# ----------------BEGIN----------------





**Nitrogen - Web Framework for Erlang**

# Getting Started

## Is there a Nitrogen Tutorial?

Yes! You can view the updated slides from Rusty's Nitrogen Tutorial at the Erlang User Conference 2010 here.

http://nitrogenproject.com/doc/tutorial.html

## How do I create a new Nitrogen application?

If you don't see a package that will run on your computer, or you want to try running from the latest code in source control, then follow the steps below:

Pull the latest source code from GitHub:

**mkdir ~/erltk/nitrogen**

**cd ~/erltk/nitrogen**

**git clone https://github.com/nitrogen/nitrogen**

Then run:

**make rel\_inets**

lubuntu-xzm@lubuntuxzm-virtual-machine:~/erltoolkit/nitrogen$ cd nitrogen/

lubuntu-xzm@lubuntuxzm-virtual-machine:~/erltoolkit/nitrogen/nitrogen$ ls

CHANGELOG.markdown CONTRIB.markdown embed Makefile MIT-LICENSE README.markdown rebar rebar.config rel support THANKS.markdown TODO.markdown

lubuntu-xzm@lubuntuxzm-virtual-machine:~/erltoolkit/nitrogen/nitrogen$ make rel\_inets

make[1]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

./rebar get-deps

==> rel (get-deps)

==> nitrogen (get-deps)

./rebar compile

==> rel (compile)

==> nitrogen (compile)

make[2]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Creating full release in ../myapp with inets

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

make[2]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[3]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

==> rel (generate)

Copying /usr/local/lib/erlang/lib/erl\_interface-3.7.18

Generating "nitrogen/rebar.config"

make[4]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

Using Erlang in /home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen/erts-6.2/bin/erl

==> site (get-deps)

==> nitrogen (get-deps)

Pulling mimetypes from {git,"git://github.com/spawngrid/mimetypes.git",

{branch,master}}

正克隆到 'mimetypes'...

Pulling simple\_bridge from {git,"git://github.com/nitrogen/simple\_bridge",

{branch,master}}

正克隆到 'simple\_bridge'...

Pulling nprocreg from {git,"git://github.com/nitrogen/nprocreg",

{branch,master}}

正克隆到 'nprocreg'...

Pulling nitrogen\_core from {git,"git://github.com/nitrogen/nitrogen\_core",

{branch,master}}

正克隆到 'nitrogen\_core'...

Pulling sync from {git,"git://github.com/rustyio/sync",{branch,master}}

正克隆到 'sync'...

==> mimetypes (get-deps)

==> simple\_bridge (get-deps)

==> nprocreg (get-deps)

==> nitrogen\_core (get-deps)

==> sync (get-deps)

Checking for Nitrogen Plugins

No Nitrogen Plugins Found

Using Erlang in /home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen/erts-6.2/bin/erl

==> mimetypes (compile)

Compiled src/mimetypes\_scan.xrl

Compiled src/mimetypes\_parse.yrl

Compiled src/mimetypes\_parse.erl

Compiled src/mimetypes\_scan.erl

Compiled src/mimetypes\_app.erl

Compiled src/mimetypes\_sup.erl

Compiled src/mimetypes\_loader.erl

Compiled src/mimetypes.erl

==> simple\_bridge (compile)

Generating crypto compatibility for simple\_bridge...

...Using: "crypto:hash(sha, Data)"

...writing "include/crypto\_compat.hrl"

Compiled src/simple\_bridge\_handler.erl

Compiled src/simple\_bridge.erl

Compiled src/sb\_uploaded\_file.erl

Compiled src/simple\_bridge\_util.erl

Compiled src/yaws\_bridge\_modules/yaws\_simple\_bridge.erl

Compiled src/yaws\_bridge\_modules/yaws\_simple\_bridge\_anchor.erl

Compiled src/mochiweb\_bridge\_modules/mochiweb\_simple\_bridge\_sup.erl

Compiled src/yaws\_bridge\_modules/yaws\_simple\_bridge\_sup.erl

Compiled src/mochiweb\_bridge\_modules/mochiweb\_simple\_bridge\_anchor.erl

Compiled src/simple\_bridge\_websocket.erl

Compiled src/simple\_bridge\_handler\_sample.erl

Compiled src/mochiweb\_bridge\_modules/mochiweb\_simple\_bridge.erl

Compiled src/webmachine\_bridge\_modules/webmachine\_simple\_bridge\_anchor.erl

Compiled src/webmachine\_bridge\_modules/webmachine\_simple\_bridge\_sup.erl

Compiled src/webmachine\_bridge\_modules/webmachine\_simple\_bridge\_static.erl

Compiled src/cowboy\_bridge\_modules/cowboy\_request\_server.erl

Compiled src/simple\_bridge\_multipart.erl

Compiled src/webmachine\_bridge\_modules/webmachine\_simple\_bridge.erl

Compiled src/cowboy\_bridge\_modules/cowboy\_simple\_bridge\_anchor.erl

Compiled src/simple\_bridge\_app.erl

Compiled src/cowboy\_bridge\_modules/cowboy\_simple\_bridge\_sup.erl

Compiled src/inets\_bridge\_modules/inets\_simple\_bridge\_anchor.erl

Compiled src/cowboy\_bridge\_modules/cowboy\_simple\_bridge.erl

Compiled src/inets\_bridge\_modules/inets\_simple\_bridge\_sup.erl

Compiled src/inets\_bridge\_modules/inets\_simple\_bridge.erl

Compiled src/sb\_file\_upload\_handler.erl

Compiled src/sbw.erl

==> nprocreg (compile)

Compiled src/nprocreg\_app.erl

Compiled src/nprocreg\_sup.erl

Compiled src/nprocreg.erl

==> nitrogen\_core (compile)

Generating crypto compatibility...

...Using: "crypto:block\_encrypt(aes\_cbc128, Key, IV, Data)"

...Using: "crypto:block\_decrypt(aes\_cbc128, Key, IV, Data)"

...Using: "crypto:hash(sha, Data)"

...writing "include/crypto\_compat.hrl"

Compiled src/handlers/config/config\_handler.erl

Compiled src/handlers/route/route\_handler.erl

Compiled src/handlers/identity/identity\_handler.erl

Compiled src/handlers/role/role\_handler.erl

Compiled src/handlers/process\_registry/process\_registry\_handler.erl

Compiled src/handlers/crash/crash\_handler.erl

Compiled src/handlers/session/session\_handler.erl

Compiled src/handlers/state/state\_handler.erl

Compiled src/handlers/cache/cache\_handler.erl

Compiled src/handlers/log/log\_handler.erl

Compiled src/handlers/security/security\_handler.erl

Compiled src/handlers/query/query\_handler.erl

Compiled src/lib/wf\_cookies.erl

Compiled src/lib/wf\_action\_queue.erl

Compiled src/lib/wf\_render\_elements.erl

Compiled src/lib/wf\_handler.erl

Compiled src/lib/wf\_render\_actions.erl

Compiled src/lib/wf\_context.erl

Compiled src/lib/wf\_tags.erl

Compiled src/lib/wf\_test\_srv.erl

Compiled src/lib/wf\_pickle.erl

Compiled src/lib/wf\_convert.erl

Compiled src/lib/wf\_utils.erl

Compiled src/lib/nitro\_mochiglobal.erl

Compiled src/lib/nitro\_mochijson2.erl

Compiled src/lib/wf\_validation.erl

Compiled src/lib/wf\_event.erl

Compiled src/lib/nitro\_mochinum.erl

Compiled src/lib/wf\_test.erl

Compiled src/wf\_core.erl

Compiled src/wf.erl

Compiled src/actions/action\_disable\_selection.erl

Compiled src/actions/action\_slide\_up.erl

Compiled src/actions/action\_set\_cookie.erl

Compiled src/actions/action\_validate.erl

Compiled src/actions/action\_set\_multiple.erl

Compiled src/actions/action\_update.erl

Compiled src/actions/action\_fade.erl

Compiled src/actions/action\_clear\_validation.erl

Compiled src/actions/action\_set.erl

Compiled src/actions/action\_buttonize.erl

Compiled src/actions/action\_make\_readonly.erl

Compiled src/actions/action\_toggle.erl

Compiled src/actions/action\_enable.erl

Compiled src/actions/action\_redirect.erl

Compiled src/actions/action\_appear.erl

Compiled src/actions/action\_before\_postback.erl

Compiled src/actions/action\_script.erl

Compiled src/actions/action\_toggle\_mobile\_panel.erl

Compiled src/actions/action\_effect.erl

Compiled src/actions/action\_event.erl

Compiled src/actions/action\_click.erl

Compiled src/actions/action\_continue.erl

Compiled src/actions/action\_comet.erl

Compiled src/actions/action\_slide\_down.erl

Compiled src/actions/action\_hide.erl

Compiled src/actions/action\_animate.erl

Compiled src/actions/action\_alert.erl

Compiled src/actions/action\_wire.erl

Compiled src/actions/action\_remove\_class.erl

Compiled src/actions/action\_api.erl

Compiled src/actions/action\_make\_writable.erl

Compiled src/actions/action\_confirm.erl

Compiled src/actions/action\_add\_class.erl

Compiled src/actions/action\_console\_log.erl

Compiled src/actions/action\_validation\_error.erl

Compiled src/actions/action\_show.erl

Compiled src/actions/action\_jquery\_effect.erl

Compiled src/actions/action\_disable.erl

Compiled src/actions/action\_function.erl

Compiled src/elements/mobile/element\_mobile\_grid.erl

Compiled src/elements/mobile/element\_mobile\_list.erl

Compiled src/elements/mobile/element\_mobile\_grid\_block.erl

Compiled src/elements/mobile/element\_mobile\_listitem.erl

Compiled src/elements/mobile/element\_mobile\_collapsible.erl

Compiled src/elements/mobile/element\_mobile\_toggle.erl

Compiled src/elements/mobile/element\_mobile\_panel.erl

Compiled src/elements/mobile/element\_mobile\_collapsible\_set.erl

Compiled src/elements/mobile/element\_mobile\_list\_divider.erl

Compiled src/elements/table/element\_singlerow.erl

Compiled src/elements/table/element\_tablerow.erl

Compiled src/elements/table/element\_tableheader.erl

Compiled src/elements/table/element\_tablecell.erl

Compiled src/elements/table/element\_table.erl

Compiled src/elements/other/element\_gravatar.erl

Compiled src/elements/other/element\_iframe.erl

Compiled src/elements/other/element\_function.erl

Compiled src/elements/other/element\_sparkline.erl

Compiled src/elements/other/element\_google\_chart.erl

Compiled src/elements/other/element\_qr.erl

Compiled src/elements/other/element\_flash.erl

Compiled src/elements/other/element\_sync\_panel.erl

Compiled src/elements/other/element\_bind.erl

Compiled src/elements/other/element\_draggable.erl

Compiled src/elements/other/element\_progress\_bar.erl

Compiled src/elements/other/element\_sortitem.erl

Compiled src/elements/other/element\_recaptcha.erl

Compiled src/elements/other/element\_spinner.erl

Compiled src/elements/other/element\_sortblock.erl

Compiled src/elements/other/element\_file.erl

Compiled src/elements/other/element\_droppable.erl

Compiled src/elements/layout/element\_panel.erl

Compiled src/elements/layout/element\_fieldset.erl

Compiled src/elements/layout/element\_placeholder.erl

Compiled src/elements/layout/element\_lightbox.erl

Compiled src/elements/layout/element\_h.erl

Compiled src/elements/layout/element\_template.erl

Compiled src/elements/layout/element\_grid.erl

Compiled src/elements/html5/element\_article.erl

Compiled src/elements/html5/element\_time.erl

Compiled src/elements/html5/element\_main.erl

Compiled src/elements/html5/element\_aside.erl

Compiled src/elements/html5/element\_html5\_header.erl

Compiled src/elements/html5/element\_mark.erl

Compiled src/elements/html5/element\_html5\_footer.erl

Compiled src/elements/html5/element\_nav.erl

Compiled src/elements/html5/element\_section.erl

Compiled src/elements/forms/element\_textbox\_autocomplete.erl

Compiled src/elements/forms/element\_inplace\_textbox.erl

Compiled src/elements/forms/element\_textarea.erl

Compiled src/elements/forms/element\_restful\_reset.erl

Compiled src/elements/forms/element\_label.erl

Compiled src/elements/forms/element\_datepicker\_textbox.erl

Compiled src/elements/forms/element\_checkbox.erl

Compiled src/elements/forms/element\_restful\_submit.erl

Compiled src/elements/forms/element\_password.erl

Compiled src/elements/forms/element\_hidden.erl

Compiled src/elements/forms/element\_range.erl

Compiled src/elements/forms/element\_wizard.erl

Compiled src/elements/forms/element\_upload.erl

Compiled src/elements/forms/element\_inplace\_textarea.erl

Compiled src/elements/forms/element\_button.erl

Compiled src/elements/forms/element\_restful\_upload.erl

Compiled src/elements/forms/element\_textbox.erl

Compiled src/elements/forms/element\_radio.erl

Compiled src/elements/forms/element\_radiogroup.erl

Compiled src/elements/forms/element\_dropdown.erl

Compiled src/elements/forms/element\_restful\_form.erl

Compiled src/elements/forms/element\_inplace.erl

Compiled src/elements/html/element\_br.erl

Compiled src/elements/html/element\_email\_link.erl

Compiled src/elements/html/element\_em.erl

Compiled src/elements/html/element\_image.erl

Compiled src/elements/html/element\_span.erl

Compiled src/elements/html/element\_listitem.erl

Compiled src/elements/html/element\_literal.erl

Compiled src/elements/html/element\_pre.erl

Compiled src/elements/html/element\_link.erl

Compiled src/elements/html/element\_hr.erl

Compiled src/elements/html/element\_value.erl

Compiled src/elements/html/element\_list.erl

Compiled src/nitrogen.erl

Compiled src/elements/html/element\_strong.erl

Compiled src/elements/html/element\_p.erl

Compiled src/validators/validator\_js\_custom.erl

Compiled src/validators/validator\_custom.erl

Compiled src/validators/validator\_confirm\_password.erl

Compiled src/validators/validator\_is\_email.erl

Compiled src/validators/validator\_confirm\_same.erl

Compiled src/validators/validator\_max\_length.erl

Compiled src/validators/validator\_min\_length.erl

Compiled src/validators/validator\_is\_required.erl

Compiled src/validators/validator\_is\_integer.erl

Compiled src/nitrogen\_dev.erl

Compiled src/handlers/route/nitrogen\_smart\_extensions.erl

Compiled src/handlers/config/default\_config\_handler.erl

Compiled src/handlers/route/nitrogen\_rest.erl

Compiled src/handlers/route/dynamic\_route\_handler.erl

Compiled src/handlers/route/passthrough\_route\_handler.erl

Compiled src/handlers/route/file\_not\_found\_page.erl

Compiled src/handlers/identity/default\_identity\_handler.erl

Compiled src/handlers/role/default\_role\_handler.erl

Compiled src/handlers/route/named\_route\_handler.erl

Compiled src/handlers/process\_registry/nprocreg\_registry\_handler.erl

Compiled src/handlers/crash/debug\_crash\_handler.erl

Compiled src/handlers/crash/default\_crash\_handler.erl

Compiled src/handlers/session/simple\_session\_handler.erl

Compiled src/handlers/state/default\_state\_handler.erl

Compiled src/handlers/cache/default\_cache\_handler.erl

Compiled src/handlers/log/default\_log\_handler.erl

Compiled src/handlers/process\_registry/gproc\_registry\_handler.erl

Compiled src/handlers/security/http\_basic\_auth\_security\_handler.erl

Compiled src/handlers/security/default\_security\_handler.erl

Compiled src/handlers/query/default\_query\_handler.erl

==> sync (compile)

Compiled src/sync\_options.erl

Compiled src/sync\_notify.erl

Compiled src/sync.erl

Compiled src/sync\_utils.erl

Compiled src/sync\_scanner.erl

==> site (compile)

Compiled src/nitrogen\_main\_handler.erl

Compiled src/nitrogen\_app.erl

Compiled src/nitrogen\_sup.erl

Compiled src/index.erl

Compiled src/mobile.erl

==> nitrogen (compile)

make[4]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

make[4]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

Generating a default cookie in /etc/vm.args

make[4]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

make[4]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

make[4]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen/rel/nitrogen'

make[3]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]: 正在进入目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

make[2]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Generated a self-contained Nitrogen project

in ../myapp, configured to run on inets.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

make[1]:正在离开目录 `/home/lubuntu-xzm/erltoolkit/nitrogen/nitrogen'

lubuntu-xzm@lubuntuxzm-virtual-machine:~/erltoolkit/nitrogen/nitrogen$

This creates a completely self-contained starter application under ./rel/nitrogen that runs on **Inets, Erlang's built in HTTP server**. "Self-contained" here means that the ./rel/nitrogen directory contains everything you need to develop and run nitrogen, including the Erlang VM. To create a .tar.gz version of this code, exactly like you would find on the Downloads page (<http://nitrogenproject.com/downloads>), run:

http://downloads.nitrogenproject.com.s3.amazonaws.com/2.2.2/linux/nitrogen-2.2.2-inets.tar.gz

make package\_inets

You can also run Nitrogen with Mochiweb, Yaws, Cowboy, or Webmachine. Run one of the following make commands:

make rel\_cowboy

make rel\_mochiweb

make rel\_webmachine

make rel\_yaws

There are also "slim release" versions that can be built. These take advantage of a new feature introduced in Erlang's R15B02 release, which packages up an application, but does not include a full Erlang distribution (instead, it assumes the target system has Erlang installed already).

You can create these "slim releases" by typing:

make slim\_cowboy

make slim\_inets

make slim\_mochiweb

make slim\_webmachine

make slim\_yaws

## How do I add Nitrogen to an existing Erlang application?

As of 2.2.0, adding Nitrogen to an existing Erlang application is simple.

Clone Nitrogen somewhere on your machine:

mkdir ~/erltk/nitrogen-git

cd ~/erltk /nitrogen-git

git clone git://github.com/nitrogen/nitrogen.git

Then, from the root of your application's directory, run the embed script from the Nitrogen repo:

/path/to/nitrogen/embed

Note: You must run this from your \*application's\* directory, \*not\* from the Nitrogen directory.

Note 2: This assumes your system has Perl installed

You can read the full writeup in in this blog post.

<http://sigma-star.com/blog/post/embedding-nitrogen>

## How do I start and stop my application?

To start Nitrogen in the console

bin/nitrogen console

To start Nitrogen, type

bin/nitrogen start

~/erltk/**nitrogen/myapp/bin**/nitrogen start

To attach to a detached Nitrogen console

bin/nitrogen attach

To stop Nitrogen

bin/nitrogen stop

## How do I change configuration settings?

Read all about the configuration options.

http://nitrogenproject.com/doc/config.html

## Where is the code for my website?

Your entire site's code can be found in site/.

Within there, you'll find:

site/src

Your erlang code

site/static

Where static resources like CSS, Javascript, and Images Go

site/templates

Where your site's HTML templates go

site/ebin

Where the compiled BEAM files go.

If you're using a full release, it's recommended to put the site/ directory under source control.

If you're using a slim release, you can put the entire application under source control.

~/erltoolkit/nitrogen$ find . -name site

./myapp/site

./nitrogen/rel/overlay/common/site

## How do I compile my code?

## How do I create new pages, custom elements, or custom actions?

## How do I make a mobile version?

## How do I work with Nitrogen Plugins?

## How do I upgrade to a new version of Nitrogen?

## What do I do if I run into errors? How do I troubleshoot?

# Introduction to Nitrogen

A step-by-step introduction to the major features and concepts behind the Nitrogen Web Framework.

http://nitrogenproject.com/doc/tutorial.html

Agenda

Part 1: Install & Run Nitrogen

Part 2: Nitrogen Pages

Part 3: Nitrogen Elements

Part 4: Nitrogen Actions

Part 5: Nitrogen Postback Events

Part 6: Session and Page State

Part 7: Security

Part 8: Validation

Part 9: Comet

Part 10: Extending Nitrogen

Conclusion

## Part 1: Install & Run Nitrogen

### Install Nitrogen

If you don't have Erlang Installed:

Download Nitrogen (http://nitrogenproject.com/downloads), unzip and cd nitrogen.

If you do have Erlang installed:

Pull the Nitrogen Source Code(http://github.com/nitrogen/nitrogen), then make rel\_inets; cd rel/nitrogen.

### Run the Website

Start Up

bin/nitrogen console

Open http://localhost:8000 in your Browser

Shut Down

Press Control-C twice.

### A Tour Through the Files

View the Directory

ls -l

Anatomy of a Nitrogen Project

BuildInfo.txt

From uname.

Makefile

Used by make.

bin/

Commands to start and stop system, plus developer tools.

etc/

Configuration settings.

site/

Contains the website files, templates, and Erlang modules.

log/

The logs.

erts-X.Y.Z/

Embedded Erlang.

releases/

Tells Erlang how to start the system.

lib/

Dependent libraries.

The site/ Directory

**The site directory should go under source control, it contains all of the information necessary to run the website.**

Emakefile

Used by make.erl to compile the system.

~/erltoolkit/nitrogen/myapp$ find . -name "Emakefile"

./lib/sync/Emakefile

~/erltoolkit/nitrogen/myapp/lib/sync$ cat Emakefile

{

[

"./src/\*",

"./src/\*/\*",

"./src/\*/\*/\*"

],

[

{ i, "./include" },

{ outdir, "./ebin" },

debug\_info

]

}.

ebin/

Compiled Erlang modules.

include/

Include files for your website.

src/

Erlang source files for your website.

static/

Static files for your website.

templates/

Template files for your website.

The site/src/ Directory

Stores the Erlang source files for your application. By default it contains:

nitrogen\_init.erl

Runs once on Nitrogen startup.

nitrogen\_PLATFORM.erl

Holds the request loop depending on platform.

index.erl

The default web page.

elements/

By convention, custom elements are placed here.

actions/

By convention, custom actions are placed here.

### Exercise:

**Modify Your First Page**

From the Erlang Shell, run:

sync:go()

erltoolkit/nitrogen/myapp

Eshell V6.2 (abort with ^G)

(myapp@127.0.0.1)1> No configuration for inets httpd, so constructing from simple\_bridge.config...

(myapp@127.0.0.1)1> Starting Inets Server at "0.0.0.0":8000

(myapp@127.0.0.1)1> Static Paths: ["js/","css/","images/","nitrogen/","favicon.ico"]

Document Root for Static: ./site/static

(myapp@127.0.0.1)1> **sync:go().**

Starting Sync (Automatic Code Compiler / Reloader)

Scanning source files...

ok

(myapp@127.0.0.1)2>

Open **site/src/index.erl**

Change "Welcome to Nitrogen" to "Welcome to My Website"

Reload the page

**Compile in a Different Way**

From a different terminal, run:

**bin/dev compile**

Change to "Welcome to my ERL-TASTIC WEBSITE!" (or, you know, whatever)

Reload the page

**Understanding sync**

- Running sync:go() from the Erlang shell or bin/dev compile start the sync application

- Sync applications constantly checks for changes to Erlang files and attempts to recompile

- To stop sync's checking, run sync:stop()

- Note: Sync will only recompile files changed since sync was launched. Sync is not aware of changes made before running sync:go()

**Debug Statements**

- Add ?DEBUG to index.erl. Then compile and reload. What happens?

- Add ?PRINT(node()) to index.erl. Then compile and reload. What happens?

**Emacs nitrogen-mode**

(add-to-list 'load-path "PATH/TO/NITROGEN/support/nitrogen-mode")

(require 'nitrogen-mode)

Without nitrogen-mode:

#panel { id=my\_panel, body=[

#panel { id=my\_panel2, body=[

#label { text="Name" },

#textbox { id=my\_textbox }

]}

]}

With nitrogen-mode:

M-x nitrogen-mode

#panel { id=my\_panel, body=[

#panel { id=my\_panel2, body=[

#label { text="Name" },

#textbox { id=my\_textbox }

]}

]}

## Part 2: Nitrogen Pages

### What is a Nitrogen Page?

A Page is **an Erlang Module**

Each page should accomplish one store or piece of functionality.

Some examples:

Allow the user to log in (user\_login.erl).

Change the user's preferences. (user\_preferences.erl)

Display a list of items. (items\_view.erl)

Allow the user to edit an item. (items\_edit.erl)

### Dynamic Routing Explained

**Dynamic routing rules:**

1. If there is an extension, assume a static file.

**http://localhost:8000**/routes/to/a/module

http://localhost:8000/routes/to/a/static/file.html

2. Root page maps to index.erl

3. Replaces slashes with underscores.

http://localhost:8000/routes/to/a/module ->

routes\_to\_a\_module.erl

4. Try the longest matching module.

http://localhost:8000/routes/to/a/module/foo/bar ->

routes\_to\_a\_module.erl

5. Modules that aren't found go to web\_404.erl if it exists.

6. Static files that aren't found are handled by the underlying platform (not yet generalized.)

### Creating Your First Page

Exercise: Create a New Page

Generate the Page

**bin/dev page my\_page**

~/erltoolkit/nitrogen/myapp/bin$ ./dev page my\_page

Created page: ./site/src/my\_page.erl

Remember to recompile!

$EDIT site/src/my\_page.erl

Replace the default body with:

body() -> "Hello World!".

Remove the event/1 function.

**Compile the page** and load http://localhost:8000/my/page

~~~/erltoolkit/nitrogen/myapp$~~ **~~rebar compile~~**

~~==> mimetypes (compile)~~

~~==> simple\_bridge (compile)~~

~~Generating crypto compatibility for simple\_bridge...~~

~~...Using: "crypto:hash(sha, Data)"~~

~~...no changes needed to "include/crypto\_compat.hrl". Skipping writing new file~~

~~==> nprocreg (compile)~~

~~==> nitrogen\_core (compile)~~

~~Generating crypto compatibility...~~

~~...Using: "crypto:block\_encrypt(aes\_cbc128, Key, IV, Data)"~~

~~...Using: "crypto:block\_decrypt(aes\_cbc128, Key, IV, Data)"~~

~~...Using: "crypto:hash(sha, Data)"~~

~~...no changes needed to "include/crypto\_compat.hrl". Skipping writing new file~~

~~==> sync (compile)~~

~~==> site (compile)~~

~~==> myapp (compile)~~

**bin/dev compile**

~/erltoolkit/nitrogen/myapp/bin$ ./dev compile page my\_page.erl

ERROR: Unknown command: ["compile","page","my\_page.erl"]

---------------------------------------------------------

**Nitrogen Developer Utility**

---------------------------------------------------------

./bin/dev compile

- Recompile changed files on a running Nitrogen system.

./bin/dev page <name>

- Create a new page with the specified name.

./bin/dev action <name>

- Create a new action with the specified name.

./bin/dev element <name>

- Create a new element with the specified name.

./bin/dev help

- This screen.

~/erltoolkit/nitrogen/myapp/bin$ **./dev compile**

Scanning source files...

~/erltoolkit/nitrogen/myapp/bin$ ./dev page my\_page

File already exists: ./site/src/my\_page.erl!

Reload load in the web browser:

http://localhost:8000/my/page

According to Dynamic routing rules(3. Replaces slashes with underscores: http://localhost:8000/routes/to/a/module ->routes\_to\_a\_module.erl):

<http://localhost:8000/my/page> -> <http://localhost:8000/my_page.erl>

### How is a Page Rendered?

1. User hits a URL.

2. **URL is mapped to a module.**

3. **Nitrogen framework calls module:main()**

4. module:main() calls a #template

5. #template calls back into the page (or other modules)

6. Nitrogen framework renders the output into HTML/Javascript.

7. (This is the simple version. Complex version will come later.)

### Anatomy of a Template

- HTML. **The Page is sliped into the Template**.

- Contains one or more callouts, ie:

[[[module:body()]]]

- Contains a script callout for Javascript:

[[[script]]]

- The callouts look like Erlang, but they are not. They can only be of the form module:function(Args). The 'page' module refers to the current page.

### Experimenting With Templates

- Change the callout from page:body() to page:body1() in the default template and reload the page. What happens?

- Create another callout. What happens?

- What happens when you change page to be a specific module?

- Replace the module call with some arbitrary Erlang code. What happens?

## Part 3: Nitrogen Elements

### What is a Nitrogen Element?

An element can be either HTML, or **some record that renders into HTML.**

Change this:

body() -> "Hello World!".

To this:

body() -> #label { text="Hello World!" }.

The #label{} element is rendered into:

<label class="wfid\_tempNNNNN label">Hello World!</label>

View the rendered page source in your browser and search for "Hello World".

### Why Nitrogen Elements?

Nitrogen elements serve two purposes:

1. Allow you to generate HTML within Erlang:

- Avoid mixing languages == clearer code.

- Fewer characters to type.

- Checked at compile time.

2. Abstraction layer:

- Avoid repeating common functionality.

- Hide complexity in a module.

### Add Elements to Your Page

Nitrogen Element Examples

Try this on my\_page.erl:

body() ->

[

#h1 { text="My Simple Application" },

#label { text="What is your name?" },

#textbox { },

#button { text="Submit" }

].

Then compile, reload, and view source.

### Nested Elements

Try a nested element:

body() ->

#panel { style="margin: 50px;", body=[

#h1 { text="My Page" },

#label { text="Enter Your Name:" },

#textbox { },

#button { text="Submit" }

]}.

### Documentation

### Anatomy of a Nitrogen Element

## Part 4: Nitrogen Actions

### What is a Nitrogen Action?

**An action can either be Javascript, or some record that renders into Javascript.**

Add a Javascript alert to the #button{} element. Then recompile and run. What do you expect will happen?

body() ->

[

#button{

text="Submit",

actions=[#event{type=click,actions="alert('hello');" }]

}

].

Do the same thing a different way.

body() ->

[

#button{

text="Submit",

actions=[#event{type=click, actions=#alert{text="Hello"}]

}

].

### Wiring an Action

Setting the actions property of an element can lead to messy code. Another, cleaner way to wire an action is the wf:wire/N function.

body() ->

wf:wire(mybutton, #effect { effect=pulsate }),

[

#button { id=mybutton, text="Submit" }

].

### Conditional Actions with #event{}

Put the #effect{} action inside of an #event{} action. This causes the effect to only get fired if the user clicks on mybutton.

body() ->

wf:wire(

mybutton,

#event{

type=click,

actions=#effect { effect=pulsate}}),

[

#button { id=mybutton, text="Submit" }

].

### Postbacks

[postback, n.回发]

http://en.wikipedia.org/wiki/Postback

In eCommerce

In the context of eCommerce, the term is used to describe a sales transaction notification from payment processors to the merchant's affiliate system site. More specifically, it is a web service written for an affiliate sales tracking software system for a third-party merchant system to send or "POST" the data to. The term "Postback" is used here to describe what the payment processor does with the transaction receipt, which is to "Post Back" to the merchant's affiliate program, notifying it of a successful transaction, so that it can then credit affiliates with their earnings.

In web development

In the context of web development, a postback is an HTTP POST to the same page that the form is on. In other words, the contents of the form are POSTed back to the same URL as the form.[1]

Postbacks are commonly seen in edit forms, where the user introduces information in a form and hits "save" or "submit", causing a postback. The server then refreshes the same page using the information it has just received.

Postbacks are most commonly discussed in relation to JSF and ASP or ASP.NET.

In ASP, a form and its POST action have to be created as two separate pages, resulting in the need for an intermediate page and a redirect if one simply wants to perform a postback. This problem was addressed in ASP.NET with the \_\_doPostBack() function and an application model that allows a page to perform validation and processing on its own form data.

In JSF, postbacks trigger the full JSF life-cycle, which just like ASP.NET performs conversion and validation of the form data that was included in the postback. Various utility methods are present in the JSF API to programmatically check if a given request is a postback or not.

## Part 5: Nitrogen Postback Events

## Part 6: Session and Page State

## Part 7: Security

## Part 8: Validation

## Part 9: Comet

## Part 10: Extending Nitrogen

## Conclusion

# Embedding the Nitrogen Web Framework into an Existing Erlang Application

http://sigma-star.com/blog/post/embedding-nitrogen

by jesse

2013-08-07

Up until very recently, adding Nitrogen to an existing application was an exercise in frustration, even for Erlang veterans. This is no longer the case, adding Nitrogen to an existing application is now a snap with the new embed helper script.

Originally, If you were starting from scratch and making a Nitrogen-focused app, it was simple, just run make rel\_XXX or make slim\_XXX) (where XXX is the backend of choice). But if you had an already existing application and you wanted to add a web front-end or web interface and use Nitrogen, then you had to follow [a bunch of steps](http://rshestakov.wordpress.com/2012/12/30/how-to-install-nitrogen/), copying files from one repository, changing paths in files, adding dependencies - in short, it was a pain.

## Enter embed

The new embed script was just added to the root of what will soon be Nitrogen 2.2.0.

This script will do all the heavy lifting for you, asking your preferences, automatically merging dependencies, adding Makefile rules, and copying the necessary files for the backend of your choice.

With this script, the only manual changes you'll have to are to your launch script: Add the necessary code paths, config file loading, and either making the call to nitrogen\_sup:start\_link(), or simply including that into your supervisor tree.

## Ninja-quick version

1. Clone nitrogen
2. Navigate to the root of the application you wish to add Nitrogen to
3. Run /path/to/nitrogen/embed
4. Follow on-screen instructions`
5. Modify your launch script to include -pa deps/nitrogen\_core/ebin, -config path/to/file.config for each config file, and a call to -eval "nitrogen\_sup:start\_link()" (or add it to your supervision tree)
6. make (if your makefile does everything necessary), or rebar get-deps followed by rebar compile.
7. Run application
8. Navigate browser to http://127.0.0.1:8000

## Detailed Demonstration

For a simple demonstration, let's do something completely contrived. Let's add a web interface to [qdate](https://github.com/choptastic/qdate), the date and timezone utility.

First, we clone qdate and Nitrogen.

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2 | $ git clone git://github.com/choptastic/qdate.git  $ git clone git://github.com/nitrogen/nitrogen.git |

Now we enter the qdate directory, and to be safe, let's make a 'web-interface' branch

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2 | cd qdate  $ git checkout -b web-interface |

Now we make the call to embed. Currently, we must make sure to make the call from the directory we wish to install to. So we call it like this (I'm just including the pwd call to demonstrate our current location).

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3 | $ pwd  /home/user/qdate  $ ../nitrogen/embed |

Which will result in you seeing this:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*\*\*\* NITROGEN WEB FRAMEWORK EMBEDDER          \*\*\*\*  \*\*\*\*                                                      \*\*\*\*  \*\*\*\* Adding Nitrogen to an existing Erlang Application  \*\*\*\*  \*\*\*\*                                                      \*\*\*\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    NOTE: BEFORE PROCEEDING IT'S HIGHLY RECOMMENDED THAT YOU MAKE     NEW BRANCH IN YOUR SOURCE CODE MANAGEMENT SYSTEM, OR AT     LEAST MAKE A BACKUP OF YOUR DIRECTORY, AS THIS WILL MAKE     CHANGES TO THE CURRENT DIRECTORY TREE BY ADDING AND     MODIFYING FILES.      Will be installing from '/home/user/nitrogen' into '/home/user/qdate'    Web server to use?  Options are (c)owboy, (i)nets, (m)ochiweb, (w)ebmachine, (y)aws.  Please choose (i/m/c/w/y): |

Here you enter your choice: c, i, m, w, or y For our demonstrating we'll choose cowboy, so we type c and press enter

Now we are presented with a series of path questions, with sensible defaults provided. Pressing "Enter" will stick with the default, or you can type the updated path (paths will be relative to the current working directory).

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5 | Where to put the Erlang code for Nitrogen pages and initialization [Default: src/nitrogen/]:  Where to put the Erlang headers [Default: include/]:  Where to put the static directory (for js, css, etc) [Default: priv/static/]:  Where to put the templates directory [Default: priv/templates/]:  Where to put the config files [Default: etc/]: |

After the paths are provided, you're presented with a series of yes/no questions:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5 | Install the plugin scripts and configs? (y/n): y  Add Nitrogen dependencies to rebar.config automatically? (nitrogen\_core, simple\_bridge, nprocreg, sync, cowboy, and any of cowboy's dependencies. (y/n): y  Add a 'make plugins' rule to the Makefile? (y/n): y  Add a 'make copy-static' rule to the Makefile? (y/n): y  Run 'make' after installation? (y/n): y |

Note: Unless you want to do something manually, it's recommended to choose "y" for all provided questions.

Finally, we get down to it. The embed script presents us with our chosen preferences, and a final "Are you sure you want to proceed?" question.

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*\*\* We're almost ready to go. Please review:  \*\*\* Erlang Source: src/nitrogen/  \*\*\* Erlang Headers: include/  \*\*\* Static Resources: priv/static/  \*\*\* Nitrogen Templates: priv/templates/  \*\*\* Config Files: etc/  \*\*\* Install Plugin Script: Yes  \*\*\* Add 'make plugins' rule to Makefile: Yes  \*\*\* Add 'make copy-static' rule to Makefile: Yes  \*\*\* Add dependencies to rebar.config: Yes  \*\*\* Run 'make' after all is complete: Yes  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Are you sure you want to proceed? (y/n): |

Entering y and pressing Enter will begin the installation process.

After fast-scrolling informative text (telling you what the script is doing, followed, if you specified it, by the running of make), you'll be presented with the following summary with some instructions:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*\*\*\*                          Installation Complete                        \*\*\*\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  There are a few manual steps you must take before you're completely set up:    1) You must make sure to start Nitrogen.  The easiest way is by      adding `nitrogen\_sup:start\_link()` to your application.    2) You must make sure that the config files in etc/ are properly     loaded. This can be done with a vm.args file if you're using a reltool     generated release, or it can be done by adding additional -config calls to     the commandline call that launches your app.    3) You may need to add the new packages (nitrogen\_core, simple\_bridge,     nprocreg, etc) to your app's code path (the easy way is with the -pa flag     in the `erl` call.)      Congratulations on adding Nitrogen to your Application |

As it says, we're almost there. Now we just need to make some changes to the launch script:

Let's edit Makefile and make a change. Currently, make run is the quick way to launch a qdate terminal, so let's edit that run rule in the Makefile.

Original:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2 | run:      erl -pa ebin/ deps/\*/ebin/ -eval "application:start(qdate)" |

Let's load some configs, and add a call to start nitrogen:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2 | run:     erl -pa ebin/ deps/\*/ebin/ -config etc/cowboy.config -config etc/app.config -eval "application:start(qdate)" -eval "nitrogen\_sup:start\_link()" |

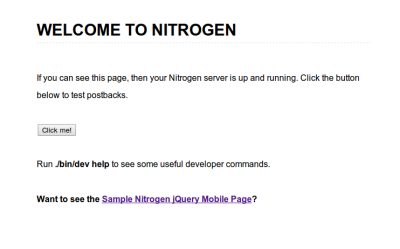
(Note: we didn't need to modify the path, since -pa deps/\*/ebin was already in there, which covers all necessary dependency paths for us.)

Finally, let's launch qdate and see if we get the default Nitrogen page:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | $ make run  erl -pa ebin/ deps/\*/ebin/ -config etc/cowboy.config -config etc/app.config -eval "application:start(qdate)" -eval "nitrogen\_sup:start\_link()"  Erlang R16B (erts-5.10.1) [source] [64-bit] [smp:4:4] [async-threads:10] [hipe] [kernel-poll:false]    Eshell V5.10.1  (abort with ^G)  1> Starting Cowboy Server (nitrogen) on 0.0.0.0:8000, root: './priv/static' |

Perfect! Let's see what happens when we navigate our browser to http://127.0.0.1:8000



Great! It's also loading the default "Welcome to Nitrogen" page.

### Let's put it all together now

The last step is to simply add a web interface for qdate.

So let's delete the unnecessary 'mobile.erl' pages from src/nitrogen (our sensible default) and we'll just make a simple change to index.erl.

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

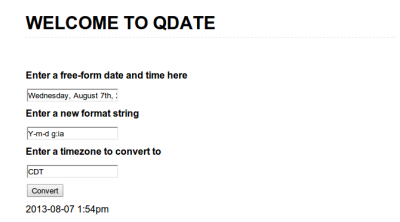
|  |  |
| --- | --- |
| 1  2 | cd src/nitrogen  rm index.erl |

Let's change it to this:

[?](http://sigma-star.com/blog/post/embedding-nitrogen)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | -module (index).  -compile(export\_all).  -include\_lib("nitrogen\_core/include/wf.hrl").    main() -> #template { file="./priv/templates/bare.html" }.    title() -> "Welcome to qdate".    body() ->      #container\_12 { body=[          #grid\_8 { alpha=true, prefix=2, suffix=2, omega=true, body=inner\_body() }      ]}.    inner\_body() ->      [          #h1 { text="Welcome to qdate" },                  #label{text="Enter a free-form date and time here"},                  #textbox{id=source\_date, text=qdate:to\_string("l, F jS, Y g:i A T")},                    #label{text="Enter a new format string"},                  #textbox{id=format\_string, text="Y-m-d g:ia"},                    #label{text="Enter a timezone to convert to"},                  #textbox{id=timezone, text="America/Chicago"},                    #br{},                    #button{text="Convert", postback=convert},                  #panel{text="", id=result}      ].    event(convert) ->          [From, String, TZ] = wf:mq([source\_date, format\_string, timezone]),          wf:update(result, qdate:to\_string(String, TZ, From)). |

And recompile (we can enabled sync, if you so desire, or just kill the server and run make). After compiling and loading the modules, we get this:



We can enter a source date, a new format, and a timezone, and it'll do the conversion for us using qdate.

## Conclusion

While this example is pretty contrived, it gives us an idea of just how much easier it is to add Nitrogen to an application now without having to jump through a bunch of hoops, or a series of tubes, if you will.

# API

# Elements

# Actions

# Validators

# Handlers

# Configuration Options

# Plugins

# Mobile

# Troubleshooting

# References

## github-nitrogen

https://github.com/nitrogen

https://github.com/nitrogen/nitrogen

## nitrogenproject.com

<http://nitrogenproject.com/>

# ----------------END----------------