Post mit Perl

Stefan Hornburg (Racke)

GPW 2018, Gummersbach, 5. April

Post mit Perl

Übersicht

- Aufbau einer Email
- Emailversand
- IMAP
- Email parsen
- Use Cases

Aufbau einer Email

- Header
- Body

Aufbau einer Email

- Attachments
- HTML / Plaintext

Aufbau einer Email

```
Email::MIME->new( $message )->debug_structure;
```

Aufbau einer Email

```
+ multipart/mixed;
    + application/octet-stream; name=83AKE9.pdf
    + multipart/alternative;
        + text/plain; charset=UTF-8
        + text/html; charset=UTF-8
```

Post verschicken

Too many ways to do it ...

Module (Legacy)

- Net::SMTP
- MIME::Lite

Module

Versand:

- Email::Send (deprecated)
- Email::Sender
- Email::Sender::Simple

Sonstiges:

- Email::MIME
- MIME::Entity
- Mail::Box

Beispiel

```perl
my $email = Email::Simple->create(
    header => [
        From    => '"Module Updates" <info@cpan.pm>',
        To      => '"Racke" <racke@racke.pm>',
        Subject => 'Changes in Module Foo::Bar',
    ],
    body => "Baz.\n",
);

sendmail($email);
```

Beispiel mit Attachment

```
my $email = MIME::Entity->build(
    From    => '"Module Updates" <info@cpan.pm>',
    To      => '"Racke" <racke@racke.pm>',
    Subject => 'Changes in Module Foo::Bar',
    Data => ["Baz.\n"],
);

$email->attach(
    Path => '../perlmail-de-handout.pdf',
    Type => 'application/pdf',
);

sendmail($email);
```

Beispiel mit Umlaut

```
my $email = Email::Simple->create(
    header => [
        From    => '"Module Updates" <info@cpan.pm>',
        To      => '"Racke" <racke@racke.pm>',
        Subject =>  'Änderungen im Modul Foo::Bar',
    ],
    body => "Baz.\n",
);

sendmail($email);
```

Mail Clients

- ☺ Thunderbird
- ☹ K9 (Android)

Korrektur: Beispiel mit Umlaut

```
my $email = Email::Simple->create(
    header => [
        From    => '"Module Updates" <info@cpan.pm>',
        To      => '"Racke" <racke@racke.pm>',
        Subject => encode('MIME-Header',
                          'Änderungen im Modul Foo',
                         ),
    ],
    body => "Bar.\n",
);

sendmail($email);
```

Kodierter Betreff

```
To: "Racke" <racke@racke.pm>
From: "Module Updates" <info@cpan.pm>
Subject: =?UTF-8?B?w4RuZGVydW5nZW4gaW4gTW9kdWxlIEZvbw==?=
```

Body der Email kodieren

```
my $email = Email::Simple->create(
    header => [
        From    => '"Module Updates" <info@cpan.pm>',
        To      => '"Racke" <racke@racke.pm>',
        Subject => encode('MIME-Header',
                          'Änderungen im Modul Foo::Bar',
                         ),
    ],
    body => encode('UTF-8', "Überflüssig"),
);

sendmail($email);
```

Transports

Transports

- Sendmail
- SMTP
- Maildir
- Redirect

Sendmail-Transport

```
sendmail( $email );
```

Maildir-Transport

```
my $maildir = path(File::HomeDir->my_home)
    ->child('Maildir');

my $transport = Email::Sender::Transport::Maildir->new(
   dir => $maildir
);

sendmail( $email , { transport => $transport, } );
```

Redirect-Transport

```
$transport_orig = Email::Sender::Transport::Sendmail->new;

$transport = Email::Sender::Transport::Redirect->new({
  transport => $transport_orig,
  redirect_address => 'racke@linuxia.de',
});

sendmail( $email , { transport => $transport, } );
```

Redirect-Transport

```
To: racke@linuxia.de
Cc: racke@linuxia.de
From: "Module Updates" <info@cpan.pm>
Subject: Hello world!
Date: Tue, 3 Apr 2018 08:45:59 +0200
X-Intercepted-To: "Racke" <racke@racke.pm>
X-Intercepted-Cc: "Info" <info@racke.pm>
X-Email-Sender-From: info@cpan.pm
X-Email-Sender-To: racke@linuxia.de
X-Email-Sender-To: racke@linuxia.de
Lines: 1

Here we go.
```

Bilder einbinden

- Bilder als Links
- Base64 kodiert
- CID Inline

Emails aus Templates

```perl
my $html = template $template, $tokens,
   { layout => 'email' };

my $f    = HTML::FormatText::WithLinks->new;
my $text = $f->parse($html);
```

Emails aus Templates

```
email {
    %args,
    body   => encode( 'UTF-8', $text ),
    type   => 'text',
    attach => {
       Charset  => 'utf-8',
       Data     => encode( 'UTF-8', $html ),
       Encoding => "quoted-printable",
       Type     => "text/html"
    },
    multipart => 'alternative',
};
```

IMAP

- Module
- Anmeldung
- Ordner auswählen
- Emails suchen
- Email herunterladen

IMAP Module

- Net::IMAP::Client
- Net::IMAP::Simple

Unterschiede IMAP Module

- Konstruktor
  - server
  - ssl / use_ssl
- unterstützte Methoden
- Rückgabewerte
  - select
  - search

Anmeldung

```
my $imap = Net::IMAP::Client->new(
    server => 'mail.linuxia.pm',
    ssl => 1,
    port => 993,
    user => 'racke@racke.pm',
    pass => 'nevairbe',
);

$imap->login;
```

Ordner auswählen

Ordner auswählen:

```
$imap->select('INBOX.Consulting.Bahn');
```

Emails suchen

Emails in ausgewählten Ordner suchen:

```
$ids = $imap->search( { subject => 'Perl' }, 'DATE' );
```

Verarbeitung der Emails

Komplette Email herunterladen:

```
$message = $imap->get_rfc822_body($id);
```

Module zur Verarbeitung

- Email::MIME
- Email::Simple

Email::MIME

- Email aus IMAP einlesen:

```
my $message = $imap->get_rfc822_body($id);
```

- Email aus Datei einlesen:

```
use IO::All;
my $message = io->file('email.eml')->all;
```

- Email parsen:

```
my $parser = Email::MIME->new($message);
```

Email::MIME

MIME Parts durchlaufen:

```
my @out;

$parser->walk_parts(sub {
  my ($part) = @_;
  return if $part->subparts;

  if ($part->content_type =~ m{text/html}i) {
    push @out, $part->body_str;
  }
});
```

Bahnfahrkarte(n) extrahieren

```
$ids = $imap->search('FROM buchungsbestaetigung@bahn.de');

for my $imap_id (@$ids) {
    my $message = $imap->get_rfc822_body($imap_id);
    my $parser = Email::MIME->new($message);
    my @out;

    ...
}
```

Bahnfahrkarte(n) extrahieren

```
$parser->walk_parts(
  sub {
    my ($part) = @_;
    return if $part->subparts;

    if ($part->content_type =~ m{application/octet-stream}i){
      push @out, [ $part->filename, $part->body ];
    }
  }
);

$out[0][1] > io($out[0][0]);
```

Use Cases

- Dancer2::Plugin::Email (Email::Sender)
- Helpdesk::Integration
- Sympa
- Spamassassin

Helpdesk::Integration

- Request Tracker
  - Web-Interface
  - Email-Interface

Helpdesk::Integration

- IMAP-Postfach durchsuchen
- Email(s) parsen
- Ticket über REST API anlegen/aktualisieren

Fragen

The end