Tanz! - Dancer von 0 auf 100

Stefan Hornburg (Racke)
racke@linuxia.de

Österreichischer Perlworkshop 2012, Wien, 17. November 2012

Tanzflur

- einfach
- Anwendung auf Knopfdruck

Tanzflur

- effektiv
- Routes und Schlüsselwörter

Tanzflur

- erweiterbar
- Plugins

Tanzflur

- flexibel
- Engines

Quickstart

- cpanm Dancer
- cpanm YAML
- dancer -a Locker
- cd Locker
- ./bin/app.pl
- x-www-browser `http://localhost:3000/`

Programm

```
#!/usr/bin/env perl
use Dancer;
use Locker;
dance;
```

Modul

```
package Locker;
use Dancer ':syntax';

our $VERSION = '0.1';

get '/' => sub {
    template 'index';
};

true;
```

Template

```
views/index.tt
views/layouts/main.tt
```

Filebrowser

- Liste der Dateien
- Template
- Route

Liste der Dateien

```
use Dancer::Plugin::Autoindex;

$files = autoindex('/');
```

Template

```
<h1><% directory %></h1>
<table>
<tr><th>Name</th><th>Type</th><th>Modified</th></tr>
<% FOREACH file IN files %>
<tr><td><% file.name %></td>
    <td><% file.type %></td>
    <td><% file.modified %></td></tr>
<% END %>
</table>
```

Route

```
get '/home' => sub {
    my $files = autoindex('/');

    template 'filebrowser', {directory => 'Home',
                             files => $files,
                            };
};
```

Konfiguration

Standard config.yml:

```
# template engine
# simple: default and very basic template engine
template: "simple"
```

Locker config.yml:

```
template: "template_toolkit"
```

Routes

- String
- String mit benannten Parametern
- String mit Wildcards
  - Splat
  - Megasplat
- Regulärer Ausdruck
- Regulärer Ausdruck mit Captures

String

```
get '/home' => sub {
    my $files = autoindex('/');

    template 'filebrowser', {directory => 'Home',
                             files => $files,
                            };
};
```

String mit benannten Parametern

```
get '/home/:file' => sub {
    my $files = autoindex(param('file'));

    template 'filebrowser', {directory => param('file'),
                             files => $files,
                            };
};
```

Splat

```
get '/images/covers/*.jpg' => sub {
    my ($isbn) = splat;

    if (-f "public/images/covers/$isbn.jpg") {
        return send_file "images/covers/$isbn.jpg";
    }

    status 'not_found';
    forward 404;
}
```

Megasplat

```
https://vsc.state.gov/lostpwd/biz@linuxia.de/e642bd543b9907bd2c06aa4

get '/lostpwd/**' => sub {
    my ($email, $hash) = splat;

    form->fill(email => $email,
               hash => $hash);

    template('lostpwd_confirm', form => $form);
}
```

Regulärer Ausdruck

Catch-All (letzte Route!)

```
any qr{.*} => sub {
    ...
};
```

Regulärer Ausdruck mit Captures

```
https://dropbox.nite.si/~racke/talks/dancer-beamer.pdf

any qr{^/~(?<user>[^/]+)/(?<file>.*?)/?$} => sub {
    my ($capts, $user, $file);

    $capts = captures;
    $file = $capts->{file};
    $user = $capts->{user};

    ...
};
```

File Download

```
get '/home/**' => sub {
    my ($parts) = splat;
    my $path_relative = join('/', @$parts);
    my $path = autoindex_file_path($path_relative);

    if (-f $path) {
        return send_file $path, system_path => 1;
    }
    elsif (-d $path) {
        return template 'filebrowser', {directory => pop(@$parts),
                                        files => autoindex($path_relative)};
    }
};
```

Upload HTML-Formular

```
<div class="upload">
<h2>Upload file </h2>
<form name="upload" id="upload_form" action="" method="post"
enctype="multipart/form-data">
<input type="file" name="upload_file"><br>
<input type="submit" value="OK">
</form>
</div>
```

Upload Route

```
post '/home/**' => sub {
    my ($parts) = splat;
    my $path_relative = join('/', @$parts);
    my $path = autoindex_file_path($path_relative);

    if ($upload_file = upload('upload_file')) {
        # upload file to directory
        my $target_file = join('/', $path, $upload_file->{filename});

        unless ($upload_file->copy_to($target_file)) {
            die 'Upload failed';
        }
    }

    redirect "/home/$path_relative";
```

before Hook

Passwort geschützte Website:

```
hook 'before' => sub {
    unless (session('user')
        || request->path eq '/login'
        || request->path =~ m%^/lostpwd%
        ) {
        redirect '/login';
    }
};
```

Dateien in public sind nicht geschützt!

before_template_render Hook

```
hook before_template_render => sub {
    my $tokens = shift;
    my $menu;

    if (session('user')) {
        $menu = [{name => 'Logout', url => '/logout'}];
    }
    else {
        $menu = [{name => 'Login', url => '/login'}];
    }

    $tokens->{menu} = $menu;
};
```

Fehlerbehandlung: 500

- Datei: `public/500.html`
- Template: `error_template`
- Stacktrace: `show_errors`

Fehlerbehandlung: 404

- `public/404.html`

-
  ```
  any qr{.*} => sub {
      status 'not_found';
      template 'my_404', { path => request->path };
  };
  ```
- XSS vermeiden

Plugins

- Features
- Schlüsselwörter
- Routes
- Konfiguration

Plugins auf CPAN

- Anzahl: mehr als 100
- Übersicht
  `https://metacpan.org/module/Dancer::Plugins`
- Suche
  `https://metacpan.org/search?q=dancer+plugin`

Plugins auf CPAN

- Dancer::Plugin::Database
  `database`
- Dancer::Plugin::Email
  `email`

Dancer::Plugin::Database Beispiel

```
my $user = database->quick_select('users',
    {username => params->{'username'}});

if ($user) {
    session user => params->{'username'};
}
```

Dancer::Plugin::Database Konfiguration

```
plugins:
  Database:
    driver: Pg
    database: dropbox
    username: vienna
    password: nevairbe
```

Weitere SQL Plugins

- Dancer::Plugin::DBIC
- Dancer::Plugin::SimpleCRUD
- ...

Dancer::Plugin::Email Beispiel

```
$message = template('lostpwd_email',
        {password => $password, url => $url},
        {layout => undef});

email ({type => 'html',
         from => 'service@linuxia.de',
         to => param('email'),
         subject => 'Forgot password',
         message => $message});
```

Dancer::Plugin::Email Redirect

- Modul:
  `Email::Sender::Transport::Redirect`
- Konfiguration:

```
plugins:
  Email:
    transport:
      Sendmail:
        redirect_address: biz@icdev.de
```

Sessions

- Engine

  ```
  session: Storable
  ```

- Verzeichnis

  ```
  session_dir: /var/run/dancer-sessions
  ```

- Ablauffrist

  ```
  session_expires: 8 hours
  ```

Logs

- Engine
  ```
  logger: file
  ```
- Verzeichnis
  ```
  log_path: log
  ```
- Format
  ```
  logger_format: "%t [%P] %L @%D> %m in %f l. %l"
  ```

Szenarien

- Standalone
  `./bin/app.pl`
- CGI, FastCGI
- plackup
  - Starman
  - Twiggy
  - Corona
- Reverse Proxy

Szenarien: Reverse Proxy

- Server
  - Apache
  - nginx
  - Perlbal
- Dancer Konfiguration

```
behind_proxy: 1
```

plackup und Starman

- Kommandozeile

```
plackup -E production
        -s Starman \
        --workers=5 \
        --pid /home/racke/Dropbox/run/Dropbox.pid \
        -p 5000 \
        -a bin/app.pl \
        -D
```

- Initscript
  - `Daemon::Control`

Perlbal

```
LOAD vpaths
XS enable headers
```

Perlbal: Pools

```
CREATE POOL prod_starman_dosqua
 POOL prod_starman_dosqua ADD 127.0.0.1:5000

CREATE POOL prod_starman_vsc
 POOL prod_starman_vsc ADD 127.0.0.1:5001
```

Perlbal: Reverse Proxy

```
CREATE SERVICE vsc_prod
 SET role              = reverse_proxy
 SET pool              = prod_starman_vsc
 SET buffer_uploads    = on
ENABLE vsc_prod
```

Perlbal: Selector

```
CREATE SERVICE vsc_selector
 SET listen                  = 86.59.13.238:80
 SET role                    = selector
 SET plugins                 = vpaths
 VPATH .*                    = vsc_prod
ENABLE vsc_selector
```

Perlbal: SSL Selector

```
CREATE SERVICE vsc_ssl_selector
 SET listen              = 86.59.13.238:443
 SET role                = selector
 SET plugins             = vpaths
 SET enable_ssl          = on
 SET ssl_key_file        = /etc/ssl/private/vsc.state.gov.key
 SET ssl_cert_file       = /etc/ssl/certs/vsc.state.gov.pem
 VPATH .*                = vsc_prod
ENABLE vsc_ssl_selector
HEADER vsc_ssl_selector INSERT X-Forwarded-Proto: HTTPS
```

Skripts

```
use Dancer ':script';

set logger => 'console';
set logger_format => '%m';
```

Dancer2

- keine globalen Variablen
- 100% OO Backend (Moo)
- Scoping for Sub-Applikationen
- überarbeitete Architektur
- YAML => Config::Any

Community

Web: `http://perldancer.org/`
Github: `git://github.com/sukria/Dancer.git`
IRC: #dancer @ irc.perl.org
Mitarbeit: Dancer::Development

The End

Slides: `http://www.linuxia.de/talks/apw2012/`