

# notmuch

Michael „sECuRE“ Stapelberg  
LUGS, 2013-04-19

powered by Lua<sup>A</sup>T<sub>E</sub>X

# Was ist notmuch?

- Ein Tool, um Emails zu indexieren/durchsuchen
- Code in einer Library → z.B. python-notmuch, mutt-notmuch
- **kein** POP/IMAP/SMTP support
- Standard-Frontend: notmuch-emacs, es gibt noch andere

## Warum interessant?

- notmuch ist schnell (benutzt Xapian)
- notmuch denkt in Threads, wie gmail
- keine Ordner mehr, stattdessen Tags oder einfach suchen
- notmuch-emacs ist ein gutes Text-Mailprogramm

# Setup

- `notmuch(1)` richtet eine config ein
- z.B. `offlineimap` zum Herunterladen neuer Mails
- `notmuch new` für das initiale Hinzufügen und neue Mails
- Automatisches Tagging:  
<http://github.com/teythoon/afew>
- address completion when composing:  
<http://notmuchmail.org/emacsstips/#index15h2>
- notmuch-emacs config: <http://code.stapelberg.de/git/configfiles/tree/emacs-zkj-notmuch.el>

demo

# Goodie 0: Nur eigene Threads auf einer ML

```
#!/bin/zsh
```

```
# "notmuch new" will apply the "new" tag on all messages it finds.  
# This script will then perform some tagging on new messages.
```

```
# Don't show messages to go-nuts in my inbox...
```

```
notmuch tag -inbox to:golang-nuts@googlegroups.com AND tag:new
```

```
# ...unless I am participating
```

```
notmuch tag +inbox to:golang-nuts@googlegroups.com AND tag:new \  
    AND from:stapelberg@debian.org
```

```
# Done processing, remove the new tag.
```

```
notmuch tag -new tag:new
```

# Goodie 1: Automatische Anrede

```
(defun get-recipient-for-body ()
  (let* ((to (message-fetch-field "To"))
        (comps (mail-extract-address-components to)))
    (if to
        (shorten-realname (nth 0 comps))
        "")))

(defun my-fill-body ()
  (let ((recipient (get-recipient-for-body)))
    (save-excursion
      (goto-char (point-max))
      (insert (format "Hi %s,\n\n" recipient)))))

(add-hook 'message-signature-setup-hook 'my-fill-body)
```

## Goodie 2: Automatische Sender-Wahl

*;;; returns nil if PATTERN is not contained in any member of HEADERS*

```
(defun any-value-matches (pattern headers)
  (if (null headers)
      nil
      (or
       (string-match pattern (car headers))
       (any-value-matches pattern (cdr headers))))))

(defun determine-sender-and-reply (r-all)
  (let* ((headers (notmuch-show-get-prop :headers nil))
        (vals (plist-values headers))
        (sender (cond
                 ((any-value-matches "debian\\.org" vals)
                  "Michael Stapelberg <stapelberg@debian.org>")
                 ((any-value-matches "i3" vals)
                  "Michael Stapelberg <michael@i3wm.org>")
                 (t nil))))
    (notmuch-mua-reply (notmuch-show-get-message-id) sender r-all)))

(define-key notmuch-show-mode-map "r"
  (lambda ()
    (interactive)
    (determine-sender-and-reply nil)))
```



## Goodie 3: Patches mergen

```
(define-key notmuch-show-mode-map "3"
  (lambda ()
    "Merge this patch into ~/i3"
    (interactive)
    (let ((id      (notmuch-show-get-message-id))
          (tempfile (make-temp-file "i3-patch"))
          (buf      (get-buffer-create "*notmuch-last-i3-merge*")))
      (with-temp-file tempfile
        (call-process notmuch-command nil t nil "show" "--format=raw"
                      id)
        (switch-to-buffer buf)
        (cd "/home/michael/i3")
        (call-process "git" nil buf nil "am" "-3" "--whitespace=fix"
                      tempfile))))))
```