



# Jifty

## Wolfgang Kinkeldei

Die drei grundlegenden Tugenden  
eines Programmierers sind  
Faulheit, Ungeduld und Hybris.

*Larry Wall*

The three principal virtues  
of a programmer are  
Laziness, Impatience, and Hubris.

*Larry Wall*

# Jifty

- Web Application Framework
- Hauptentwicklung: Best Practical
- Akronym „JFDI“ = Just Do It
- Einfache Dinge sind einfach
- unmögliches ist auch einfach

# Ein paar Features vorab

- benutzt Templating engine (HTML::MASON)
- enthält Application Server
- unterstützt AJAX Anwendungen
- weitgehend unabhängig von JavaScript
- eigener Datenbank-Layer
- DRY - „Don't repeat yourself“
- Intelligentes Formular-Management
- wird mit Pony geliefert

# Installation aus CPAN

```
$ cpan
```

```
cpan shell -- CPAN exploration and modules installation (v1.87)
ReadLine support enabled
```

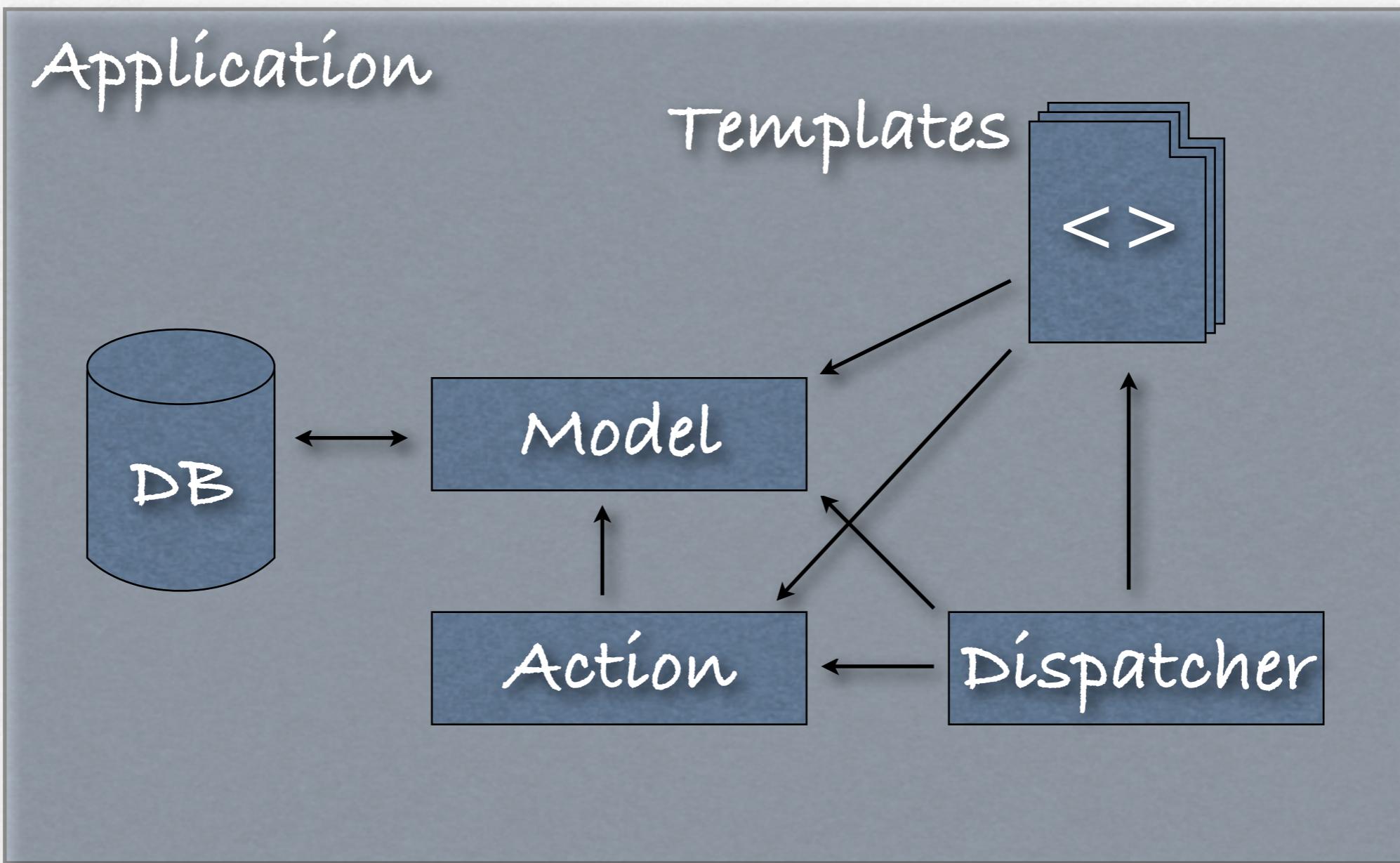
```
cpan> install Jifty
```

```
...einige Zeit später...
```

```
cpan> quit
```

```
$
```

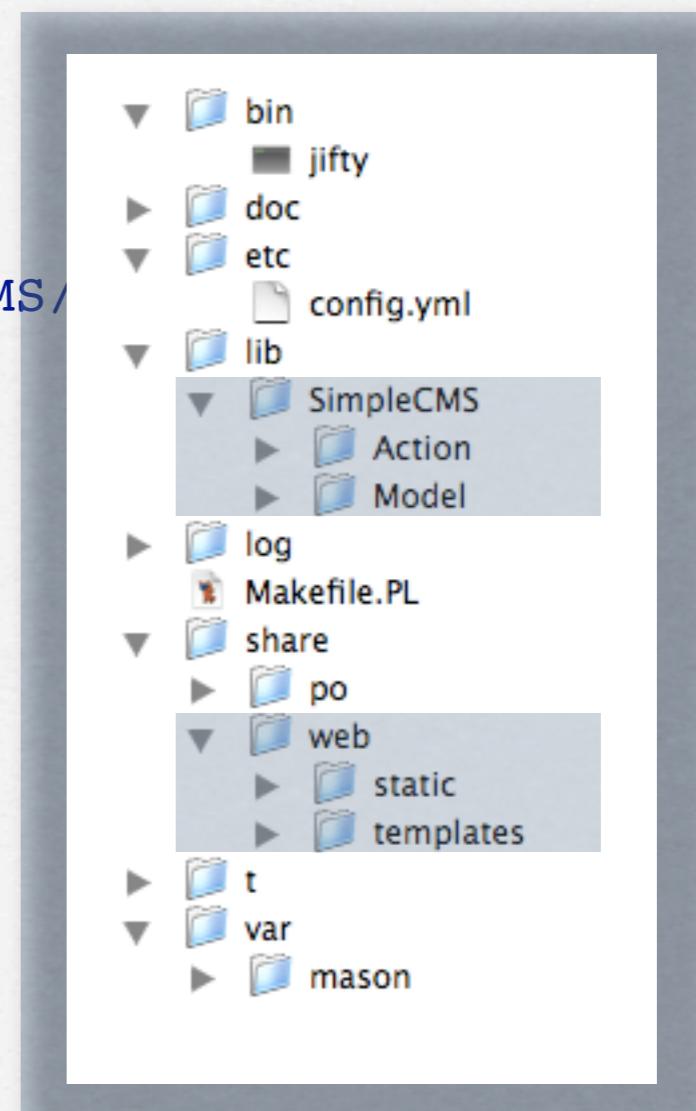
# Jifty Architektur



# Anlegen eines Projekts

## □ Erstellung einer „application“

```
$ jifty app --name SimpleCMS
Creating new application SimpleCMS
Creating directory lib
....
Creating configuration file SimpleCMS/
$
```

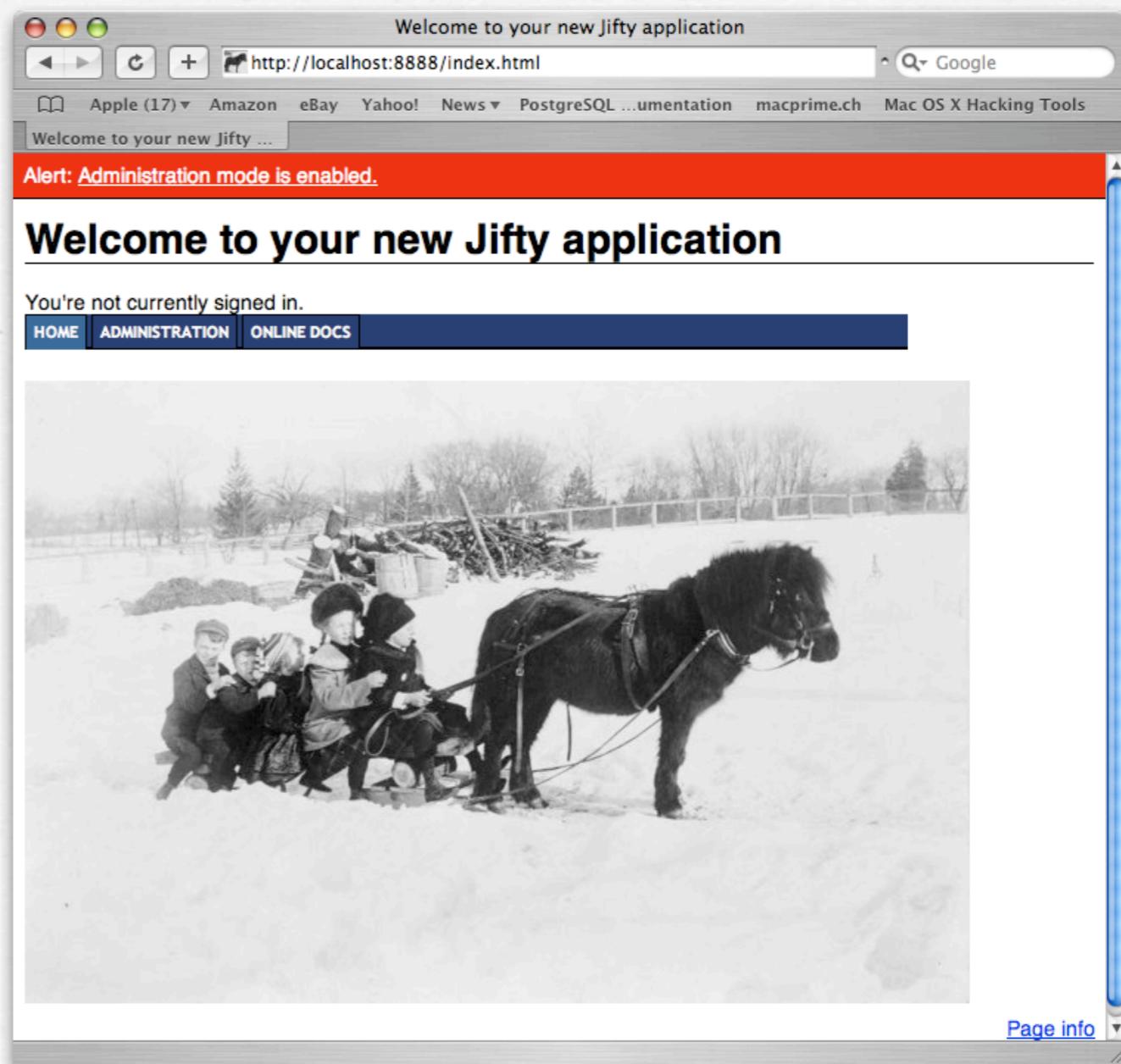


# Erster Start

## □ Eingebauter Web-Server

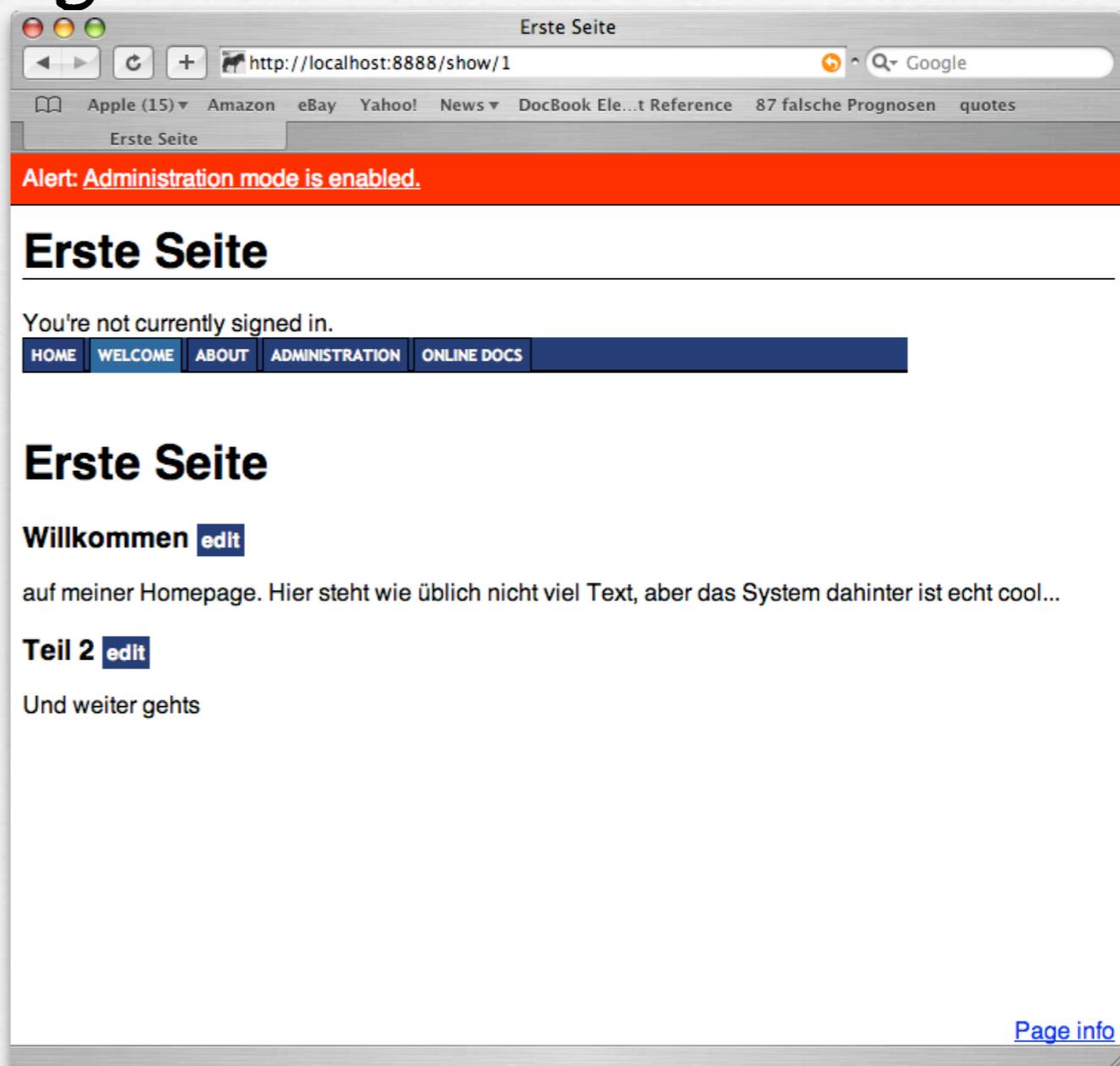
```
$ cd SimpleCMS  
$ ./bin/jifty server
```

INFO – You can connect



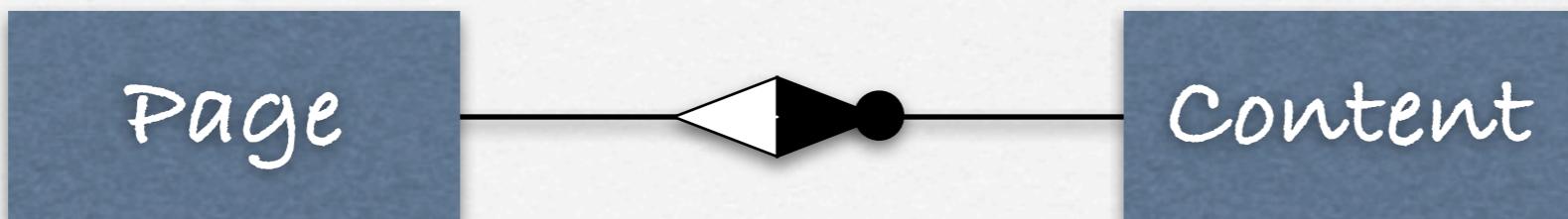
# Das Ziel: einfaches CMS

- in-Place Editierung
- Dynamischer Tausch von Teilen der Seite
- AJAX



# Unser Datenmodell

- „Page“
  - Navigations-Text, Headline
- „Content“
  - Zwischenüberschrift, Inhalt



# Model

- Perl Klasse zur Abstrahierung von Daten
- „Model“ = Datenbank-Tabelle
- Beschreibungssprache zur Definition
- Hinterlegung von Formular-Anzeige
- Versionierung der Datenbank-Definition
- Abstrahierung von SQL Kommandos
- „Collections“ für  $n (n \geq 0)$  Datensätze

# Beispiel Model

```
use strict;
use warnings;
package SimpleCMS::Model::Page;
use Jifty::DBI::Schema;

use SimpleCMS::Record schema {

    column 'name' => type is 'text',
            label is 'Name',
            render_as 'Text',
            since '0.0.1';

    column 'headline' => type is 'text',
            label is 'Headline',
            render_as 'Text',
            since '0.0.1';

};

# Your model-specific methods go here

1;
```

Erzeugung mit:

\$ ./bin/jifty model --name Page

syntaktisch korrektes Perl

Datenbank anlegen/ändern:

\$ ./bin/jifty schema --setup

# Daten anlegen

- Eingebaute Administrations-Oberfläche
  - ✓ nutzt selbst Jifty's Features
  - ✓ Anlegen, Bearbeiten und Suchen

The screenshot shows a web browser window with the URL [http://localhost:8888/\\_jifty/admin/model/Page](http://localhost:8888/_jifty/admin/model/Page). The title bar of the browser window also displays this URL. The page content includes:

- A red alert bar at the top stating "Alert: Administration mode is enabled."
- A message "You're not currently signed in." followed by a navigation menu with links for HOME, ADMINISTRATION, ONLINE DOCS, CONTENT, and PAGE.
- A section titled "Manage records: Page" listing existing records:
  - Welcome (Headline: Erste Seite) with an "Edit" link
  - About (Headline: Seite 2) with an "Edit" link
- A form for creating new records with fields for "Name" and "Headline", a "Create" button, and "Edit" links for the existing records.
- A "Done?" button.
- A link to "Back to the admin console".
- A "Page info" link in the bottom right corner.

# Templates

- HTML::MASON basiert
- Erstellung einzelner Komponenten
  - ✓ Funktionsblöcke, AJAX-Fragmente
- Logik im Template möglich (gut?)
- Einfache Erstellung von:
  - ✓ Formularen (incl. Prüfung und Autocomplete)
  - ✓ Links, Buttons
  - ✓ Dynamische Seiteninhalte (PageRegion)

# Beispiel Template

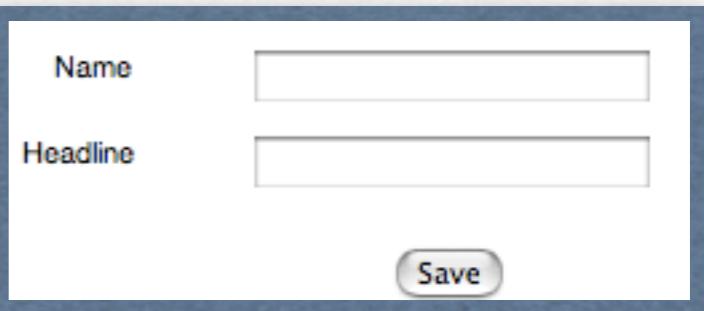
```
<%init>
my $action = Jifty->web->new_action(class => 'CreatePage');
<%init>

<% Jifty->web->form->start %>

<% $action->form_field('name') %>
<% $action->form_field('headline') %>

<% Jifty->web->form->submit(label => 'Save') %>

<% Jifty->web->form->end %>
```

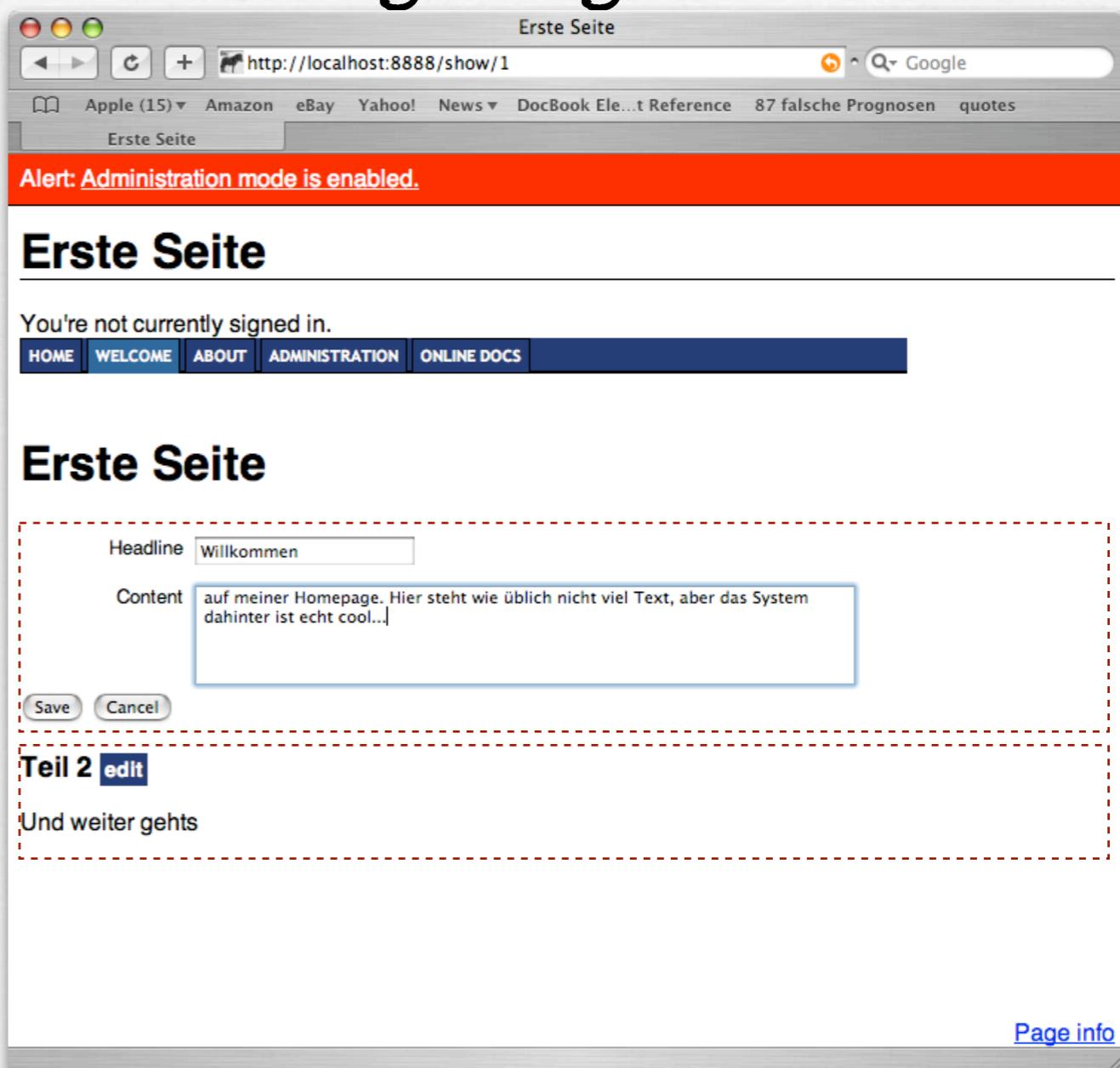


The image shows a screenshot of a web browser displaying a form template. The form consists of two text input fields: one labeled "Name" and another labeled "Headline", both currently empty. Below the input fields is a blue rectangular button labeled "Save". The entire form is contained within a dark grey rectangular border.

# CMS - Seitenaufbau

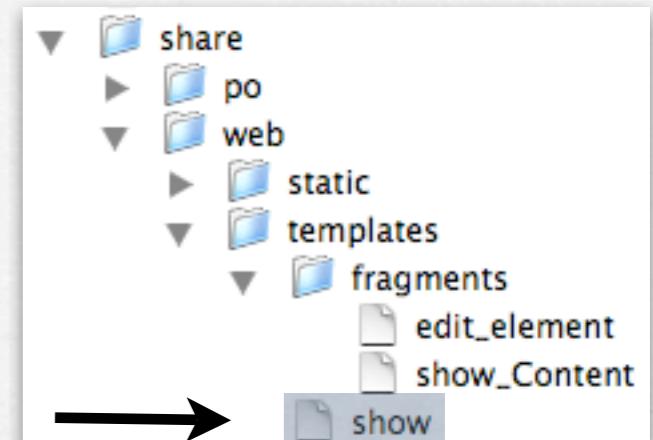
## □ Einteilung der Seite in PageRegions

- dynamischer Austausch
- auch ohne JavaScript lauffähig :-)



# Austausch von Inhalten

```
%# $content ist Collection von „Content“-Elementen  
...  
% while (my $element = $content->next) {  
% my $content_id = $element->id;  
  
<% Jifty->web->region(name      => "content$content_id",  
                         path       => '/fragments/show_Content',  
                         defaults  => { id => $content_id },  
                         ) %>  
  
% } # while  
...  
...
```



# Komponente: Darstellung

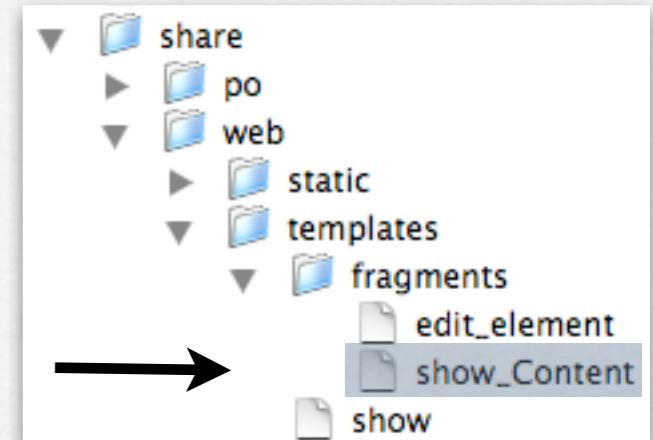
```
<%args>
$id
</%args>

<%init>
my $content = new SimpleCMS::Model::Content;
$content->load($id);
</%init>

<h3><% $content->headline %></h3>

<% Jifty->web->link(label    => 'edit',
                      onclick  => {
                        replace_with => '/fragments/edit_element',
                        args          => { id      => $id,
                                              model   => 'Content' },
                      },
                      ) %>

<% $content->content | n %>
```

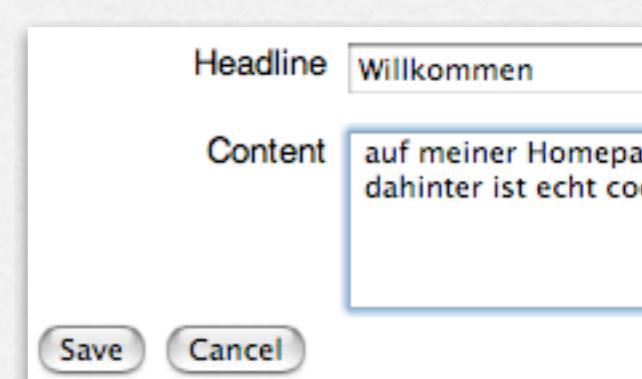
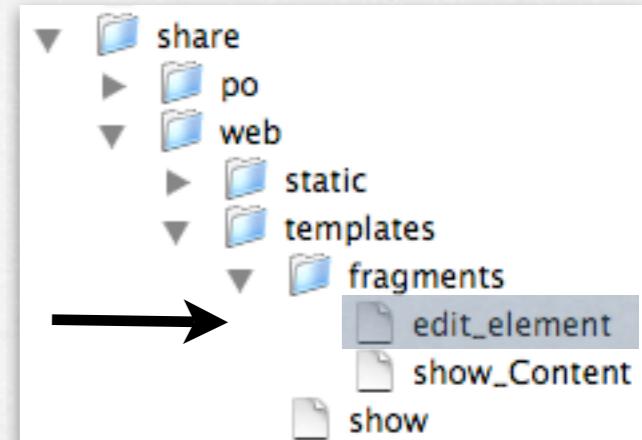


Willkommen **edit**  
auf meiner Homepage.

# Komponente: Bearbeiten

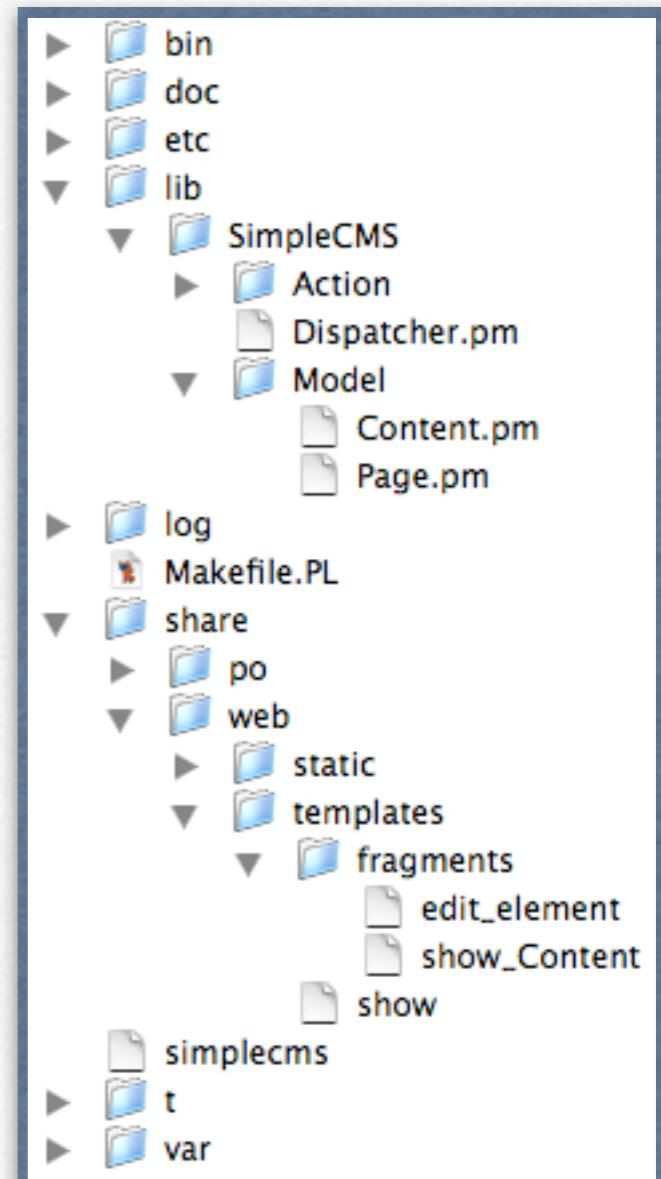
```
<%args>$id, $model</%args>
<%init>
my $item = "SimpleCMS::Model::$model" -> new();
$item->load($id);
my $action = Jifty->web->new_action(
    class      => "Update$model",
    moniker   => "edit$model-$id",
    record     => $item );
<%init>

% foreach my $field_name ($action->argument_names) {
    <% $action->form_field($field_name) %>
%
<% $action->button(label=> 'Save',
                    onclick => { submit      => $action,
                                replace_with => "/fragments/show_$model",
                                args        => { id=>$id } } ) %>
<% $action->button(label=> 'Cancel',
                    onclick => { submit      => [],
                                replace_with => "/fragments/show_$model",
                                args        => { id=>$id } } ) %>
```



# SimpleCMS Dateien

- Nur 160 Zeilen Code
- Funktionen:
  - Bearbeiten von Content
  - Datenbank unabhängig
  - Javascript unabhängig



# Weitere Jifty Features

- Dispatcher
  - ✓ „Logik in URL packen“
- Benutzerverwaltung
  - ✓ Log-in / Berechtigungsprüfungen
- Continuations
  - ✓ „GOTO“, „GOSUB“ und „RETURN“
- ausgeklügelte Vererbung
  - ✓ geschicktes Überladen hilft :-)
  - ✓ selbst CSS/Javascript „überladbar“

# Hilfe zur Selbsthilfe

- Projektseite: <http://jifty.org>
- CPAN: <http://search.cpan.org>
- perldoc Jifty::Manual::Tutorial [de]
- Inoffiziell: <http://wiki.kinkeldei-net.de>
- Oder: [www.tcool.org](http://www.tcool.org)

## Jifty ドキュメント日本語版

Jifty も Catalyst と並んで Ruby on Rails をおびやかすのではないかと噂されている Perl のフレームワークのひとつです。まだ開発初期の段階にあるため今後どのように推移していくかはわかりませんが、現状では HTML::Mason をテンプレート・クラスに用い、「継続(Continuation)」を使った状態管理をする方向に向かっています。現在のテキストはCPANにあがっている [Jifty-0.51228](#) をベースに、Subversion trunk の内容を多少反映させたものです。

Jifty rocks

<%end>

Danke für die Aufmerksamkeit

<%/end>

Tutorial: <http://www.kinkeldei.de/downloads>