Jifty

Audrey Tang
Best Practical
Best Practical

»|«  RT
Best Practical

»|« RT

»|« SVK
Best Practical

»|« RT

»|« SVK

»|« Hiveminder
Hiveminder
Hiveminder

»|«  Multiplayer TODO
Hiveminder

»|« Multiplayer TODO

»|« Web Application
Hiveminder

»|« Multiplayer TODO
»|« Web Application
»|« Just like RT
RT

»|«  Predates web frameworks
RT

»|« Predates web frameworks

»|« $^{20\text{th}}$ century technology
RT

»|« Predates web frameworks

»|« 20\textsuperscript{th} century technology

»|« Pain!
Use a Framework
Use a Framework

»|« We really want to
Use a Framework

»|« We really want to

»|« Nothing fits Jesse’s head
Use a Framework

»|« We really want to

»|« Nothing fits Jesse’s head

»|« This can only mean...
Jesse is Crazy
Jifty
Jifty

✓ Web framework
Jifty

✓ Web framework
✓ Easy things easy
Jifty

✓ Web framework
✓ Easy things easy
✓ Impossible things easy
Jifty = JFDI
Jifty = JFDI

✓ Just Do It
Jifty = JFDI

✓ Just Do It
✓ Lets you JFDI
Jifty = JFDI

✓ Just Do It
✓ Lets you JFDI
✓ So let’s JFDI!
Demo
Demo
✓ How about a Blog?
Demo

✓ How about a Blog?
✓ Glad you like it ;)}
Demo

✓ How about a Blog?
✓ Glad you like it ;)
✓ Let’s set it up!
$ jifty app --name Blog
$ jifty app --name Blog
Creating new application Blog
Creating directory lib
Creating directory lib/Blog
Creating directory bin
Creating directory etc
Creating directory doc
Creating directory log
Creating directory var
Creating directory var/mason
Creating directory share
Creating directory share/po
Creating directory share/web
Creating directory share/web/templates
Creating directory share/web/static
Creating directory lib/Blog/Model
Creating directory lib/Blog/Action
Creating directory t
Creating configuration file Blog/etc/config.yml
3 Files
3 Files

✓ bin/jifty
3 Files

✓ bin/jifty
✓ etc/config.yml
3 Files

✓ bin/jifty
✓ etc/config.yml
✓ Makefile.PL
It runs!
It runs!

This is not a good blog. (Good blogs have entries.)
$ cd Blog
$ cd Blog

$ jifty model --name Entry
$ cd Blog

$ jifty model --name Entry
Writing file t/00-model-Entry.t
Writing file lib/Blog/Model/Entry.pm
$ cd Blog

$ jifty model --name Entry
Writing file t/00-model-Entry.t
Writing file lib/Blog/Model/Entry.pm

$ vim lib/Blog/Model/Entry.pm
use strict;
use warnings;

package Blog::Model::Entry;
use Jifty::DBI::Schema;

use Blog::Record schema {

};

# Your model-specific methods go here.

1;
use strict;
use warnings;

package Blog::Model::Entry;
use Jifty::DBI::Schema;

use Blog::Record schema {

column title =>
    type is 'text',
    default is 'Untitled';

column body =>
    type is 'text',
    render as 'Textarea';

};

# Your model-specific methods go here.

1;
$ jifty schema
$ jifty schema
INFO - Generating SQL for application Blog...
INFO - Using Blog::Model::Entry
INFO - Using Jifty::Model::Session
INFO - Using Jifty::Model::Metadata
INFO - Set up version v0.0.1, jifty version 0.610250
$ jifty schema
INFO - Generating SQL for application Blog...
INFO - Using Blog::Model::Entry
INFO - Using Jifty::Model::Session
INFO - Using Jifty::Model::Metadata
INFO - Set up version v0.0.1, jifty version 0.610250

$ jifty server
$ jifty schema
INFO - Generating SQL for application Blog...
INFO - Using Blog::Model::Entry
INFO - Using Jifty::Model::Session
INFO - Using Jifty::Model::Metadata
INFO - Set up version v0.0.1, jifty version 0.610250

$ jifty server
INFO - You can connect to your server at
  http://localhost:8888/
Welcome to your new Jifty application

You're not currently signed in.
Welcome to your new Jifty application

You're not currently signed in.

Alert: Administration mode is enabled.
Database Administration

This console lets you manage the records in your Jifty database. Below, you should see a list of all your database tables. Feel free to go through and add, delete or modify records.

To disable this administrative console, add "framework: AdminMode: 0" to your application's configuration file.

Models

- Entry

Actions

- Jifty::Action::Autocomplete
- Jifty::Action::Redirect

Done?

Back to the application
Jifty Administrative Console

You're not currently signed in.

Database Administration

This console lets you manage the records in your Jifty database. Below, you should see a list of all your database tables. Feel free to go through and add, delete or modify records.

To disable this administrative console, add "framework: AdminMode: 0" to your application's configuration file.

Models

- Entry

Actions

- Jifty::Action::Autocomplete
- Jifty::Action::Redirect

Done?

Back to the application
You're not currently signed in.

Manage records: Entry

No items found.

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Post</td>
<td>This is my blog post!</td>
</tr>
</tbody>
</table>

Create

Done?

Back to the admin console

Hotkeys:
You're not currently signed in.

Manage records: Entry

No items found.

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Post</td>
<td>This is my blog post!</td>
</tr>
</tbody>
</table>

Done?

Back to the admin console

Hotkeys:
Alert: Administration mode is enabled.

You're not currently signed in.

Manage records: Entry

First Post  This is my blog post!

Edit

Create

Done?

Back to the admin console

Hotkeys:
Alert: Administration mode is enabled.

You're not currently signed in.

Manage records: Entry

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Post</td>
<td>This is my blog post! (Redux)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untitled</td>
<td></td>
</tr>
</tbody>
</table>

Save

Create

Done?

Back to the admin console

Hotkeys:
Alert: Administration mode is enabled.

You're not currently signed in.

Manage records: Entry

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Post</td>
<td>This is my blog post! (Redux)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untitled</td>
<td></td>
</tr>
</tbody>
</table>

Done?

Back to the admin console

Hotkeys:
Alert: Administration mode is enabled.

You're not currently signed in.

Updated

Manage records: Entry

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Post</td>
<td>This is my blog post! (Redux)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>title</th>
<th>body</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untitled</td>
<td></td>
</tr>
</tbody>
</table>

Edit

Create

Done?

Back to the admin console

Hotkeys:
You're not currently signed in.

Manage records: Entry

First Post  This is my blog post! (Redux)

Nice!

(Still, not really a blog.)
lib/Blog/Dispatcher.pm

(Map URLs to view logic.)
package Blog::Dispatcher;
use Jifty::Dispatcher -base;
package Blog::Dispatcher;
use Jifty::Dispatcher -base;

before '/*' => run {
    my $top = Jifty->web->navigation;
    $top->child( 'List Entries' => url => '/');
    $top->child( 'New Entry' => url => '/new_entry');
};
package Blog::Dispatcher;
use Jifty::Dispatcher -base;

before '('* => run {
    my $top = Jifty->web->navigation;
    $top->child( 'List Entries' => url => '/');
    $top->child( 'New Entry' => url => '/new_entry');
};

on '/new_entry' => run {
    set create => Jifty->web->new_action(
        class => 'CreateEntry'
    );
};
package Blog::Dispatcher;
use Jifty::Dispatcher -base;

before '*' => run {
    my $top = Jifty->web->navigation;
    $top->child( 'List Entries' => url => '/');
    $top->child( 'New Entry' => url => '/new_entry');
};

on '/new_entry' => run {
    set create => Jifty->web->new_action(
        class => 'CreateEntry'
    );
};

on '/' => run {
    my $entries = Blog::Model::EntryCollection->new;
    $entries->unlimit;
    set entries => $entries;
};
Alert: Administration mode is enabled.

Create an article

You're not currently signed in.

<table>
<thead>
<tr>
<th>HOME</th>
<th>LIST ENTRIES</th>
<th>NEW ENTRY</th>
<th>ADMINISTRATION</th>
<th>ONLINE DOCS</th>
</tr>
</thead>
</table>

**Title:** Untitled

**Body:** I’m in deep shit if the NSA intercept this

Save

Hotkeys:
$create # From the dispatcher
</ARGS>
$create # From the dispatcher

<&|/_elements/wrapper, title => 'Create an article' &>
$create # From the dispatcher

&lt;&lt;elements/wrapper, title =&gt; 'Create an article' &gt;&gt;
$create # From the dispatcher

&t_elements/wrapper, title => 'Create an article' &>
% my $form = Jifty->web->form;
% $form->start %
% for my $arg ($create->argument_names) {
    % $create->form_field($arg) %
% }
\$create  # From the dispatcher
</%ARGS>

\&|/_elements/wrapper, title \=> 'Create an article' \&>
% my $form = Jifty->web->form;
% $form->start %>
% for my $arg ($create->argument_names) {
    % $create->form_field($arg) %>
% }
% $form->submit( label \=> 'Save' ) %>
% $form->end %>
</&>
share/web/templates/index.html

Alert: Administration mode is enabled.

You're not currently signed in.

First Post
This is my blog post! (Redux)

Untitled
I'm in deep shit if the NSA intercept this

Hotkeys:
Alert: Administration mode is enabled.

You're not currently signed in.

First Post
This is my blog post! (Redux)

Untitled
I'm in deep shit if the NSA intercept this

Hotkeys:
$entries # From the dispatcher
</%ARGS>
$entries # From the dispatcher

% while (my $entry = $entries->next) {


$entries # From the dispatcher

while (my $entry = $entries->next) {

    <h2><% $entry->title %></h2>
    <div class="body">
        <% $entry->body %>
    </div>

}
$entries # From the dispatcher

% while (my $entry = $entries->next) {

<h2>% $entry->title %</h2>
<div class="body">
    % $entry->body
</div>

}
All Done!
All Done!

✓ lib/Blog/Dispatcher.pm
All Done!

✓ lib/Blog/Dispatcher.pm
✓ lib/Blog/Model/Entry.pm
All Done!

✓ lib/Blog/Dispatcher.pm
✓ lib/Blog/Model/Entry.pm
✓ web/templates/new_entry
All Done!

✓ lib/Blog/Dispatcher.pm
✓ lib/Blog/Model/Entry.pm
✓ web/templates/new_entry
✓ web/templates/index.html
Blog Demo
Blog Demo

😊 It runs!
Blog Demo

😊 It runs!

😊 Ship it!
Blog Demo

😊 It runs!

😊 Ship it!

😊 Oh wait...
New U.S. Law
New U.S. Law

😊 Banning profanity
New U.S. Law

😊 Banning profanity

😊 To protect the children
New U.S. Law

😊 Banning profanity
😊 To protect the children
😊 ...from the internet
Blog Censorship
Blog Censorship

😊 No problem
Blog Censorship

😊 No problem

😊 Back to the code!
Blog Censorship

😊 No problem
😊 Back to the code!
😊 lib/Blog/Model/Entry.pm
package Blog::Model::Entry;
package Blog::Model::Entry;

use Regexp::Common 'profanity_us';
package Blog::Model::Entry;

use Regexp::Common 'profanity_us';

sub validate_body {
    my ($self, $body) = @_;
package Blog::Model::Entry;

use Regexp::Common 'profanity_us';

sub validate_body {
    my ($self, $body) = @_;

    if ($body =~ /$RE{profanity}/i) {
        return (0, 'Would you speak like that . 
            'in front of your mother? *cough*');
    }
}
package Blog::Model::Entry;

use Regexp::Common 'profanity_us';

sub validate_body {
    my ($self, $body) = @_;

    if ($body =~ /$RE{profanity}/i) {
        return (0, 'Would you speak like that. ' .
                'in front of your mother? *cough*');
    }

    return 1;
}

Create an article

You're not currently signed in.

title
My life

body
I am in deep shit if the NSA have been monitoring my phone calls

Would you speak like that in front of your mother? *cough*
Validator

😊 Not good enough
Validator

😊 Not good enough

😊 Stops people from blogging
Validator

😊 Not good enough
😊 Stops people from blogging
😊 Let’s clean it up directly!
package Blog::Model::Entry;
use Regexp::Common 'profanity_us';

sub canonicalize_body {
    my ($self, $body) = @_;

    $body =~ s/$RE{profanity}/**expletives**/gi;
    return $body;
}
Create an article

You're not currently signed in.

**Alert:** Administration mode is enabled.

**Create an article**

You're not currently signed in.

- HOME
- LIST ENTRIES
- NEW ENTRY
- ADMINISTRATION
- ONLINE DOCS

**title**

Untitled

**body**

I'm in deep **expletives** if the NSA intercept this!

[Save]

Hotkeys:
Canonicalizer
Canonicalizer

😊 Part of the model
Canonicalizer

😊 Part of the model

😊 Same for autocompleters
Canonicalizer

😊 Part of the model
😊 Same for autocompleters
😊 *Now we ship it!*
Anatomy of a Jifty App
Anatomy of a Jifty App

😊 Model : Columns
Anatomy of a Jifty App

😊 Model : Columns

😊 Action : Parameters
Anatomy of a Jifty App

😊 Model: Columns
😊 Action: Parameters
😊 Template: Fragments
More Components
More Components

😊 Dispatcher
More Components

😊 Dispatcher

😊 ClassLoader
More Components

- Dispatcher
- ClassLoader
- Continuations
Jifty::DBI
Jifty::DBI

CREATE TABLE
Jifty::DBI

Ø CREATE TABLE
Ø ALTER TABLE
Jifty::DBI

- CREATE TABLE
- ALTER TABLE
- CREATE FOREIGN KEY
Object Relational Mapper
Object Relational Mapper

∅ SELECT
Object Relational Mapper

Ø SELECT

Ø INSERT
Object Relational Mapper

∅ SELECT  ∅ UPDATE

∅ INSERT
Object Relational Mapper

Ø SELECT
Ø INSERT
Ø UPDATE
Ø DELETE
Jifty::DBI::Handle
Jifty::DBI::Handle

SQLite
Jifty::DBI::Handle

- SQLite
- Postgres
Jifty::DBI::Handle

- SQLite
- MySQL
- Postgres
Jifty::DBI::Handle

- SQLite
- Postgres
- MySQL
- Oracle
Jifty::DBI::Collection

Joins and Limits
Jifty::DBI::Collection

- Joins and Limits
- Paging = Data::Page
Jifty::DBI::::Collection

- Joins and Limits
- Paging = Data::::Page
- Caching = Memcached
Jifty::DBI::Schema
Jifty::DBI::::Schema

- Declarative Columns
Jifty::DBI::::Schema

- Declarative Columns
- Remember our Blog?
Jifty::DBI::Schema

- Declarative Columns
- Remember our Blog?
- Time for 0.0.2!
package Blog::Model::Comment;
use Jifty::DBI::Schema;
use Blog::Record schema {

since '0.0.2';
package Blog::Model::Comment;
use Jifty::DBI::Schema;
use Blog::Record schema {

since '0.0.2';

column entry => refers to Blog::Model:Entry;
column author => refers to Blog::Model::User;
package Blog::Model::Comment;
use Jifty::DBI::Schema;
use Blog::Record schema {

since '0.0.2';

column entry => refers to Blog::Model::Entry;
column author => refers to Blog::Model::User;

column body =>
  label is 'Content',
type is 'Text',
render as 'Textarea',
default is 'First Post!';
package Blog::Model::Comment;
use Jifty::DBI::Schema;
use Blog::Record schema {

since '0.0.2';

column entry => refers to Blog::Model:Entry;
column author => refers to Blog::Model::User;

column body =>
  label is 'Content',
type is 'Text',
!15
render as 'Textarea',
default is 'First Post!';

column title =>
  is mandatory,
  label is 'Title',
  length is 40,
  hints is '40 bytes max';
package Blog::Model::Comment;
use Jifty::DBI::Schema;
use Blog::Record

schema {

  since '0.0.2';

  column entry => refers to Blog::Model::Entry;
  column author => refers to Blog::Model::User;

  column body =>
    label is 'Content',
    type is 'Text',
    render as 'Textarea',
    default is 'First Post!';

  column title =>
    is mandatory,
    label is 'Title',
    length is 40,
    hints is '40 bytes max';

};
Database Upgrade
Database Upgrade

- Used to be painful
Database Upgrade

- Used to be painful
- Not anymore
Database Upgrade

- Used to be painful
- Not anymore
- Jifty fixed it!
$ vim etc/config.yml
$ vim etc/config.yml

(Set version to 0.0.2)
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
Application schema version in database (v0.0.1) doesn't match application schema version (0.0.2)
Please run `bin/jifty schema --setup` to upgrade the database.
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
Application schema version in database (v0.0.1) doesn't match application schema version (0.0.2) Please run `bin/jifty schema --setup` to upgrade the database.

$ jifty schema --setup
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
Application schema version in database (v0.0.1) doesn't match application schema version (0.0.2)
Please run `bin/jifty schema --setup` to upgrade the database.

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog v0.0.1 database to v0.0.2
INFO - Upgraded to version v0.0.2
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
Application schema version in database (v0.0.1) doesn't match application schema version (0.0.2)
Please run `bin/jifty schema --setup` to upgrade the database.

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog v0.0.1 database to v0.0.2
INFO - Upgraded to version v0.0.2

$ jifty server
$ vim etc/config.yml
(Set version to 0.0.2)

$ jifty server
Application schema version in database (v0.0.1) doesn't match application schema version (0.0.2)
Please run `bin/jifty schema --setup` to upgrade the database.

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog
  v0.0.1 database to v0.0.2
INFO - Upgraded to version v0.0.2

$ jifty server
INFO - You can connect to your server at http://localhost:8888/
column tags => since '0.0.3', refers to Blog::Model::TagCollection by 'entry';
column tags =>
  since '0.0.3',
  refers to Blog::Model::TagCollection by 'entry';

column mood =>
  since '0.0.4',
  default is 'happy',
  valid are
    { display => 'Sad', value => 'sad' },
    { display => 'Happy', value => 'happy' };
column **tags** =>
since '0.0.3',
refers to Blog::Model::TagCollection by 'entry';

column **mood** =>
since '0.0.4',
default is 'happy',
valid are
  
    { display => 'Sad', value => 'sad' },
    { display => 'Happy', value => 'happy' };

column **created** =>
since '0.0.5',
**is** immutable,
type is 'timestamp',
default is literal 'now()',
filters are 'Jifty::DBI::Filter::DateTime';
$ vim etc/config.yml
$ vim etc/config.yml
(Set version to 0.0.5)
$ vim etc/config.yml
(Set version to 0.0.5)

$ jifty schema --setup
$ vim etc/config.yml
(Set version to 0.0.5)

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog v0.0.2 database to v0.0.5
INFO - Upgraded to version v0.0.5
$ vim etc/config.yml
(Set version to 0.0.5)

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog
v0.0.2 database to v0.0.5
INFO - Upgraded to version v0.0.5

$ jifty server
$ vim etc/config.yml
(Set version to 0.0.5)

$ jifty schema --setup
Jifty version 0.605070 up to date.
INFO - Generating SQL to upgrade Blog
    v0.0.2 database to v0.0.5
INFO - Upgraded to version v0.0.5

$ jifty server
INFO - You can connect to your server at
    http://localhost:8888/
Action = APIs
Action = APIs

⇒ Method Calls
Action = APIs

⇛ Method Calls
⇛ Typed named arguments
Action = APIs

⇨ Method Calls
⇨ Typed named arguments
⇨ Returns results or errors
Action ≠ URLs
Action ≠ URLs

⇒ Many actions on one page
Action ≠ URLs

⇒ Many actions on one page
⇒ Called with web forms
Action ≠ URLs

⇒ Many actions on one page
⇒ Called with web forms
⇒ Sounds like RPC?
Action = RPC
Action = RPC

⇒ Web form encoding
Action = RPC

⇒ Web form encoding
⇒ XML/YAML encoding
Action = RPC

⇒ Web form encoding
⇒ XML/YAML encoding
⇒ Web services for free!
Model $\Rightarrow$ Action
Model $\Rightarrow$ Action

$\Rightarrow$ Blog::Action::CreateEntry
Model $\Rightarrow$ Action

$\Leftrightarrow$ Blog::Action::CreateEntry

$\Leftrightarrow$ Blog::Action::SearchEntry
Model $\Rightarrow$ Action

$\Rightarrow$ Blog::Action::CreateEntry

$\Rightarrow$ Blog::Action::SearchEntry

$\Rightarrow$ Blog::Action::UpdateEntry
Model $\implies$ Action

$\implies$ Blog::Action::CreateEntry

$\implies$ Blog::Action::SearchEntry

$\implies$ Blog::Action::UpdateEntry

$\implies$ Blog::Action::DeleteEntry
Make your own Actions
package Hello::Action::SayHi;
use Jifty::Param::Schema;
use Jifty::Action schema {

  param name =>
    label is 'Your name?';

};
package Hello::Action::SayHi;
use Jifty::Param::Schema;
use Jifty::Action schema {

  param name =>
    label is 'Your name?';

};

sub take_action {
  my $self = shift;
  my $name = $self->argument_value('name');
  $self->result->message( "Hello, $name" );
}
<%init>
my $action = Jifty->web->new_action(
    class => 'SayHi'
);
</%init>
<%init>
my $action = Jifty->web->new_action(
    class => 'SayHi'
);
</%init>

<&|/_elements/wrapper&>
<% Jifty->web->form->start %>
<% $action->form_field('name') %>
<% Jifty->web->form->submit %>
<% Jifty->web->form->end %>
</&>
<%init>
my $action = Jifty->web->new_action(
    class => 'SayHi'
);
</%init>

<&_elements/wrapper&>
<% Jifty->web->form->start %>
<% $action->form_field('name') %>
<% Jifty->web->form->submit %>
<% Jifty->web->form->end %>
</&>
my $action = Jifty->web->new_action(
    class => 'SayHi'
);
</%init>

&lt;&|/_elements/wrapper&
&lt;% Jifty->web->form->start %
&lt;% $action->form_field('name') %
&lt;% Jifty->web->form->submit %
&lt;% Jifty->web->form->end %
&lt;/&>
$ curl \\
$ curl \

1: !!perl/hash:Jifty::Result
   _content: {}
   action_class: Hello::Action::SayHi
   failure: 0
   field_errors:
      name: ~
   field_warnings:
      name: ~
   message: 'Hello, Jesse'
Jifty::Dispatcher
Dispatcher = Kernel
Dispatcher = Kernel

_simple
Dispatcher = Kernel

- Simple
- Declarative
Dispatcher = Kernel

- Simple
- Declarative
- Encapsulates view logic
Three Stages
Three Stages

Before: Sets up state
Three Stages

☸ Before: Sets up state
☸ On: Shows your pages
Three Stages

- Before: Sets up state
- On: Shows your pages
- After: Cleans up state
Jifty::ClassLoader
Scaffolding Bad
Scaffolding Bad

☸ Ugly
Scaffolding Bad

- Ugly
- Redundant
Scaffolding Bad

- Ugly
- Redundant
- Fragile
Scaffolding Bad

- Ugly
- Annoying
- Redundant
- Fragile
Scaffolding Bad

- Ugly
- Redundant
- Fragile
- Annoying
- Redundant
Scaffolding Bad

✪ Ugly
✪ Redundant
✪ Fragile

✪ Annoying
✪ Redundant
✪ Redundant
Why Empty Classes?
Why *Empty* Classes?

- Jifty::ClassLoader
Why Empty Classes?

- Jifty::ClassLoader
- Auto-creates classes
Why Empty Classes?

- Jifty::ClassLoader
- Auto-creates classes
- Never writes them to disk
Auto-created Classes
Auto-created Classes

Blog
Auto-created Classes

- Blog
- Blog::Record
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
- Blog::Model::EntryCollection
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
- Blog::Model::EntryCollection
- Blog::Action::CreateEntry
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
- Blog::Model::EntryCollection
- Blog::Action::CreateEntry
- Blog::Action::SearchEntry
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
- Blog::Model::EntryCollection
- Blog::Action::CreateEntry
- Blog::Action::SearchEntry
- Blog::Action::UpdateEntry
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser
- Blog::Model::EntryCollection
- Blog::Action::CreateEntry
- Blog::Action::SearchEntry
- Blog::Action::UpdateEntry
- Blog::Action::DeleteEntry
Auto-created Classes

- Blog
- Blog::Record
- Blog::Collection
- Blog::Bootstrap
- Blog::Upgrade
- Blog::Dispatcher
- Blog::CurrentUser

- Blog::Model::EntryCollection
- Blog::Action::CreateEntry
- Blog::Action::SearchEntry
- Blog::Action::UpdateEntry
- Blog::Action::DeleteEntry
- ...et cetera
Continuation = Job Control
Continuation = Job Control

Continuations in Perl?
Continuation = Job Control

☯ Continuations in Perl?

☯ Not Full Continuations
Continuation = Job Control

- Continuations in Perl?
- Not Full Continuations
- Delimited Continuations
Full Continuations
Full Continuations

 Serialized everything
Full Continuations

☯ Serializes everything

☯ %ENV, STDIN, STDOUT
Full Continuations

☯ Serializes everything
☯ %ENV, STDIN, STDOUT
☯ DBI Handle, Log Handle
Full Continuations

- Serializes everything
- %ENV, STDIN, STDOUT
- DBI Handle, Log Handle
- Phase of the moon
Jifty Continuations
Jifty Continuations

☯ Only the application state
Jifty Continuations

.paused Only the application state

.paused Requests and Actions
Jifty Continuations

☯ Only the application state

☯ Requests and Actions

☯ That’s all we need
Why Continuations?
Why Continuations?

☯ The web is built on GOTO
Why Continuations?

☯ The web is built on GOTO

☯ Do you *like* GOTO?
Why Continuations?

☯ The web is built on GOTO

☯ Do you *like* GOTO?

☯ Continuations are GOSUB
Why Continuations?

☯ The web is built on GOTO

☯ Do you like GOTO?

☯ Continuations are GOSUB

☯ Welcome to Civilization!
A Brief Tangent...
A Brief Tangent...

☯ Back to our Blog
A Brief Tangent...

☯ Back to our Blog

☯ Requiring Login
A Brief Tangent...

☯ Back to our Blog

☯ Requiring Login

☯ lib/Blog/Dispatcher.pm
package Blog::Action::Login;
use Jifty::Param::Schema;
use Jifty::Action schema {

param address =>
  is mandatory,
  ajax validates,
  label is 'Email address';

param password =>
  is mandatory,
  type is 'Password',
  label is 'Password';

};
package Blog::Action::Login;
use Jifty::Param::Schema;
use Jifty::Action schema {

param address =>
    is mandatory,
    ajax validates,
    label is 'Email address';

param password =>
    is mandatory,
    type is 'Password',
    label is 'Password';

};

sub take_action {
    my $self = shift;
    if ( $self->validate_login ) {
        $self->login_as_user( $self->user );
        return 1;
    }
    return 0;
}
package Blog::Dispatcher;

## Auth and login
before '*' => run {
    if (not Jifty->web->current_user->id ) {
        tangent '/login';
    }
};
my $action = Jifty->web->new_action(
    class => 'Login',
);

```merlin
my $action = Jifty->web->new_action(
    class => 'Login',
);

<h2>Login</h2>
<% Jifty->web->form->start %>
  <% $action->form_field('address') %>  
  <% $action->form_field('password') %>  
  <% Jifty->web->return( label => 'Login' ) %>
<% Jifty->web->form->end %>
```
Templates = View
Templates = View

HTML::Mason
Templates = View

- HTML::Mason
- Mason has Components
Templates = View

- HTML::Mason
- Mason has Components
- Server-side inclusion
Fragments
Fragments

Components 2.0
Fragments

Components 2.0

Client-side for AJAX
Fragments

- Components 2.0
- Client-side for AJAX
- Server-side for Lynx
\%
Jifty->web->region(
    name => "myweblog-posts",
    path => "/fragments/page_of_posts"
)
\%
Chat

🌟 Server-push (Comet)
Chat

🔹 Server-push (Comet)
🔹 Refreshed fragments
Chat

🌟 Server-push (Comet)
🌟 Refreshed fragments
🌟 Live Demo!
Development Tools
Development Tools

∀ Standalone Server
Development Tools

∀ Standalone Server
∀ Module::Install
Development Tools

- Standalone Server
- Module::Install
- PAR Distribution
Halos
∀ SQL Logging
Halos

∀ SQL Logging

∀ SQL Profiling
Halos

∀ SQL Logging
∀ SQL Profiling
∀ Template Profiling
Halos
Halos

∀  View Tracing
Halos

∀ View Tracing
∀ Online Editing
Halos

- View Tracing
- Online Editing
- ...and more!
Devel Mode
Devel Mode

∀ Module::Refresh
Devel Mode

∀ Module::Refresh

∀ Tracing with Log4Perl
Devel Mode

∀ Module::Refresh
∀ Tracing with Log4Perl
∀ Administrative Console
Deployment
Deployment

∀ Lighttpd / Apache
Deployment

- Lighttpd / Apache
- Fast Static Server
Deployment

∀ Lighttpd / Apache
∀ Fast Static Server
∀ CSS::Squish, JS::Squish
More Goodies
More Goodies

∀ Comprehensive I18N
More Goodies

∀ Comprehensive I18N
∀ Client-Side Sessions
More Goodies

∀ Comprehensive I18N
∀ Client-Side Sessions
∀ Events and Subscriptions
Future Plans
Future Plans

✈ Wiki-style UI Editor
Future Plans

✈ Wiki-style UI Editor

✈ Even less code
Future Plans

✈ Wiki-style UI Editor
✈ Even less code
✈ “Jifty in a Jiffy” MTVs
Sample Applications
Sample Applications

✈ Wifty
Sample Applications

✈ Wifty

✈ CommitBit
Sample Applications

✈️ Wifty
✈️ CommitBit
✈️ Ping / Chat / Clock
Get Jifty
Get Jifty

✈ cpan   Jifty
Get Jifty

_airplane_cpan  Jifty

_airplane_svn.jifty.org/svn/jifty.org
Get Jifty

気軽に Jifty を

- cpan Jifty
- svn.jifty.org/svn/jifty.org
- http://jifty.org/
Getting Started
Getting Started

✈ Jifty::Manual::Tutorial
Getting Started

✈ Jifty::Manual::Tutorial
✈ irc.freenode.net #jifty
Getting Started

✈ Jifty::Manual::Tutorial
✈ irc.freenode.net #jifty
✈ jifty-devel@lists.jifty.org
Thank you!