

Merb Slices

Slice By Who?

- Fabien Franzen a.k.a. loob2
- Actually remembers Merb 0.0.0.1 a.k.a 'the pastie'
- Representing Belgium (or maybe not...)

Who Am I?

- Daniel Neighman a.k.a. hassox
- Engine Yarder
- Proud Aussie



What's a Merb Slice?



What's a Merb Slice?

- Full Stack Mini Merb Application
- Designed to be shared between applications
- Built on Merbs Public API



What's Included?

- Full MVC Components
- Assets
- Routes
- Full Namespacing
- Hot Code Loading / Unloading
- Distribution as Gems



Slice It

Slice It

- Get it:
 - `$ sudo gem install blog_slice`

Slice It

- Get it:

→ `$ sudo gem install blog_slice`

- Install it:

→ `$ rake slices:blog_slice:install`

Configure It

Configure It

config/init.rb

```
dependency "merb-slices"  
dependency "blog_slice"
```

Configure It

config/init.rb

```
dependency "merb-slices"  
dependency "blog_slice"
```

config/routes.rb

```
Merb::Router.prepare do  
  all_slices  
end
```

Run It

Run It

\$ merb

Use It

Use It

- By Default, your slice is available at:
 - http://localhost:4000/blog_slice

Use It

- By Default, your slice is available at:
 - http://localhost:4000/blog_slice
- Get the posts:
 - http://localhost:4000/blog_slice/posts

Rake It

Rake It

There are many rake tasks available

Rake It

There are many rake tasks available

Rake It

There are many rake tasks available

- rake slices

Rake It

There are many rake tasks available

- rake slices
- rake slices:install_as_gem

Rake It

There are many rake tasks available

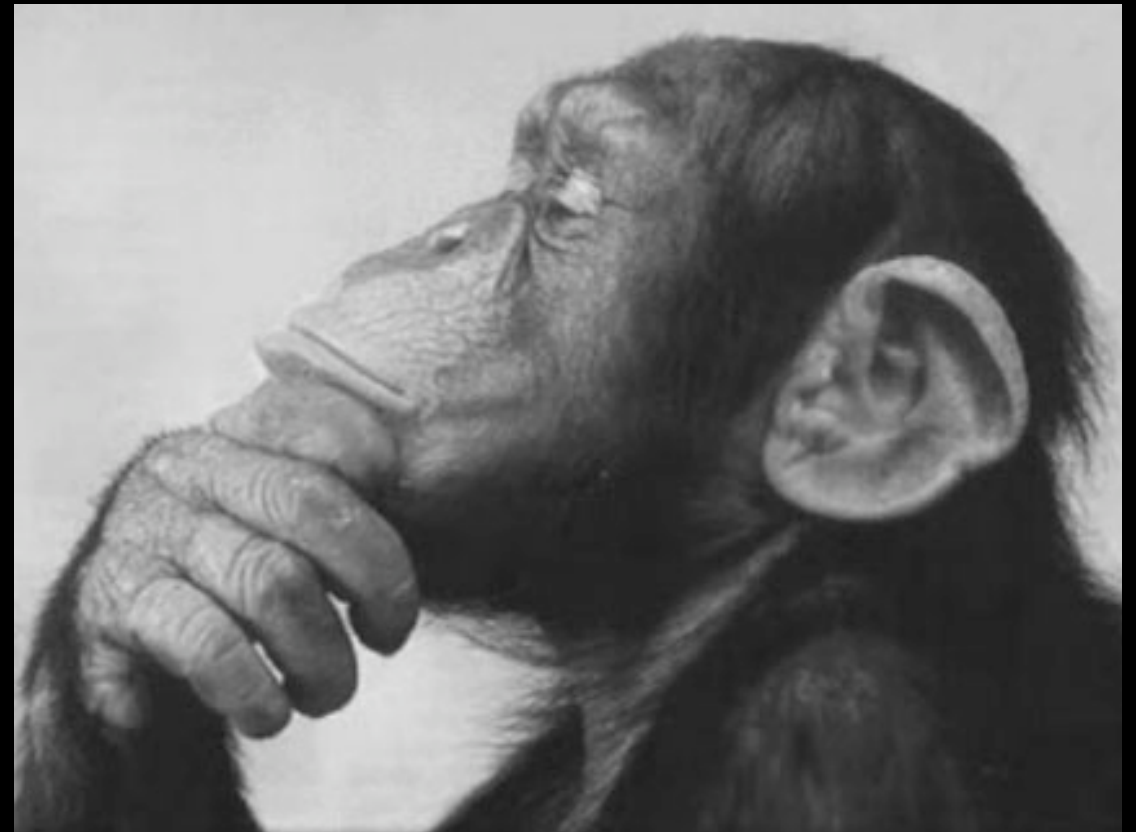
- `rake slices`
- `rake slices:install_as_gem`
- `rake -T slices:blog_slice`

Common Rake Tasks

- `rake slices:blog_slice:copy_assets`
- `rake slices:blog_slice:patch`
- `rake slices:blog_slice:freeze(:*)`
- `rake slices:blog_slice:migrate`
- `rake slices:blog_slice:spec`

Customize It

- It's just ruby. Monkey Patch it
- Use the “patch” rake task
- Slice code is in your app:
 - `Merb.root/slices/blog_slice`



Custom Options

Custom Options

- Each Slice has a Configuration Hash
- `Merb::Slices::config[:blog_slice]`

Custom Options

- Each Slice has a Configuration Hash
- `Merb::Slices::config[:blog_slice]`
- Aliased To:
- `BlogSlice[]`

Custom Actions

Custom Actions

Adding the Action to the controller

```
$ rake slices:blog_slice:patch
```

Custom Actions

Adding the Action to the controller

```
$ rake slices:blog_slice:patch
```

Edit: Merb.root/blog_slices/app/controllers/posts.rb

```
def publish
  @post = Post.first(:permalink => params[:permalink])
  if @post.publish!
    redirect slice_url(:post, @post), :message => "Posts Published"
  else
    render :edit
  end
end
```

Custom Views

Custom Views

```
$ rake slices:blog_slice:freeze:views
```

Custom Views

```
$ rake slices:blog_slice:freeze:views
```

Available in:

```
Merb.root/slices/blog_slice/app/views
```

Custom Views

```
$ rake slices:blog_slice:freeze:views
```

Available in:

```
Merb.root/slices/blog_slice/app/views
```

Add templates for additional formats

Custom Layout

Custom Layout

- By Default the Slice layout is used

Custom Layout

- By Default the Slice layout is used

```
Merb::BootLoader.after_app_loads do  
  BlogSlice[:layout] = :blog_slice_layout  
end
```

Custom Models

Custom Models

Stubs or Freeze?

```
$ rake slices:blog_slice:freeze:models
```


Custom Models

Stubs or Freeze?

\$ rake slices:blog_slice:freeze:models

```
module BlogSlice
  class Post

    property :published, Boolean
    property :published_at, DateTime

    def publish!
      self.published = true
      self.published_at = DateTime.now
      save
    end

  end # Post
end # BlogSlice
```

Custom Routes

```
Merb::Router.prepare do

  add_slice(:blog_slice, :path_prefix => "blog", :default_routes => nil) do
    identify BlogSlice::Post => :permalink do
      match("/posts/:permalink/publish").
        to(:controller => "posts", :action => "publish").
        name(:publish)
    end
  end
end

end
```

Prefix URL Paths

Prefix URL Paths

- Setup a Path Prefix
- `add_slice(:blog_slice, "awesome")`
- Example: `/awesome/posts`

Prefix Named Routes

Prefix Named Routes

- Default `name_prefix` on named routes:
 - `add_slice(:blog_slice)`
 - `url(:blog_slice_posts)`

Prefix Named Routes

- Default name_prefix on named routes:
 - `add_slice(:blog_slice)`
 - `url(:blog_slice_posts)`
- Setup a Name Prefix
 - `add_slice(:blog_slice, :name_prefix => "blog")`
 - `url(:blog_posts)`

Write It

Development Flow

Development Flow

- Generate
- Write
- Use
- Write
- Use
- Finish - Install

Development Flow

- Generate
- Write
- Use
- Write
- Use
- Finish - Install

\$ slice

Generate It

```
$ merb-gen slice blog_slice
```

New Slice Structure

blog_slice

|-app

|---controllers

|---helpers

|---models

|---views

|-----layout

|-----main

|-lib

|---blog_slice

|-pkg

blog_slice

|-public

|---images

|---javascripts

|---stylesheets

|-spec

|---controllers

|-stubs

|---app

|-----controllers

|-----models

Some Important Files

blog_slice/lib

blog_slice

merbtasks.rb

slicetasks.rb < Here be dragons

spectasks.rb

blog_slice.rb

Initialize It

Initialize It

- `blog_slice.rb` is where the slice initializes

Initialize It

- `blog_slice.rb` is where the slice initializes
 - Dependencies

Initialize It

- `blog_slice.rb` is where the slice initializes
 - Dependencies
 - Router

Initialize It

- `blog_slice.rb` is where the slice initializes
 - Dependencies
 - Router
 - Hooks

Fake It

- Setup a fake “host app” env in config/init.rb
- config/init.rb is not used normally

Controllers

Controllers

- `blog_slice/app/controllers/posts.rb`

```
class BlogSlice::Posts < BlogSlice::Application  
  # Your Controller Code Here  
end # Posts
```

Views

- `blog_slice/app/views/posts/show.html.haml`
- `blog_slice/app/views/layouts/application.html.haml`

Layouts

- `blog_slice/lib/blog_slice.rb`

```
Merb::Slices::config[:blog_slice][:layout] ||= :blog_slice
```


slice_url

slice_url

- Use slice_url for url generation inside a slice
- slice_url(:controller => ..., :action => ...)
- slice_url(:post, @post)
- slice_url(:merb_auth_slice_password, :login)

Models

Namespace your models (you don't have to)

```
class BlogSlice::Post
  include DataMapper::Resource

  property :id,          Serial
  property :title,       String, :length => 255
  property :body,        Text
  property :slug,        Slug

  before :save do
    self.slug = self.title unless self.permalink
  end
end # BlogSlice::Post
```

Assets

Assets

- `blog_slice/public`
 - `images, css, javascript`
- Read:
`blog_slice/app/helpers/application_helper.rb`

Images

- `blog_slice/public/images`
- Helper: `image_path(image)`

Javascript

- `blog_slice/public/javascripts`
- Helper: `javascript_path(javascript)`

Stylesheets

- `blog_slice/public/stylesheets`
- Helper: `stylesheet_path(style)`

Install Assets

- `rake slices:blog_slice:copy_assets`

Route It

blog_slice/lib/blog_slice.rb (the init.rb)

```
def self.setup_router(scope)
  scope.identify Post => :slug do
    resource :posts
  end
end
```

Hook It

Hook It

- loaded

Hook It

- loaded
- init

Hook It

- loaded
- init
- activate

Hook It

- loaded
- init
- activate
- deactivate

Hooks - loaded

- Slice Code Loads
- Hook - loaded
- Boot Loader LoadClasses

Hooks - init, activate

- Slice Hook - init
- BootLoader AfterAppLoads
- Slice Hook - activate: triggered by
`Merb::Slices.activate(BlogSlice)`

Hooks - deactivate

- Triggered by
`Merb::Slices.deactivate(BlogSlice)`

Spec It

- Specs go in `blog_slice/spec`
- `rake -T spec`
- Spec your slice like an app

Spec It - Setup

- Setup the routes:

```
before :all do
  Merb::Router.prepare do
    add_slice(:blog_slice)
  end if standalone?
end
```

Distribute It

- rake gemspec
- rake install

Questions?