Seite vorhanden sind.

Diese Variable ist in der Vorlage vorhanden.

```
HEADER_TEMPLATE = $(shell if [ -f ../rx/header.html ];
then echo \"../rx/header.html\";
else echo \"../rumex/default/header.html\";
fi)
```

**FUSS\_TEMPLATE** Definiert die Datei der Informationen des Fußbereiches.

Diese Variable ist in der Vorlage vorhanden.

```
FUSS_TEMPLATE = $(shell if [ -f ../rx/fuss.html ];
then echo \"../rx/fuss.html\";
else echo \"../rumex/default/fuss.html\";
fi)
```

Jede der drei Seitentypen bekommt in HEADER der HTML Datei die `meta_robots` Variable mitgeliefert. Anhand der nachfolgenden drei Konfigurationsvariablen werden diese Parameter für jeden Seitentyp gesetzt.

**META\_ROBOTS\_STANDARD** Meta Angabe für die Standard Seiten rx?s.

Diese Variable ist in der Vorlage vorhanden.

```
META_ROBOTS_STANDARD = "all"
```

**META\_ROBOTS\_VERSTECKT** Meta Angabe für die versteckten Seiten rx?v.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

META\_ROBOTS\_VERSTECKT = "noindex,nofollow,noarchive"

**META\_ROBOTS\_WEITERLEITUNG** Meta Angabe für die Seiten die weiterleiten rx?w.

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

META\_ROBOTS\_WEITERLEITUNG = "noindex, follow,noarchive"

**INDEX** Programm zum erstellen der index.rx0x Datei

Diese Variable ist in der Vorlage vorhanden.

INDEX="../.rumex/bin/index.pl"

**RX2RSS** Programm zum Erstellen der RSS Feed Datei.

Diese Variable ist in der Vorlage vorhanden.

RX2RSS = "../.rumex/bin/rx2rss.pl"

**RSS\_FILE** Name der RSS Datei die in den Quelltext eingebaut wird

Diese Variable ist in der Vorlage vorhanden.

Diese Variable ist in der Benutzerdatei vorhanden.

RSS\_FILE = "rss.xml"

```
# Externe RSS URL verwenden
# Ist diese Variable gesetzt wird die interne
# RSS Verarbeitung, RSS_FILE, übersprungen.
# und die externe URL verwendet.
```

```
#RSS_EXTERN = "http://www.it-bayer.de/rss.xml"
```

```

\konfig{}
{}
{Diese Variable ist in der Vorlage vorhanden.}
{Diese Variable ist in der Benutzerdatei vorhanden.}
{\scriptsize
\begin{Verbatim}

\end{Verbatim}
}

```

```

#-----
# XML Sitemap
#-----
#
# Programm zum erstellen der XML Sitemap Datei
SITEMAP_XML = "../rumex/bin/sitemap_xml.pl"

```

```

\konfig{}
{}
{Diese Variable ist in der Vorlage vorhanden.}
{Diese Variable ist in der Benutzerdatei vorhanden.}
{\scriptsize
\begin{Verbatim}

\end{Verbatim}
}

```

```

# Zieldatei der Sitemap Datei
SITEMAP_XML_FILE = "../sitemap.xml"

```

```

\konfig{}
{}
{Diese Variable ist in der Vorlage vorhanden.}
{Diese Variable ist in der Benutzerdatei vorhanden.}
{\scriptsize
\begin{Verbatim}

\end{Verbatim}
}

```

```
}
```

```
# Programm zum erstellen der rx0v Sitemap Datei  
SITEMAP_RX = "../rumex/bin/sitemap_rx.pl"
```

```
\konfig{}  
{}  
{Diese Variable ist in der Vorlage vorhanden.}  
{Diese Variable ist in der Benutzerdatei vorhanden.}  
\scriptsize  
\begin{Verbatim}  
  
\end{Verbatim}  
}
```

```
# Zieldatei der HTML Sitemap Datei  
SITEMAP_RX_FILE = "sitemap.rx0v"
```

```
\konfig{}  
{}  
{Diese Variable ist in der Vorlage vorhanden.}  
{Diese Variable ist in der Benutzerdatei vorhanden.}  
\scriptsize  
\begin{Verbatim}  
  
\end{Verbatim}  
}
```

```
#-----  
# Verzeichnis in dem sich die Bilder befinden.  
#-----  
BVZ="../bilder"  
\konfig{}  
{}  
{Diese Variable ist in der Vorlage vorhanden.}  
{Diese Variable ist in der Benutzerdatei vorhanden.}  
\scriptsize  
\begin{Verbatim}  
  
\end{Verbatim}
```

```
}
```

```
#-----  
# Pandoc Programm  
#-----  
PANDOC = pandoc  
# PANDOC = /usr/bin/pandoc  
  
\konfig{}  
{}  
{Diese Variable ist in der Vorlage vorhanden.}  
{Diese Variable ist in der Benutzerdatei vorhanden.}  
{\scriptsize  
\begin{Verbatim}  
  
\end{Verbatim}  
}
```

```
#-----  
# Rumex Suche  
#  
# Programm zum Erstellen der rumexsuche_config.js  
#-----  
SUCHE = "../.rumex/bin/suche.pl"  
  
\konfig{}  
{}  
{Diese Variable ist in der Vorlage vorhanden.}  
{Diese Variable ist in der Benutzerdatei vorhanden.}  
{\scriptsize  
\begin{Verbatim}  
  
\end{Verbatim}  
}
```

```
#-----  
# Ziel Datei für die JavaScript Suche  
#-----
```

```

SUCHE_JS_CONFIG="../$(RUMEXSUCHE)/rumexsuche_config.js"
\konfig{}
{}
{Diese Variable ist in der Vorlage vorhanden.}
{Diese Variable ist in der Benutzerdatei vorhanden.}
{\scriptsize
\begin{Verbatim}

\end{Verbatim}
}

```

```

#-----
# Rumex Versionshinweis für die HTML Dateien
#-----
META_GENERATOR = "rumex "$(shell cat ../.rumex/default/version.txt)
\konfig{}
{}
{Diese Variable ist in der Vorlage vorhanden.}
{Diese Variable ist in der Benutzerdatei vorhanden.}
{\scriptsize
\begin{Verbatim}

\end{Verbatim}
}

```

# 8. Diskussion

## 8.1. moot.it

@todo: hier fehlt noch die Beschreibung

- ein/ausschalten
- Kontoname einstellen



## **9. Weblog**

## **10. RSS Feed**

# 11. Vim Kurztasten

Alle Rumex Kurztasten, für den Editor VIM, beginnen mit einem ,r<sup>1</sup>. Für die Bedienung von Rumex braucht man nur ein paar. Viele der Kurztasten beinhalten eine Kombination einzelner Kurztasten bzw. Befehle.

Am Anfang ist es sicher einfacher GVIM zu verwenden, da hier ein Menü eingebaut ist welches unter anderem auch die Kurztasten anzeigt.

Wer jedoch mit der **tilda** Unterstützung arbeitet sollte sich schon ein paar Kurztasten einprägen.

## 11.1. Textbausteine

Mittels dieser Kurztasten können Textbausteine eingebunden werden.

### 11.1.1. ,rnd (RumexNeueDatei)

Erstellt ein neues Datei Gerüst. Dabei wird der Dateiname gleich mit eingebunden.

**So schaut's aus**

```
% vim-kurztasten
%
%
```

```
<!--
```

```
[vim-kurztasten](vim-kurztasten.html)
```

```
=====
```

---

<sup>1</sup>Ausnahmen bestätigen die Regel. So wurde für die zwei spaltige Darstellung ,spn2 verwendet.

Vortext INDEX

-->

Vortext INDEX und SEITE

<!-- schnipp -->

TEXT DER SEITE

<!-- vim: set ft=pandoc: -->

### **11.1.2. ,rnn (RumexNeueNachricht)**

Erstellt einen neuen News Eintrag. Diese Kurztaste macht eigentlich nur in der Datei `rss.rx0x` Sinn. Das Datum hinter dem Titel, Überschrift 1, wird von Rumex gesetzt, ist also immer die aktuelle Systemzeit beim ausführen von `,rnn`.

```
# Neue Nachricht <!-- 2013/11/10 00:30 -->
```

```
<!--
```

```
!> Link: http://www.it-bayer.de/rumex/
```

```
!> Autor: IT-Bayer
```

```
!> Kategorie: Neues
```

```
-->
```

Ab hier geht die neuen Nachricht los.

### **11.1.3. ,rw1 (RumexWeiterLeitung)**

Erstellt eine neue Weiterleitungsseite.

**So schaut's aus**

```
% Weiterleitung nach ....html
```

```
%
```

```
%
```

```
<script language="javascript">
```

```
<!--  
//window.location.href="....html";  
// -->  
</script>
```

Anmerkung: Die Zeile `window.location.href="....html";` wurde hier Kommentiert da sonst die javascript Weiterleitung greift. Normalerweise findet man kein `//` vor der Zeile `window.location.href=....html`.

#### **11.1.4. ,rmk (RumexMootitKommentar)**

Erstellt einen Moot.it Kommentar Abschnitt. Als Kennzeichnung wird der Dateiname ohne `rx??` eingebaut / angehängt.

```
# Kommentare
```

```
<a class="moot" href="https://moot.it/i/rumex/blog/vim-kurztasten"></a>
```

#### **11.1.5. ,rnb (RumexNeuenweBlog)**

Erstelle einen neuen Weblog Eintrag.

```
# Rumex WebLog
```

```
_am 09.09.2013 um 15:21 schrieb IT-Bayer_
```

Text für den Eintrag

```
<div class="weblog">  
Text der vorerst ausgeblendet ist.  
</div>
```

## **11.2. Git Kommandos**

### **11.2.1. ,rgp (RumexGitPull)**

Git pull

### **11.2.2. ,rgs (RumexGitStatus)**

Git status

## **11.3. Text Formatierung**

### **11.3.1. ,rff (RumexFormatFett)**

Markierte Textstelle fett darstellen

### **11.3.2. ,rfk (RumexFormatKursiv)**

Markierte Textstelle kursiv darstellen

### **11.3.3. ,rfl (RumexFormatListe)**

Markierte Zeilen in eine Liste wandeln

### **11.3.4. ,rfn (RumexFormatNummernliste)**

Markierte Zeile in eine Nummernliste wandeln

### **11.3.5. ,rfb (RumexFormatBlock)**

Markierte Zeile in einen Block wandeln

**,rfc** Markierte Zeile als Code darstellen

**,spn2** Text mit 2er Spalten Formatierung umschließen

**,spn3** Text mit 3er Spalten Formatierung umschließen

**,spn4** Text mit 4er Spalten Formatierung umschließen

## **11.4. Make Befehle**

### **11.4.1. ,rma (RumexMakeAll)**

Speichert alle offenen Dokumente und erstellt daraus die HTML Datei.

### **11.4.2. ,rmb (RumexMakeBild)**

Erstellt die unterschiedlichen Bildgrößen

### **11.4.3. ,rmca (RumexMakeCleanAll)**

Löscht alle html Dateien, alle Bildergrößen und alle xml Dateien. beinhaltet die drei Befehle make bclean, make hclean, make xclean.

### **11.4.4. ,rmcb (RumexMakeCleanBilder)**

Löscht alle Bildgrößen aus dem bilder/ Verzeichnis. Es werden nur die Bilder gelöscht die von Rumex erstellt wurden siehe make bilder.

### **11.4.5. ,rmch (RumexMakeCleanHtml)**

Löscht alle .html Dateien die von Rumex erstellt wurden.

### **11.4.6. ,rmcx (RumexMakeCleanXml)**

Löscht alle .xlm Dateien die von Rumex erstellt wurden.

### **11.4.7. ,rmcf5 (RumexMakeCleanF8htm)**

Löscht alle .htm Dateien die mittels der Gvim Taste F5, siehe [HTML und PDF Dateien mit pandoc und gvim erstellen](#), erstellt wurden.

#### **11.4.8. ,rmh (RumexMakeHtml)**

Erstellt die .html Dateien.

#### **11.4.9. ,rmi (RumexMakeIndex)**

Erstellt die index Datei.

#### **11.4.10. ,rmm (RumexMakesiteMap)**

Erstellt die sitemap Datei.

#### **11.4.11. ,rmo (RumexMakeOnline)**

Speichert alle Dateien und stellt diese online.

#### **11.4.12. ,rmr (RumexMakeRss)**

Erstellt die rss Datei.

#### **11.4.13. ,rms (RumexMakeSuche)**

Erstellt die Rumex Suche, bzw. die Dateiliste die für die Suche verwendet werden soll.

### **11.5. Vorschau**

#### **11.5.1. ,rsf (RumexShowFile)**

Zeigt eine Vorschau der Seite der Datei file:/// im Standardbrowser.

#### **11.5.2. ,rsw (RumexShowWww)**

Zeigt eine Vorschau der Seite **Online** im Standardbrowser.



### **11.5.3. ,rsl (RumexShowLocal)**

Zeigt eine Vorschau der Seite auf dem eigenen Rechner. Diese Kurztaste kann aber nur verwendet werden wenn auf dem Rechner der Apache installiert und entsprechend eingerichtet ist.

## **11.6. Sonstiges**

### **11.6.1. ,rku**

Öffnet das Unterverzeichnis .rx in einem neuen VimTab Fenster. Keine Ahnung warum ich diese Kurztaste so benannt habe.

### **11.6.2. ,ros (RumexOpenStart)**

Öffnet die start.rx0s in einen neuen VIM Tab Fenster.

### **11.6.3. ,ror (RumexOpenRss)**

Öffnet die rss.rx0x in einen neuen VIM Tab Fenster.

## 12. Statik Funktion

Rumex verwendet pandocs markdown weil man damit sehr einfach und schnell Text erstellen und in verschiedenen Formate wandeln kann. Für die Erstellung von Denkschriften<sup>1</sup> wurde zusätzlich eine, ich nenne sie *statik Funktion* eingebaut. Mit dieser Funktion ist es möglich innerhalb eines Unterverzeichnisses verschiedenen Ausgabe Formate zu erstellen. Zur Zeit werden, von Rumex, folgenden Formate unterstützt:

- .html
- .pdf
- .odt OpenDokument
- .epub E-Book
- .mobi E-Book (Kindle)

Erstellt werden die einzelnen Formate über **Funktionstasten** die innerhalb (g)vim<sup>2</sup> zur Verfügung stehen. Eine Besonderheit ist dass die .htm Datei auch ohne die zusätzlichen Dateien wie Bilder oder die CSS Datei funktionieren. Alle Daten werden in die .htm Datei eingebunden.<sup>3</sup>

Auch wurde die **Literaturfunktion** von Pandoc eingebaut sodass Verweise auf anderen Quellen in den Denkschriften verwendet werden können.

Wer sich Rumex nicht installieren möchte aber dennoch eine einfache Möglichkeit sucht HTML Seiten zu erstellen, kann sich einmal **cirsium** anschauen. Cirsium ist eine Auskopplung aus Rumex, unterteilt aber die Quelltextdatei in verschiedene Abschnitte. Am besten einfach mal auf [github.com](https://github.com) anschauen.

---

<sup>1</sup>Im Neudeutschen würde man die Denkschrift als Memorandum bezeichnen.

<sup>2</sup>Die Funktionstasten sind im gvim Rumex Menü nicht eingebaut. Man sollte sie sich also merken :-).

<sup>3</sup>Diese Eigenschaft wurde wieder entfernt da eine Datei mit sehr vielen Bildern oder gar Videos, die alle in die HTML Datei eingebunden werden, sehr groß wird. Dieses erhöht die Ladezeit der HTM Datei. Bei der Verwaltung durch git wirkt sich diese Eigenschaft auch ungünstig aus.

## 12.1. Die (g)vim statik Kurztaste in Rumex

Ab der Rumex Version 0.8.2 sind die Funktionstasten in Rumex enthalten. Folgende F-Tasten wurden belegt.

**F5** Erstellt die .htm Datei ohne Inhaltsverzeichnis.

**Shift+F5** Erstellt die .htm Datei mit Inhaltsverzeichnis bis zur Ebene 3.

**Alt+F5** Erstellt die .htm Datei mit kompletten Inhaltsverzeichnis bzw. bis Ebene 5 da LaTeX auch nicht mehr Ebenen unterstützt und die beiden Ausgabe Format annähernd identisch sein sollten.

**Ctrl+F5** Öffnet die .htm Datei.

**F6** Erstellt die .pdf Datei ohne Inhaltsverzeichnis.

**Shift+F6** Erstellt die .pdf Datei mit Inhaltsverzeichnis bis zur Ebene 3.

**Alt+F6** Erstellt die .pdf Datei mit kompletten Inhaltsverzeichnis. Hier ist anzumerken dass LaTeX nur Inhaltsverzeichnisse bis zur Ebene 5 unterstützt. Das HTML Format wurde entsprechend angepasst, siehe oben.

**Ctrl+F6** Öffnet die .pdf Datei. Zur Zeit wird nur zathura unterstützt.

**F7** Erstellt die restlichen Formate, .epub, .odt und .mobi. Voraussetzung für das .mobi Format ist calibre.

**Ctrl-F7** Öffnet die Literatur Verwaltung rumex.bib. Voraussetzung, jabref ist installiert.

## 12.2. HTML Formatierung

Die erzeugte HTML Datei besitzt Standardmäßig keine Formatierung bzw. verwendet die Standard Darstellung des Browsers.

Kopf- und Fusszeile werden dadurch nicht, vom restlichen Text, unterschieden. Auch das Inhaltsverzeichnis ist im ersten Moment als solches nicht gleich zu erkennen. Dieses kann mit ein wenig CSS geändert werden. Diese CSS Datei ist ab Rumex Version 0.8.2 enthalten muss aber unter Umständen noch eingerichtet werden.

```
cd .rx
ln -s ../.rumex/default/statik.css statik.css
```

## 12.3. Die Literaturverzeichnis Funktion

Beim Lesen des Artikels *“PDF-Dokumente schreiben mit Pandoc und Markdown”* [stenderprolinux] ist mir die Idee gekommen die Rumex statik Funktion mit einem Literaturverzeichnis, die ja auch in pandoc zur Verfügung steht, zu versehen.

## 12.4. Installation pandoc manuell

Für die Verwendung der Literaturfunktion muss pandoc um das Zusatzprogramm pandoc-citeproc erweitert werden. Wer Pandoc über die Paketverwaltung installiert hat braucht hier nichts zu machen. Wer Pandoc manuell, so wie ich, installiert hat muss dieses Programm nachinstallieren.

Dazu erweitert man die Installationszeile um das neue Programm

```
cabal update
cabal install pandoc pandoc-citeproc
```

Zu guter Letzt erstellt man noch die symbolischen Links der beiden Programme.

```
sudo ln -s /home/USER/.cabal/bin/pandoc /usr/local/bin/.
sudo ln -s /home/USER/.cabal/bin/pandoc-citeproc /usr/local/bin/.
```

### Nachinstallation Rumex

Wer Rumex schon im Einsatz hat muss für die Erweiterung ein wenig Hand anlegen. Zu erste holt man sich die neue Version<sup>4</sup> von rumex.

Dann braucht man noch drei zusätzliche Dateien im Verzeichnis .rx.

- rumex.bib
- rumex.csl
- statik.css

Wobei der **Literatur Vorlage Stiel** und die CSS Datei nur verlinkt wird. In der rumex.bib werden dann die Literatur Verweise verwaltet.

---

<sup>4</sup>Die Literatur Erweiterung ist ab der Rumex Version 0.8.2 enthalten.

```
cd .rx
touch rumex.bib
ln -s ../.rumex/default/din-1505-2.csl rumex.csl
ln -s ../.rumex/default/statik.css statik.css
```

## Literatur Stil

Als Literatur Stil kommt `din-1505-2.csl` zum Einsatz. Andere Stile findet man im git Repository <https://github.com/citation-style-language/styles.git>. Als Name für die Stil Vorlage wurde `rumex.csl` gewählt damit mit eine Änderung des Stils einfach über den Symlink gemacht werden kann.

## Literatur Verwaltung

Für die Verwaltung der Literatur Datenbank verwende ich [Jabref](#).

```
sudo apt-get install jabref
```

Der Aufruf des Programms wurde auch auf einen F Taste gelegt. Wer eine anderes Programm verwenden will muss diesen entsprechend anpassen.

## Tipps

- Auf [@wiki:bibtex](#) findet man eine schöne Beschreibung über die Literatur Verwaltung mit BibTeX.
- Die BibTeX Einträge muss man sich unter Umständen nicht einmal selber erstellen. Da verschiedene Seiten entsprechende Dienste anbieten. Gelungen finde ich die Seite von <http://www.literatur-generator.de/> aber auch auf <http://lead.to/amazon/en/?op=bt> findet man BibTeX Einträge. Zwar muss man diese unter Umständen noch ein wenig überarbeiten aber das Grund Gerüst wird einem sozusagen frei Haus geliefert.  
Wer einen Wikipedia Artikel zitieren dem wird unter "Werkzeuge -> Seite zitieren" weiter geholfen.
- Für das zitieren von Internetseiten verwende ich folgende Formate, siehe dazu Abschnitt [Literaturverzeichnis](#).

```
@ELECTRONIC{ wiki:bibtex,
  author = "Wikipedia",
  title = "BibTeX --- Wikipedia{,} Die freie Enzyklopädie",
  year = "2013",
  url = "http://de.wikipedia.org/w/index.php?title=BibTeX&oldid=124",
  note = "[Online; Stand 18. Dezember 2013]"
}
```

...oder

**Achtung:** Das Formate @WWW wird von jabref nicht unterstützt und gegen ein anders ausgetauscht. Bei dem Einsatz von Jabref am besten @ELECTRONIC verwenden.

```
@WWW{ wiki:bibtex,
  author = "Wikipedia",
  title = "BibTeX --- Wikipedia{,} Die freie Enzyklopädie",
  year = "2013",
  url = "http://de.wikipedia.org/w/index.php?title=BibTeX&oldid=124",
  note = "[Online; Stand 18. Dezember 2013]"
}
```

## 12.5. Verwendung in- und außerhalb von Rumex?

Innerhalb von Rumex erstellt man in einem separatem Unterverzeichnis die entsprechende markdown Datei und dann kann es auch schon los gehen.

Außerhalb von Rumex kann man diese Funktion natürlich auch verwenden. Mit Außerhalb meine ich Denkschriften die nicht veröffentlicht werden. Dazu gibt es zwei Möglichkeiten.

1. Die Datei bzw. das Verzeichnis in .gitignore hinterlegen. Somit wird diese nicht verwaltet und auch nicht, bei einem make online, hoch geladen.
2. Eine zweite lokale Rumex Installation die nur für Denkschriften verwendet wird.
3. ... und dann wäre da noch **Cirsium**, eine Auskopplung aus Rumex, mit der man einfache html und pdf Seiten erstellen kann. Die Literaturverzeichnisoption ist auch enthalten. Die Formate odt, epub und mobi sind jedoch nicht eingebaut.

### **12.5.1. Setzen der Überschrift für das Literaturverzeichnis**

Die Überschrift für das Literaturverzeichnis muss immer am Ende des Artikels gesetzt werden.

#### **Beispiel**

```
# Literaturverzeichnis
```

### **12.5.2. Einbinden von Bildern**

Bei dem Einbinden der Bilder muss man beachten dass die Erstellung der statik Datei vom Verzeichnis `.rx` ausgeht.

Will man also ein Bild, dass im Ordner der Statik Datei liegt einbinden so muss auch auf das Bild aus der Sicht des `.rx` Verzeichnisses eingebunden werden.

Beispiel:

Das Bild liegt im Ordner `statik` somit müsste der Markdown Befehl so aussehen.

```
![Beispielbild](../statik/beispiel.png)
```

In Rumex kann man diese Funktion natürlich auch verwenden. Am besten erstellt man sich dazu ein eigenes Unterverzeichnis und dort die Datei `index.md` mit den Texten.

### **12.5.3. Statik Dateien im `.rx` Verzeichnis**

Es wird sicher passieren dass man die Funktionstasten der Statik Seiten auch bei der Bearbeitung der eigentlichen Rumex Dateien drückt. Durch entsprechende Einträge in der `.gitignore` Datei werden solche Dateien von einem Upload ausgeschlossen. Mit den Aufruf von `make statikclean` können die erstellten statik Dateien Schlussendliche aus dem `.rx` Verzeichnis entfernt werden. Dieser Befehl wird auch bei `make clean` ausgeführt.

### 12.5.4. Tipps

Das PDF Programm zathura hat die Eigenschaft dass wenn sich die Datei ändert diese automatisch nachgeladen wird. Eine schöne Funktion wenn man seinen Text, an dem man gerade arbeitet, immer wieder einmal im Ausgabe Format betrachten will. Einfach die Taste F6 drücken, die Datei wird auch gleich gespeichert, und mit ALT-TAB das Programm Fenster wechseln.

Bei Format HTML geht das natürlich auch. Nur muss hier eine Erweiterung installiert werden. Für die Browser Chromium und Firefox habe ich mit Auto Refresh Plus<sup>Chromium</sup> bzw. Tab Auto Reload<sup>Firefox</sup> gute Erfahrungen gemacht.

---

Die PDF Datei dieser Beschreibung kann man sich [hier](#) ansehen. Die Markdown Quelldatei kann man sich [hier](#) holen.

---

## 12.6. Literaturverzeichnis



# 13. Lager

Zwischenlager der Texte die noch nicht aufgearbeitet wurden.

## 13.0.1. gvim unterstützung einrichten

@todo: hier fehlt noch die beschreibung@

- erstellen der startdatei
- einrichten

### tipps zur benutzung gvim

#### vim kurztasten

eine übersicht der rumex kurztasten für den editor vim findest du auf der seite [vim-kurztasten](#).

#### gvim menü

eine übersicht des rumex menüs für den editor gvim findest du auf der seite [gvim-menü](#).

**dateiname ergänzen** will man in die datei einen dateinamen einbauen, weiß aber nicht mehr genau wie er heißt, kann man folgenden trick verwenden. in diesem beispiel wird ein bildname gesucht. im text schreibe man `../bilder/tw` und drückt dann die tastenkombination `c-x + c-f`, gvim öffnet ein dialogfeld in dem alle dateien die auf dieses muster übereinstimmen geöffnet. gibt es nur einen treffer wird dieser gleich eingefügt.

**wort innerhalb des dokumentes suchen** sucht man ein wort das man im dokument schon einmal verwendet hat, um zum beispiel darauf zu verweisen. schreibt man den wortanfang und drückt dann c-p. es öffnet sich ein dialogfeld in dem alle wörter die auf dieses muster passen angezeigt werden. gibt es nur einen treffer wird dieser gleich eingefügt.

**nützliche erweiterungen** vim bietet ein paar nützliche erweiterungen in form von plugins an. hier eine liste, der plugins, die ich gerne verwende.

**pathogen.vim** diese erweiterung macht das installieren weitere erweiterungen einfach. dabei ist die installation von pathogen schnell erledigt. in die .vimrc muss dann noch nachfolgende zeile eingebaut werden.

```
call pathogen#infect()
```

eingebaut werden.

jetzt braucht man die erweiterungen nur mehr in das verzeichnis .vim/bundle zu kopieren und vim neu starten. in verbindung mit git wieder eine einfache sache.

**vim-pandoc** erweiterung rund um pandoc.

**fuzzyfinder** dateien schnell zum editieren öffnen. installiert ist diese erweiterung, vorausgesetzt man verwendet pathogen, mit dem befehl:

```
wget -o /tmp/vim-fuzzyfinder.zip http://www.vim.org/scripts/download
mkdir ~/.vim/bundle/vim-fuzzyfinder
unzip /tmp/vim-fuzzyfinder.zip -d ~/.vim/bundle/vim-fuzzyfinder/
```

für das öffnen dies datei dialogs sollte man sich dann noch eine kurz-taste konfigurieren.

```
" -----
" fuzzyfinder file suche auf <f12> binden
map <f12> :fuzzyfinderfile <cr>
```

## 13.0.2. seite auf einem github.com server einrichten

@todo

- einrichten eines github.com zugangs
- arbeits- repository auf den ap<sup>1</sup> holen

---

<sup>1</sup>mit ap ist der arbeitsplatz rechner gemeint.

- die rumex zip datei
- die dateien des zip archives in das arbeits- repository kopieren
- grund dateien anpassen
  - @fixme: angabe welche dateien fehlt noch@
- erste änderungen vornehme
- make online - fertig.

### 13.0.3. seite auf einem nicht github.com server einrichten

#### datei upload per git

@todo

auf der seite [rumex.it-bayer.de](http://rumex.it-bayer.de) findet man eine beschreibung wie man den rumex baukasten auf einen **nicht** github.com server installiert.

#### datei upload per ftp

@todo

## 13.1. aufbau des rumex baukastens

@todo

### 13.1.1. root verzeichnis

im root verzeichnis findet man alle html dateien der seite. diese werde vom baukasten erstellt und müssen nicht von hand verändert werden. zusätzlich findet man noch ein folgende systemdateien:

**rss.xml** news feed datei, wird vom system erstelle

**readme.md** beschreibungsdatei die von github.com gebraucht wird

**robots.txt** datei für die suchmaschinen

**favicon.ico** icon für den browser

**.htaccess** konfiguration für den apache server

**cname** @fixme: beschreibung@

### 13.1.2. unterverzeichnisse

**.rumex/** @fixme: beschreibung@

**.rx/** @fixme: beschreibung@

**rxtpl/** standard template verzeichnis. in diesem verzeichnis befinden sich die dateien die für das aussehen der seite verantwortlich sind.

folgende dateien und verzeichnisse sind hier zu finden

- index.html (weiterleitung zum root verzeichnis)
- css/
- js/
- img/

**bilder/** in diesem verzeichnis werden alle bilder der seite abgelegt.

### 13.1.3. steuerung durch dateiendung

das aussehen der dateien bezüglich des inhaltsverzeichnisses könnte auch durch die dateiendung gesteuert werden.

#### beispiel

**datei.rx0s** würde eine html datei ohne inhaltsverzeichnis erstellen.

**datei.rx1s** erstellt die html datei mit dem inhaltsverzeichnis aus den einträgen der h1 überschriften.

**datei.rx2s** erstellt eine html datei mit dem inhaltsverzeichnis aus den einträgen der h1 und h2 überschriften.

zusätzlich könnte die dateiendung auch eine unterschiedliche verwendung der dateien ermöglichen.

**datei.rx0s, datei.rx1s ...** standard datei.

**datei.rx0x, datei.rx1x ...** datei die nicht in die liste auf index.html eingebunden wird.

**datei.rx0v, datie.rx1v ...** versteckte datei. diese taucht weder in der index.html noch in der sitemap.xml auf.

**datei.rx0w** datei mit einer weiterleitung. die weiterleitung wird dabei mittels javascript realisiert, da github keine .htaccess weiterleitung unterstützt.

```
% weiterleitung nach beschreibung.html
%
%

<script language="javascript">
<!--
window.location.href="beschreibung.html";
// -->
</script>
```

bei der änderung der dateiendung bleibt der eigentliche html name gleich. nur die funktion der einbindung ändert sich.

### 13.1.4. sonderseiten

im verzeichnis pandoc befinden sich sonderseiten.

**rumex/index.rx0x** diese datei wird vom programm .bin/make\_index.pl erstellt, muss somit nicht vorhanden sein.

**rumex/start.rx0s** die start.rx0s wird als vortext in die index.rx0x eingebunden. mit ihr kann man oberhalb der seiten liste einen extra text in die index.html eingebunden werden.

**diese datei ist erforderlich und muss vorhanden sein.**

**rumex/rss.rx0x** datei für die rss feed funktion.

**rumex/impressum.rx0x** datei für die impressumsangaben.

**ist nicht zwingend erforderlich. jedoch muss die .inc/fuss.html entsprechende bearbeitet, der link muss raus genommen, werden.**

**rumex/makefile** sie make steuerdatei.

### 13.1.5. aufbau der startseite

die startseite markdown/start.rx0s muss vorhanden sein. es reicht auch ein 'touch markdown/start.rx0s.

der normale aufbau könnte so ausschauen. die pandoc kopfzeile sind nicht zwingend erforderlich.

```
% start.rx0s
%
%
```

hier kommt auch schon der vortext für die index.html

### 13.1.6. aufbau der einzelseiten

die einzelseiten liegen alle im verzeichnis rumex und zwar in der sprache markdown bzw. der erweiterung von pandoc.

diese einzelseiten werden in chronologischer reihenfolge in die startseite index.html eingebunden und bilden sozusagen das inhaltsverzeichnis der seite. in jeder einzelseite wird dazu ein sogenannter "*vortext*" hinterlegt. die seite bzw. der kopf der seite hat dabei folgenden aufbau.

```
% seiten überschrift 1
% seiten überschrift 2
% seiten überschrift 3
```

```
<!--
```

```
# listen-überschrift
```

überschrift und text der in der listenübersicht  
auf index.html angezeigt wird.

alles was innerhalb der html kommentar marken  
steht wird nur auf der index.html seite angezeigt.

```
-->
```

alles was sich außerhalb der html marken

befindet wird auch auf der eigentlichen seite angezeigt.

durch das schlüsselwort "schnipp", das auch in html kommentar marken stehen muss, wird der vortext beendet. auf der index.html erscheint an dieser stelle der link "... weiter lesen".

```
<!-- schnipp -->
```

ab hier geht dann der inhalt der eigentlichen seite los.

### 13.1.7. template

@todo

änderung gegenüber des original pandoc templates.

## 13.2. rss feed funktion

@todo

wird nicht aus den einzelnen dateien erstellt sondern muss manuell editiert werden, datei .rx/rss.rx0x.

jede überschrift eines eintrags muss mit einem {.nn1} enden.

danach kommen die angaben zu:

**link:** *verweis zur seite mit weiteren informationen*

**autor:** *autor der den eintrag geschrieben hat*

**kategorie:** *kategorie des eintrags*

**datum:** *datum des eintrags*. das richtige format bekommt man mit dem befehl `date -r`.

die in html kommentar marker eingeschlossen sind.

anschließend folgt die meldung. zur zeit werden folgende pandoc formatierungen unterstützt.

- überschriften ab der stufe 3 ###
- aufzählungen -
- aufzählungen \*
- zitate >

- links [link](http://muster.tdl). diese dürfen nicht am anfang einer zeile stehen.
- bilder ![bild](../bilder/muster.png "muster.png"), diese dürfen nicht am anfang einer zeile stehen.
- code 'code'.

### **13.2.1. beispiel einer rss feed seite**

### **13.2.2. rss dateiname**

der dateiname ist mit `rss.xml` vorbelegt und kann über die variable `rss_file` in der `config.md` geändert werden.

### **13.2.3. rss titel**

der title des rss feed wird durch die variable `rss_titel` angepasst.

```
rss_titel = "neuigkeiten von rumex baukasten"
```

### **13.2.4. rss auslagern**

den rss link kann man auch auslagern so dass dieser auf eine andere seite zeigt. dazu setzt man die variable `rss_extern` mit dem entsprechenden link. die variabel `rss_file` wird dadurch nicht mehr verwendet. auch der rss lauf wird dadurch ausgeschaltet und durch eine meldung ersetzt.

### **13.2.5. rss kurztaaste**

für die einzelnen einträge steht auch eine kurztaaste `.rnn` zur verfügung. in `gvim` unter

```
rumex -> textbausteine -> neuernews eintrag
```

eingefügt wird dann folgende vorgabe. der wert hinter datum wird von system ausgelesen und entsprechende gesetzt.



```
# neue nachricht{.nn1}

<!--
| link: http://www.it-bayer.de/rumex/
| autor: it-bayer
| kategorie: neues
| datum: mon, 28 oct 2013 07:36:56 +0100
-->
```

ab hier geht die neue nachricht los.

### 13.2.6. interna

durch die beschriebenen rss variablen wird die erstellung des rss feed gesteuert. es wird in jede html datei nachfolgender header abschnitt eingebaut wenn die rss\_title variable gesetzt wurde. rss\_file bzw. rss\_extern steuern den href eintrag.

## 13.3. rumex auf einem usb stick

rumex kann auch auf einem usb stick installiert werden. der stick muss aber ein dateiformat besitzt welches mit dateirechten und symbolische links umgehen kann.

**usb sticks im vfat format funktionieren nicht.** man kann zwar die daten darauf ablegen. das arbeiten über den stick funktioniert nicht wirklich. auch wenn man die daten, von einem vfat stick, auf ein \*nix system kopiert werden muss händisch nach gebessert werden.

linktipp: [usb stick unter linux verschlüsseln](#)

## 13.4. make steuerung

@todo

gesteuert wird der baukasten mittels make im unterverzeichnis rumex. folgende make befehle stehen dabei zur verfügung.

**make html** erstellt die einzelnen html dateien. hier kann auch nur make verwendet werden.

**make index** erstellt die `index.md` datei aus der dann die `index.html` datei erstellt wird.

**make all** eine zusammenstellung aus `make index` und `make html`.

**make online** daten auf github hoch laden.

**make bilder** erstellt bilder in verschiedenen auflösungen.

## 13.5. update

@todo

## 13.6. homepage änderung, schnell und immer aktuell

wer wünscht es sich nicht einen schnellen zugriff auf seine homepage. sei es um schnell was zu ändern oder genauso schnell was neues einzustellen. rumex bietet in verbindung mit tilda eine schöne, einfache und auch schnelle lösung.

### 13.6.1. installation der tilda unterstützung

die installation ist nicht umfangreich. man braucht vim und tilda und dann noch zwei bash script. eine kopiervorlage der beiden scripte findet man findet man im `.rumex/bin/` verzeichnis.

```
sudo apt-get install tilda vim
cp ../rumex/.rumex/bin/rxt-rumex.sh ~/bin/.
cp ../rumex/.rumex/bin/rumex-tilda.sh ~/bin/.
```

diese beiden bash scripte müssen anschließend noch angepasst werden.

### shell scripte

die scripte starten zum einem tilda und zum anderem übergeben sie tilda die kommandos die dafür sorgen vim im richtigen rumex ordner zu öffnen. installiert werden können diese scripte im `~/bin` oder `/usr/local/bin/` verzeichnis.

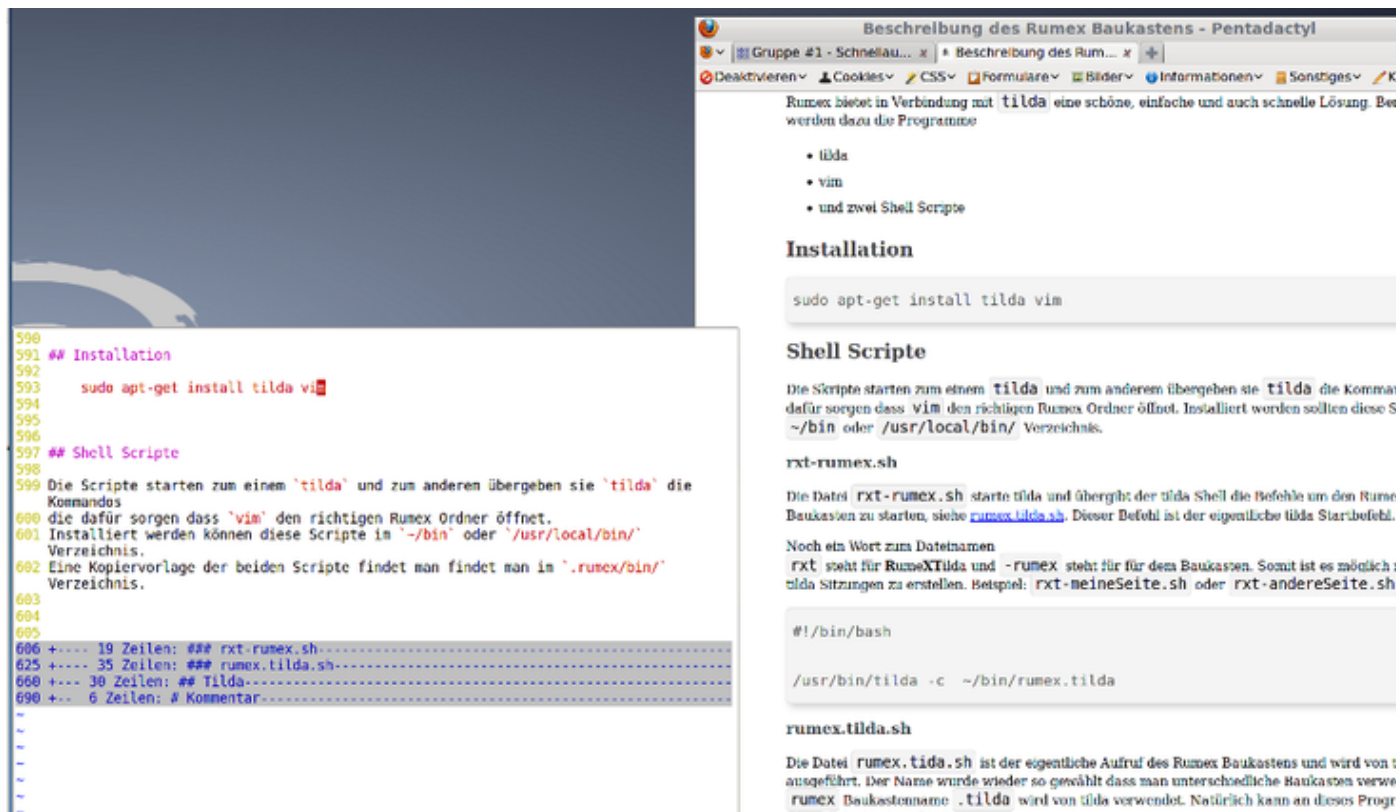


Abbildung 13.1.: rumex im tilda fenster. mit einem tastendruck öffnet sich das tilda fenster und man kann die texte eintippen. ein erneuter tastendruck schließt das tilda fenster wieder und der bildschirm ist wieder frei. hat man seine änderung abgeschossen kann man mit den rumex **vim kurztasten** die änderung schnell online stellen.

**rumex-tilda.sh** die datei rumex-tilda.sh starte tilda und übergibt der tilda shell die befehle um den rumex baukasten zu starten, siehe rumex.vim.sh.

```
#!/bin/bash
```

```
/usr/bin/tilda -c ~/bin/rumex-vim.sh
```

**rumex-vim.sh** mit dem befehl rumex-vim.sh wird der rumex baukastens aufgerufen. dieser befehl wird unter anderem auch von rumex-tilda.sh verwendet. rumex-vim.sh kann natürlich auch in einem shellfenster ausgeführt werden.

**rumex-gvim.sh** mit dem befehl rumex-gvim.sh wird der rumex baukasten mit dem editor gvim gestartet.

### 13.6.2. tilda einrichten

nach dem ersten start wird tilda in linken oberen bildschirm bereich eingeblendet. man sollte tilda nun noch an seine bedürfnissen anpassen. dazu in das tilda fenster mit der rechten maustaste klicken und eigenschaften aus wählen.

**übrigens:** man kann tilda mehrfach starten. somit kann auf mehreren rumex installationen parallel über diese weiße zugegriffen werden. man sollte nur jede tilda sitzung ein wenig anders konfigurieren.

**nachteil:** ein nachteil von tilda darf man aber nicht verschweigen. bei wechseln zwischen den fenstern kann man die tastenkombination <alt>+<tab> nicht verwenden bzw. man kommt mit dieser kombination nicht mehr zurück nach tilda. schließt und öffnet man tilda mit der definierten taste bekommt man aber den fokus wieder in das fenster.

will man die <alt>+<tab> kombination doch verwenden muss man die standardeinstellung von tilda ändern. den erforderlichen schalter findet man in der konfiguration, reiter *allgemein* -> schalter *nicht in der taskleiste anzeigen*.

## 13.7. Warum wurde diese Beschreibung nicht mit Rumex erstellt

Eigentlich sollte Rumex sich auch selber beschreiben. Bei der Erstellung meiner *Sensen Denkschrift* ist mir Rumex jedoch zu klein geworden. Große Dokumente sind mit Rumex nicht mehr so einfach zu handeln. Da es sich bei der Rumex Beschreibung auch um ein großes Dokument handelt. Es wächst zumindest immer weiter. Bin ich auf meine Vorlage für tex4ht umgestiegen. Ich kann somit die Texte mit einem Programm wie Texstudio<sup>2</sup> bearbeiten und fühle mich bei der Bearbeitung größere Dokumente einfach wohler.

Außerdem finde ich die PDF Ausgabe mittels KOMA-Scripts auch schöner als die von Pandoc.

---

<sup>2</sup>Obwohl mir beim Texstudio die Tasten zur Navigation, wie ich sie von vim gewohnt bin, fehlen.

# A. Lizenz

Diese Vorlage veröffentliche ich mit den Bedingungen der **cc-by-nc-sa Lizenz**.

**Namensnennung** Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen.

**Keine kommerzielle Nutzung** Dieses Werk bzw. dieser Inhalt darf nicht für kommerzielle Zwecke verwendet werden.

**Weitergabe unter gleichen Bedingungen** Wenn Sie das lizenzierte Werk bzw. den lizenzierten Inhalt bearbeiten oder in anderer Weise erkennbar als Grundlage für eigenes Schaffen verwenden, dürfen Sie die daraufhin neu entstandenen Werke bzw. Inhalte nur unter Verwendung von Lizenzbedingungen weitergeben, die mit denen dieses Lizenzvertrages identisch oder vergleichbar sind.

---

Außerdem müssen die Lizenzen der Programme, die zum Erstellen des jeweiligen Dokumentes verwendet werden, berücksichtigt werden.

## B. Impressum / Kontakt

**Verwendung:** Privat

**Anschrift:** Stefan Blechschmidt

Auggenbach 9

94357 Konzell

oder per E-Mail an [sb\(at\)it-bayer.de](mailto:sb(at)it-bayer.de).

## **C. Literatur**



# Literaturverzeichnis

- [1] Blechschmidt, Stefan: *Die Sensen Denkschrift*, 2012, <http://www.hochfelder.de/sense/>
- [2] Blechschmidt, Stefan: *Ein HTML Baukasten mit pandoc*, 2012, <http://www.it-bayer.de/rumex/>
- [3] Gurari, Eitan: *TeX4ht: LaTeX and TeX for Hypertext*, 2009, <http://www.tug.org/applications/tex4ht/mn.html/>
- [4] Kohm, Markus : *KOMA-Script Documentation Project*, —, <http://www.komascript.de/>
- [5] van der Zander, Benito : *TeXstudio is an integrated writing environment for creating LaTeX documents*, —, <http://texstudio.sourceforge.net/>
- [6] Mittelbach, Frank: *varioref - Intelligent page referendes*, 1992 - 2011, <http://www.ctan.org/pkg/varioref>
- [7] Schöpf, Rainer: *verbatim - Reimplementation of and extensions to L<sup>A</sup>T<sub>E</sub>X verbatim*, 1989 - 2003, <http://www.ctan.org/pkg/verbatim>
- [8] Van Zandt, Timothy; Voß, Herbert; Girou, Denis; Rahtz, Sebastian; Mansfield, Niall : *fancyvrb - Sophisticated verbatim text*, Version 2.8, <http://www.ctan.org/pkg/fancyvrb>
- [9] Carlisle, David: *graphicx - Enhanced support for graphics*, 1995-1997,1999 David Carlisle, <http://ctan.org/pkg/graphicx>
- [10] Melchior, Franz: *soul - Hyphenation for letterspacing, underlining, and more*, Version 2.4, <http://ctan.org/pkg/soul>
- [11] Theiling, Henrik: *eurosym: Das Eurosymbol-Paket für LaTeX*, Version 1.4, 2004, <http://www.theiling.de/eurosym.html>
- [12] Oberdiek, Heiko: *hyperref - Extensive support for hypertext in LaTeX*, Version 6.83m, 2001 - 2012, <http://www.ctan.org/pkg/hyperref>
- [13] Lickert, Knut: *blindtext - Producing 'blind' text for testing*, Version 2.0, <http://www.ctan.org/pkg/blindtext>

- [14] Rennish, George: *TeX4ht: Options*, Blue Danube, Webseite 2012,  
<http://www.cvr.cc/?p=504>
- [15] Gurari, Eitan M.: *TEX4ht: HTML Production*, TUGboat, Volume 25  
(2004), No. 1 — Proceedings of the Practical TEX 2004 Conference,  
<https://www.tug.org/TUGboat/tb25-1/gurari.pdf>
- [16] Arseneau, Donald: *The ulem package: underlining for emphasis*,  
2011,  
<http://texdoc.net/texmf-dist/doc/generic/ulem/ulem.pdf>