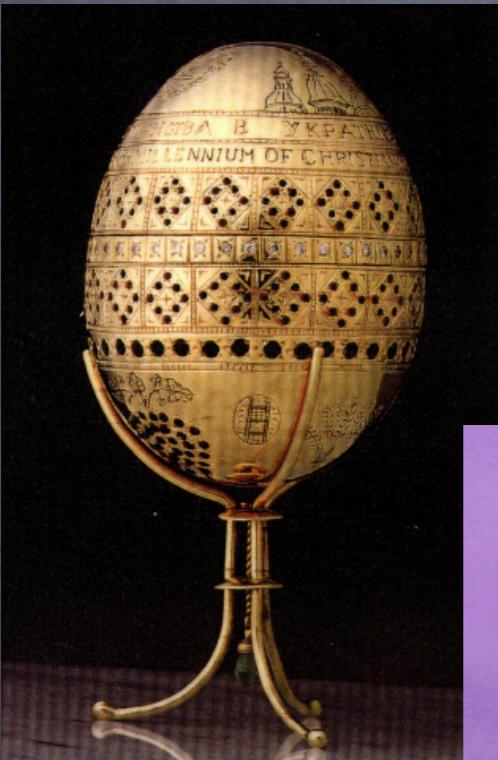


# MerbAuth

## Darwinian Authentication



# In The Beginning

- ⦿ Conditions for Authentication were ripe
- ⦿ Merb
  - ⦿ Plugins
  - ⦿ Rubigen with scopes (6th Nov 2007)
- ⦿ Restful Authentication was missing



merbful\_authentication

# merbful\_authentication

- ⦿ Released 4th January 2008
- ⦿ Direct port of Restful Authentication
- ⦿ Supported DataMapper & ActiveRecord
- ⦿ Supported RSpec & Test::Unit

# Pros

- First plugin with multi-ORM support in Merb
- A lot of applications used it

# Cons

- ⦿ Generated code
- ⦿ Very complex
- ⦿ Maintenance sucked

# The Catalyst

- Slices were born



Original merb-auth

# merb-auth

- ⦿ Mostly direct port of merbful\_authentication
- ⦿ Used brand new slices plugin
- ⦿ Moved to Library Code not Generated
- ⦿ ORM support via mixins
- ⦿ Forgotten Passwords
- ⦿ 17 June 2008

# merb-auth - Pros

- ⦿ Live code. Not generated
- ⦿ Minimal Application Configuration
- ⦿ Implemented as a slice
- ⦿ Easier to maintain (still sucked)

# merb-auth - Cons

- ⌚ User model hidden
- ⌚ Hard to please all through configuration
- ⌚ Unclear how to customize it
- ⌚ Tied to one model type
- ⌚ Dictates user model
- ⌚ Extensions difficult (No OpenID)
- ⌚ Difficult to change logic

Evolutionary Step  
Required

# The Catalyst

- ⦿ Adam French proposed:
  - ⦿ Authenticating Sessions
  - ⦿ Simple session based api
  - ⦿ Using Exceptions to refuse entry
  - ⦿ Provides correct status code

# ExceptionalAuthentication

- ⦿ Adam created a prototype
  - ⦿ ExceptionalAuthentication
- ⦿ Application including his proposals

# Session API

- ⌚ session.authenticated?
- ⌚ session.authenticate!
- ⌚ session.user
- ⌚ session.user=
- ⌚ session.abandon!

# Exceptional Authentication

- Originally a DataMapper based system
- Decided to allow arbitrary “user” objects



Code Named - Mauth

# MerbAuth - What is it?

- Authentication Framework
- Cascading Strategy Concept
- Only what's needed to support authentication
- Supports user objects such as
  - DM, AR, Sequel, Hash, String, File, IO, or just plain old Object

# MerbAuth - What it's Not

- ➊ A user management system

# Default Merb Stack

- Merb stack includes MerbAuth with:
  - Password Slice
  - Password based form authentication
  - Basic Authentication
- Salted Users
- Routes / Forms

# Activating Merb Stack

- ⌚ Comes with a Basic User model (Change it up)
- ⌚ `rake db:automigrate`
- ⌚ Make something to protect

# Protect It

- ⦿ Route Level

```
protect do  
  resources :paychecks  
end
```

- ⦿ Controller Level

```
before :ensure_authenticated
```

# Activating Merb Stack

- Add a user to the system
- Login “/login”

# Protect - Router Based

```
protect(BasicAuth) do
  resources :api
```

```
  protect do
    resources :posts do
      resources :comments
    end
  end
```

```
end
```

# Controller Based

- ⌚ :ensure\_authenticated

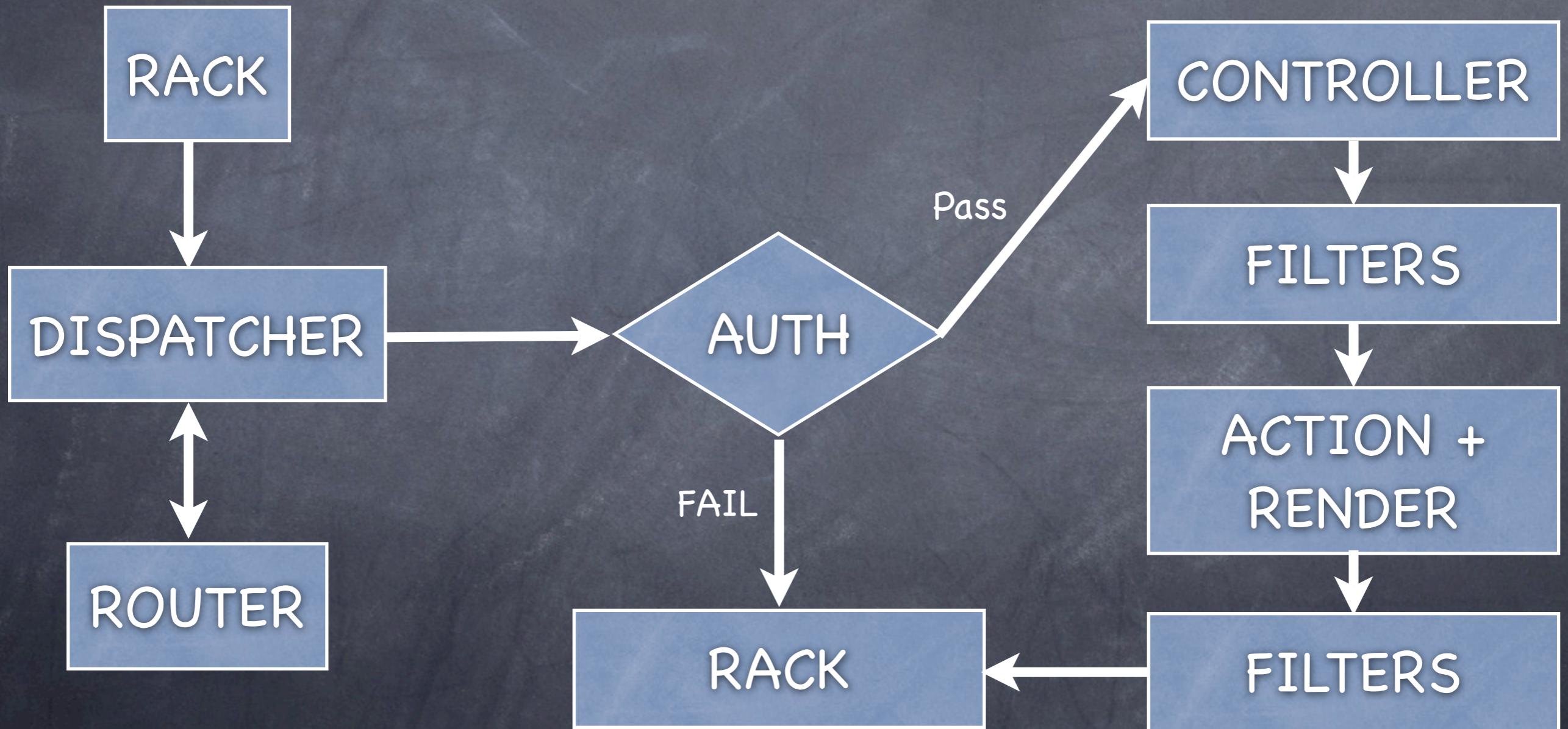
```
before :ensure_authenticated
```

# OR

```
before :ensure_authenticated, :with => [OpenID, BasicAuth]
```

# Why Protect Routes?

# Why Protect Routes?



# What is a Strategy?

- Strategy contains logic for “Authentication”
- Implements a #run! method

```
class PasswordStrategy < Merb::Authentication::Strategy
  def run!
    login      = request.params[:login]
    password   = request.params[:password]
    User.authenticate!(login, password) if login
  end
end
```

- Declare many. One for each login type

# MerbAuth

- Cascading Strategies
- Each strategy is run in order
- Success == First Strategy to return object
- Failure == No Strategies return object
- Stops trying when authenticated
- Re-order strategies with
  - `Merb::Authentication.default_strategy_order=`

# What Happens on Fail?

- ⌚ Raises Unauthenticated exception
- ⌚ Uses Merbs Exception Handling
  - ⌚ Exceptions#unauthenticated
- ⌚ Sets correct status code

# Display Errors

```
<%= error_messages_for session.authentication %>
```

# Customize Login Form

- Exceptions#unauthenticated (view)

# How to Logout?

- session.abandon!

# What Strategies Are There?

- ⦿ Password based form login
- ⦿ Open ID
- ⦿ Basic Authentication
- ⦿ Require a default strategy to load / define it
- ⦿ Monkey patch parts of it you need to change

# Mixins

- Use mixins to extend your User or MerbAuth
- Salted User mixin
- Redirect Mixin (`redirect_back_or`)

# Advanced Strategies

- ⦿ Loaded in order of declaration
- ⦿ Implement a #run! method
- ⦿ Use Inheritance to share code
- ⦿ Arbitrary Object Session Serialization
- ⦿ Return an object from #run! to authenticate

# Advanced Strategies

- ⦿ YourStrategy#redirect!
- ⦿ YourStrategy.abstract!
- ⦿ YourStrategy#user\_class
- ⦿ YourStrategy#headers
- ⦿ YourStrategy#status
- ⦿ request.params (request params)
- ⦿ params (route params)

# Failure Messages

- Inside a strategy
- `session.authentication.errors.add(:label, "message")`

# Overview of Advanced Setup

- ⦿ require 'merb-auth-core'
- ⦿ Define <User> model (Maybe)
- ⦿ Setup session storage
- ⦿ Declare strategies
- ⦿ Protect methods
- ⦿ Setup login / logout actions

# Where Next?

- ⦿ Implement slices
- ⦿ Implement Strategies & Mixins

# Resources

- [http://adam.speaksoutofturn.com/articles/  
authentication\\_vs\\_authorization.html](http://adam.speaksoutofturn.com/articles/authentication_vs_authorization.html)
- [http://github.com/wycats/merb-plugin/merb\\_auth](http://github.com/wycats/merb-plugin/merb_auth)
- <http://github.com/ck/cookbook>
- <http://github.com/RichGuk/merb-auth-example>

# Thanks

- Adam French
- Ben Burkett

