Rumex Beschreibung

Stefan Blechschmidt

Juni 2014

An diesem Dokument wird zur Zeit gearbeitet.

0.1 Vorwort

Die Beweggründe mit einem eigenen System eine Internet Seite zu Verwalten sind mit der Zeit in mir gewachsen.

Es gibt immer mehr Systeme die zum Erstellen und Verwalten von Internetseiten eingesetzt werden. Probiert habe ich schon viele und mit den meisten war ich auch sehr zufrieden, wenn da nicht immer der Hacken mit dem Updates wäre. Irgendwie, zumindest kommt es mir so vor, ist der Aufwand das System der Seite auf dem aktuellen Stand zu halten größer geworden als den Inhalt der eigentlichen Seite zu pflegen.

Ich wünschte mir wieder eine Internetseite wie in früheren Zeiten. Eine Seite die aus einfachen HTML Seiten besteht.

Irgendwann bin ich dann auf `markdown` und `pandoc` gestoßen und die Idee dieses Home Page Baukastens ist entstanden.

0.1.1 Rumex?

Rumex ist die lateinische Bezeichnung für den Ampfer und dieser taucht in der Natur dann auf, wenn der Boden Überdüngt, Verdichtung 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.

0.1.2 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[1] 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.

[1] Obwohl mir beim Texstudio deii Tasten zur Navigation, wie ich sie von vim gewohnt bin, fehlen.

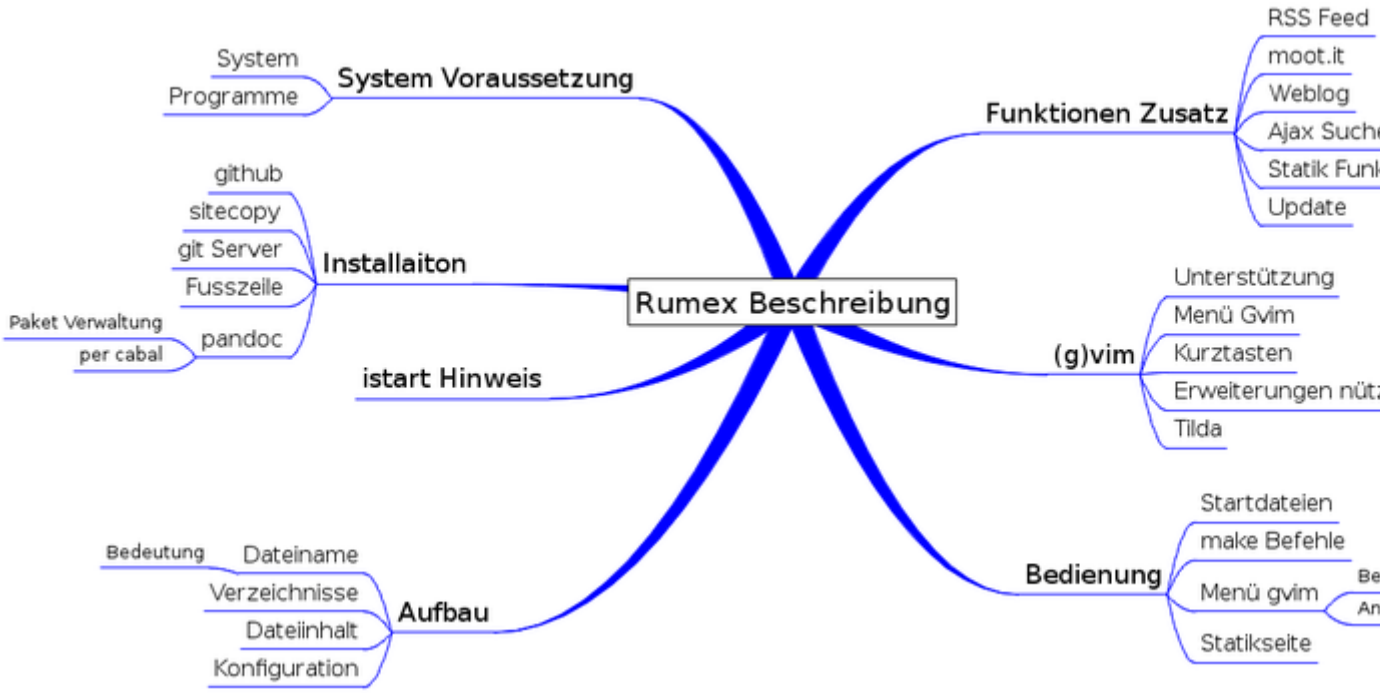

Abbildung 0.1: Kleine Gedankenkarte der Rumex Beschreibung. Zur Zeit in Arbeit.

0.2 Installation und Einrichten des Rumex Baukastens

0.2.1 Vorbereitung des Rechners

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

- bash
- make
- perl
- git
- pandoc
- imagemagick
- wget
- ~~wput~~
- sitecopy

- vim (g)vim
- texlive

Wobei `bash`, `make` und `perl` eigentlich bei jeder `*nix` Installation bereits vorhanden sein dürft. Die restlichen Programme müssen nach installiert werden.

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

`texlive`[2] wird nur gebraucht wenn man auch PDF Dateien erstellen möchte.

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

```
sudo apt-get install make perl

sudo apt-get install git-core wget sitecopy pandoc imagemagick

sudo apt-get install gvim

sudo apt-get install texlive
```

0.2.2 Installation von Rumex auf dem lokalen Arbeitsplatz

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

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

oder

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

Jetzt wechsle 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
```

Fertig :-)

[2] Die Funktion der PDF Erstellung ist zur Zeit noch nicht enthalten.

0.2.3 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 Startseite von Rumex oder hier.

0.2.4 moot.it

@todo: Hier fehlt noch die Beschreibung

- ein/auschalten
- Kontoname einstellen

0.2.5 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_script.php?
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 Kurztaste konfigurieren.

```
" ---------------------------------------------------------
" FuzzyFinder File Suche auf <F12> binden
map <F12> :FuzzyFinderFile <CR>
```

0.2.6 Seite auf einem github.com Server einrichten

@todo

- Einrichten eines github.com Zugangs
- Arbeits- Repository auf den AP[3] holen
- 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.

0.2.7 Seite auf einem nicht github.com Server einrichten

Datei upload per git

@todo

Auf der Seite 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

0.3 Aufbau des rumex Baukastens

@todo

[3] Mit AP ist der **A**rbeits**P**latz Rechner gemeint.

0.3.1 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@

0.3.2 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 Dateinen 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.

0.3.3 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 aud 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.

0.3.4 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.

0.3.5 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
```

0.3.6 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 Inhlat der eigentlichen Seite los.
```

0.3.7 Template

@todo

Änderung gegenüber des Original `pandoc` Templates.

0.4 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: *Kategoie 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‘.

0.4.1 Beispiel einer RSS Feed Seite

0.4.2 RSS Dateiname

Der Dateiname ist mit `rss.xml` vorbelegt und kann über die Variable `RSS_FILE` in der `config.md` geändert werden.

0.4.3 RSS Titel

Der Title des RSS Feed wird durch die Variable `RSS_TITEL` angepasst.

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

0.4.4 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.

0.4.5 RSS Kurztaste

Für die einzelnen Einträge steht auch eine Kurztaste `.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.
```

0.4.6 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.

0.5 Rumex auf einem USB Stick

Rumex kann auch auf einem USB Stich 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

0.6 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.

0.7 Update

@todo

0.8 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.

0.8.1 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.

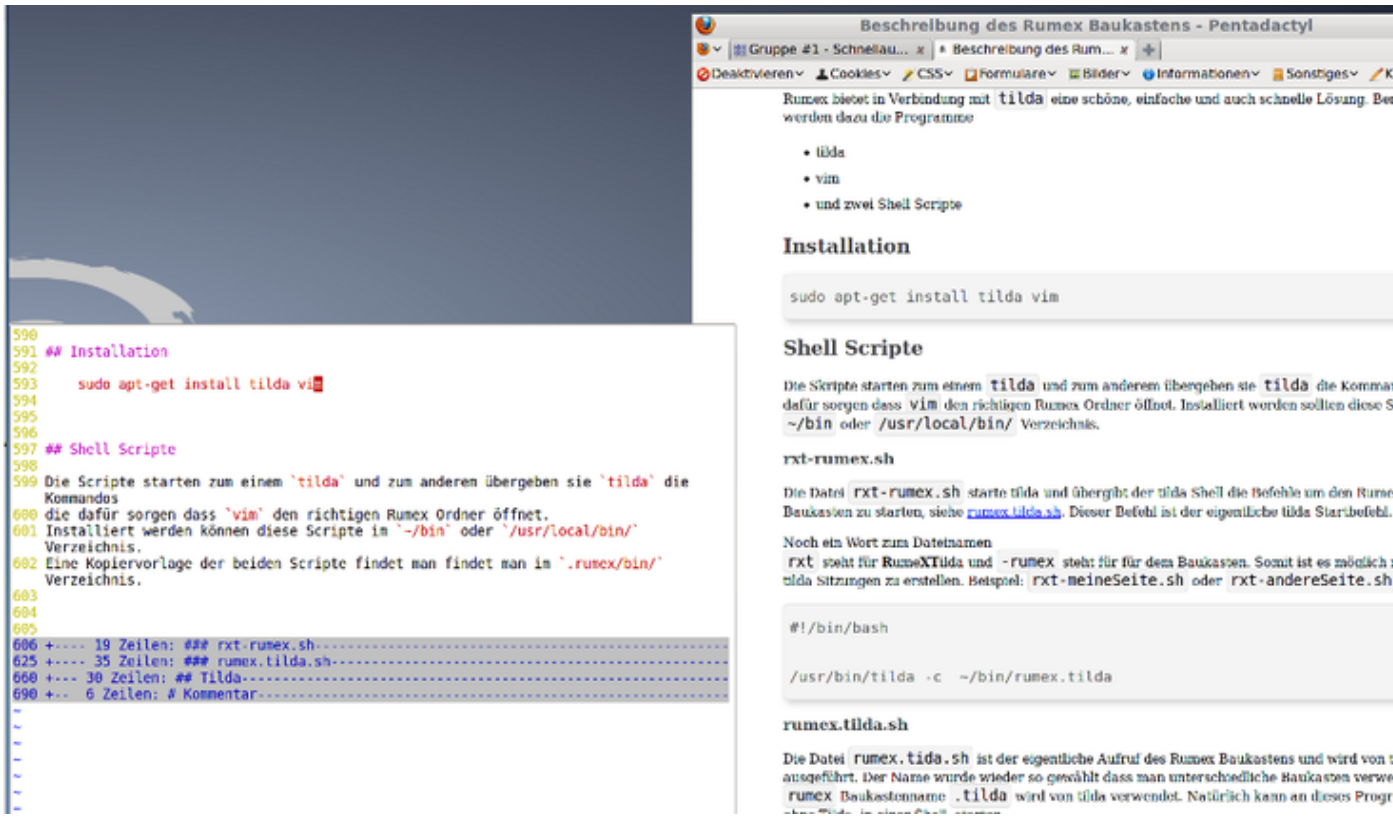

Abbildung 0.2: Rumex im tilda Fenster. Mit einem Tastendruck öffnet sch das tilda Fenster und man kann die Texte eintippen. Ein erneuter Tastendruck schließt das tilda Fenster wieder und der Bildschirm ist wieder frei. Hat man seine Änderung abgeschossen kann man mit den Rumex vim Kurztasten die Änderung schnell online stellen.

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 öffnet. Installiert werden können diese Scripte im `~/bin` oder `/usr/local/bin/` Verzeichnis.

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.

0.8.2 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.*

1 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[1] 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[2] 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.~~[3]
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 anschauen.

1.1 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.

[1] Im Neudeutschen würde man die Denkschrift als Memorandum bezeichnen.
[2] Die Funktionstasten sind im gvim Rumex Menü nicht eingebaut. Man sollte sie sich also merken :-).
[3] 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.

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.

1.2 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
```

1.3 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.

1.4 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[4] 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.

```
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.

[4] Die Literatur Erweiterung ist ab der Rumex Version 0.8.2 enthalten.

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=124228120",
    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=124228120",
    note = "[Online; Stand 18. Dezember 2013]"
}
```

1.5 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.

1.5.1 Setzen der Überschrift für das Literaturverzeichnis

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

Beispiel

```
# Literaturverzeichnis
```

1.5.2 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.

1.5.3 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.

1.5.4 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`^{Chromium} bzw. `Tab Auto Reload`^{FireFox} gute Erfahrungen gemacht.

Die PDF Datei dieser Beschreibung kann man sich hier ansehen. Die Markdown Quelldatei kann man sich hier holen.

1.6 Literaturverzeichnis

2 vim-Kurztasten

Alle Rumex Kurztasten, für den Editor VIM, beginnen mit einem `,r`[1]. Für die Bedienung von Rumex braucht mein 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.

2.1 Textbausteine

Mittels dieser Kurztasten können Textbausteine eingebunden werden.

2.1.1 `,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)
 ========================================================

Vortext INDEX
```

[1] Ausnahmen bestätigen die Regel. So wurde für die zwei spaltige Darstellung `,spn2` verwendet.

```
-->

Vortext INDEX und SEITE

<!-- schnipp -->

TEXT DER SEITE

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

2.1.2 ,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.
```

2.1.3 ,rwl (RumexWeiterLeitung)

Erstellt einen 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`.

2.1.4 ,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>
```

2.1.5 ,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>
```

2.2 Git Kommandos

2.2.1 ,rgp (RumexGitPull)

Git pull

2.2.2 ,rgs (RumexGitStatus)

Git status

2.3 Text Formatierung

2.3.1 ,rff (RumexFormatFett)

Markierte Textstelle fett darstellen

2.3.2 ,rfk (RumexFormatKursiv)

Markierte Textstelle kursiv darstellen

2.3.3 ,rfl (RumexFormatListe)

Markierte Zeilen in eine Liste wandeln

2.3.4 ,rfn (RumexFormatNummernliste)

Markierte Zeile in eine Nummernliste wandeln

2.3.5 ,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

2.4 Make Befehle

2.4.1 ,rma (RumexMakeAll)

Speichert alle offenen Dokumente und erstellt daraus die HTML Datei.

2.4.2 ,rmb (RumexMakeBild)

Erstellt die unterschiedlichen Bildgrößen

2.4.3 ,rmca (RumexMakeCleanAll)

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

2.4.4 ,rmcb (RumexMakeCleanBilder)

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

2.4.5 ,rmch (RumexMakeCleanHtml)

Löscht alle `.html` Dateien die von Rumex erstellt wurden.

2.4.6 ,rmcx (RumexMakeCleanXml)

Löscht alle `.xlm` Dateien die von Rumex erstellt wurden.

2.4.7 ,rmcf5 (RumexMakeCleanF8htm)

Löscht alle `.htm` Dateien die mittels der Gvim Taste F5, siehe HTML und PDF Dateine mit pandoc und gvim erstellen, erstellt wurden.

2.4.8 ,rmh (RumexMakeHtml)

Erstellt die `.html` Dateien.

2.4.9 ,rmi (RumexMakeIndex)

Erstellt die index Datei.

2.4.10 ,rmm (RumexMakesiteMap)

Erstellt die sitemap Datei.

2.4.11 ,rmo (RumexMakeOnline)

Speichert alle Dateien und stellt diese online.

2.4.12 ,rmr (RumexMakeRss)

Erstellt die rss Datei.

2.4.13 ,rms (RumexMakeSuche)

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

2.5 Vorschau

2.5.1 ,rsf (RumexShowFile)

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

2.5.2 ,rsw (RumexShowWww)

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

2.5.3 ,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.

2.6 Sonstiges

2.6.1 ,rku

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

2.6.2 ,ros (RumexOpenStart)

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

2.6.3 ,ror (RumexOpenRss)

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