



Web Ready with Ruby

Will Mason

mybrightideas
Make Your Future Incredible



Outline



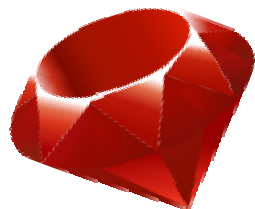
- Introduction
- What's it about?
- What's Ruby?
- Just Do It
 - Software Engineering?
- ramaze
 - netbeans
- Diversity, Potential & Futures



Silicon Valley Tarot

Scope and Purpose

- Ruby What!?
- Web Platforms
 - Camping
 - Rails
 - Ramaze
- Ruby Distinction
- Tools
- A Look at **ramaze**



What's It About?

- Flexibility
- Dynamic capability
- Orthogonality
 - D.R.Y. (Nope)
- Rapid, Leveraged, Behaviour Driven Code (BDD)
 - Bacon :: <http://github.com/chneukirchen/bacon>
- Building; Rather than 'development' (*opinion*)



Rails, Camping, Ramaze, ...?

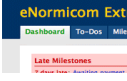







- **Camping**... <http://wiki.github.com/why/camping> 
 - Micro-framework (4k footprint)
- **Rails**... <http://www.rubyonrails.org> 
 - Web Development Framework
 - **Instiki** wiki... <http://instiki.org>
 - Using [Instiki As a CMS](http://instiki.org)
- **Ramaze**... <http://ramaze.net> 
 - Modular Web Development Platform

Camping sites











-  <http://Camping.rubyforge.org>
- **RedHanded** <http://redhanded.hobix.com>
-  <http://hackety.org>



sites

-  **Basecamp**
-  **Shopify**
-  **GitHub**
-  **StreetEasy**
-  **MTV Style**
-  **Twitter**
-  **FunOrDie**
-  **Jango**

ramaze sites

-  **Warmlake Swim** - Swimwear and equipment online.
 -  **JotBot** - Effortless time tracking.
 -  **Freezy** - Give / Get free stuff.
 -  **Selfmarks** - Anti-social bookmarks
 -  **Weez International Ltd.**
 -  **Ramaze by Example**
 -  **Rumors Facebook Application**
 -  **Weewar Players Association** wonko.com
 -  **STCA**, tactical combat
 -  **Thoth**
- <http://wiki.ramaze.net/SitesPoweredByRamaze>

What's Ruby?



- Gemstone: sapphire family
 - Ruby lore: *A ruby attracts your heart's desire. Often the ruby is lost once the desire is satisfied.*
- Object Oriented 3rd Generation Computer programming language
- Message passing model
 - Smalltalk, FORTH, Erlang, Ruby
 - Not message passing facility
- Every thing *is-a* first class object.

<http://www.ruby-lang.org>

Zero cross-talk

- There is Only Object (Simsript, Smalltalk)
- Least Surprise (FORTH)
- [Model-View-Controller](#)
- Message Passing (RSX, CTOS, RTOS, UDP or TCP)
- Mix-ins (partial class)
- Platforms: Java classes, .Net framework, Python/Ruby



Ruby Principles



- Don't Repeat Yourself (DRY)
- Message Passing
 - Everything “quacks”
- DUCK typing:
 - It walks like-a duck, quacks like-a duck = **DUCK**
 - Principle of Least Surprise (PLS)
- Lambda(λ) coding (and Blocks)
- Meta-programming (Smalltalk / FORTH)

<http://rubyforge.org>

Wither *Ruby*?



- Linux
- Solaris
- OS X
- Any Java platform
- Any platform with a C++ compiler
- Windows
 - .NET

<http://www.ruby-lang.org/downloads>

Just Do It



- “**Ruby Essentials**” ::
 - http://www.techotopia.com/index.php/Ruby_Essentials
- Ruby de web ::
 - <http://tryruby.hobix.com>
- Get a Start ::
 - **NetBeans** ... <http://www.netbeans.org>
 - <http://www.ruby-lang.org/en/documentation>
- “**Programming Ruby**”
 - <http://www.ruby-doc.org/docs/ProgrammingRuby>

show it



- Ruby de web ::
 - <http://tryruby.hobix.com> (Camping)
- Get a Start ::
 - **NetBeans** ... <http://www.netbeans.org>
- “**Programming Ruby**” (help file)
 - <http://www.ruby-doc.org/docs/ProgrammingRuby>

Looking Under The Hood **Ramaze**



- Behaviour driven
 - Why Wiki ... (60 lines)
 - *Hapricot* for web pages
- What's **not** modular?
 - Not ramaze
 - LEGO™ for apps.
- Rapid Web Framework
 - Camping clone (50 lines)
 - ramaze/examples/app/ **whyWiki**

```
spec_require 'bluecloth',
             'hapricot'
::
it 'should have main page' do
  t.body = page('/show/Home')
  t.should match(
    /^MicroWiki Home$/ )
  t.body.at('h1').inner_html.
    should == 'Home'
  t.body.at(
    'a[ @href=/edit/Home' ] ).
    inner_html.should
    == 'Create Home'
end
```



<http://ramaze.net>

‘empty’ prototype **Ramaze**



- c:\> ramaze create empty
 - Ready to roll prototype
 - Project name: “empty”
- What's what?
 - Controller ... Business logic
 - Layout Page design
 - Model Data model
 - Spec Behaviour specs
 - View Content

```
empty\
|... app.rb
|... config.ru
|... start.rb
+-- controller\
|   .... init.rb
|   .... main.rb
+-- layout\
+-- model\
+-- public\
|   +-- css\
|   +-- js\
+-- spec\
+-- view
```



<http://ramaze.net>

ramaze Tools

- Development & IDE
 - NetBeans, Visual Studio, Eclipse
 - Notepad, SciTE, etc.
- Ruby gems
 - C:\> `gem install ramaze`
- Build
 - Rake
 - Reap
- Test / Specification
 - rSpec ... Behaviour driven development
 - `require 'ramaze/spec/helper'`



<http://rubyforge.org>

Why Wiki

Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

Obligatory Demo ...

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki

Why Wiki

Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
require 'rubygems'
require 'ramaze'
require 'bluecloth'

Db=Ramaze::YAMLStoreCache.new(...)
  unless defined?(Db)

class WikiController
  < Ramaze::Controller
  def index
    redirect R(:show, 'Home')
  end
  def show ...
  def edit ...
  def save ...
  end
  Ramaze.start :adapter => :mongrel
```

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki

Why Wiki

Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
def show page = 'Home'
  @page = url_decode(page)
  @text = Db[page].to_s
  @edit_link = "/edit/#{page}"

  @text.gsub!(/\[\[([.]*?)\]\]/) do
    |m|
      exists = Db[$1] ? 'exists' :
        'nonexists'

      A( $1,
        :href => Rs(:show,
          url_encode($1)),
        :class => exists)
  end
  @text=BlueCloth.new(@text).to_html
end
```

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki



Why Wiki



Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
def edit page = 'Home'
  @page = url_decode(page)
  @text = Db[page]
end
```

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki



Why Wiki



Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
def save
  redirect_referer
  unless request.post?
    page = request['page'].to_s
    text = request['text'].to_s
    Db[page] = text
    redirect Rs( :show,
                 url_encode( page )
               )
  end
end
```

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki



Why Wiki



Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
ramaze Helpers
http://wiki.ramaze.net/Features/Helpers
A( 'link' )
  => "<a href='text'>text</a>"
Rs( :page )
  => "mySite/page"

def save
  : :
  redirect Rs( :show,
               url_encode( page )
             )
end
```

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki



Why Wiki



Example

- WikiController
 - 44 lines
- show page
- edit page
- save page
- BlueCloth
 - Mark-down
- YAMLStore

```
class WikiController
  < Ramaze::Controller
  def index ...
  def show ...
  def edit ...
  def save ...
  End

  Ramaze.start :adapter => :mongrel
end
```

And now ...

<http://github.com/manveru/ramaze> :: ramaze / examples / app / whywiki

ncg.asn.au
Newcastle Coders' Group

Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

NCG minimal Walkthrough

```

src\
|... start.rb
+-- controller\
|   ... controller.rb
+-- layout\
|   ... default.xhtml
+-- model\
|   ... mytables.rb
+-- view\
|   ... index.xhtml
Netbeans set-up
project\
spec\
  
```

Business logic
Page layout
Database
Content

<http://www.netbeans.org/features/ruby/index.html>

ncg.asn.au
Newcastle Coders' Group

Hullo Web

Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

class MainController
  < Ramaze::Controller
  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...
End

Ramaze.start
  :adapter => :mongrel,
  :port    => 7000
  
```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (one)

Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

class MainController
  < Ramaze::Controller
  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...
End

Ramaze.start
  :adapter => :mongrel,
  :port    => 7000
  
```

Ruby BDD support

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

think a bit

- Replace default page: ... index.xhtml.
 - `def index ... end;` → index.xhtml.
- Structure project (folders)
 - **layout/** : default.xhtml.
 - **controller/** : controller-one.xhtml.
 - **view/**
 - Apply layout to 'List' page
 - Make: list.xhtml.
- Quit message ... (??)

<http://www.netbeans.org/features/ruby/index.html>

netbeans

Hullo Web (one)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```
# Controller/controller.rb
class MainController
  < Ramaze::Controller
  ::
end
```

```
# start.rb
require 'ramaze'
require 'src/controller'

def main( argv )
  Ramaze.start
  :adapter => :mongrel
  :port    => 7000
end
```

<http://wiki.ramaze.net/Screencasts>

Hullo Web (one)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```
class MainController
  < Ramaze::Controller

  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...

end #MainController class
```

<http://wiki.ramaze.net/Screencasts>

Hullo Web (one)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```
class MainController
  < Ramaze::Controller

  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...

end #MainController class
```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (one)

Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

class MainController
  < Ramaze::Controller

  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...

end #MainController class

```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

one → two

- Replace default page: ... index.html.
 - `def index ... end;` → index.html.
- Set-up a layout: default.html.
 - Apply layout to 'List' page: list.html
 - **layout/** folder
- Structure project
 - **controller/** folder
- Quit message ...

<http://www.netbeans.org/features/ruby/index.html>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (two)

Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

Class MainController
  < Ramaze::Controller

  # Http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...

End #maincontroller class

```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group


two → three

- Add Forms processing: **Details**
 - Details.html
 - `def details ... end`
 - request.post? and request.get?
- Mutable page @title (<header /> element)
- Add 'Help'
 - http://localhost:7000/**help/topic**
- Re-brand: "Hello *Newcastle Coder's Group*"

<http://www.netbeans.org/features/ruby/index.html>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (three)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

class MainController
  < Ramaze::Controller

  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/quitx
  def quitx ...


end #MainController class

```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (three)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```

class MainController
  < Ramaze::Controller

  # http://localhost:7000/index
  def index ...

  # http://localhost:7000/list
  def list ...

  # http://localhost:7000/detail
  def details ...

  # http://localhost:7000/quitx
  def quitx ...


end #MainController class

```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

Hullo Web (three)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- An ORM

```


# http://localhost:7000/detail
def details
  @title = 'Collect details [a]'
  if( request.post? )
    @hidden = request['the_hidden']
    @realName= request['realName' ]
    @eMail   = request['eMail'   ]
    @password= request['password' ]
    @maritalStatus
      = request['maritalStatus']
    @showName= request['showName' ]
                ? 'shown' : 'hidden'
    ::
  end #details
  ::
end #MainController class

```

<http://wiki.ramaze.net/Screencasts>

ncg.asn.au
Newcastle Coders' Group

three → four



- Improve Presentation
 - Fill-out the index
 - Generate a dynamic table of contents
 - Include some extra content: "otherstuff"
 - Change "something" from page to an inclusion
 - Make the footing into a *real* footer
- Parameterise code (for example only)
- New pages ... new.xhtml & otherstuff.xhtml
- Add persistent data
 - model/user.rb

<http://www.netbeans.org/features/ruby/index.html>

Hullo Web (four)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters Sequel
- Forms
 - Parameters
- An ORM
 - Data stuff

```
# start.rb
require 'ramaze'
require 'controller/controller-four'
require 'model/user'

Ramaze.start :adapter => :mongrel,
             :port    => 7000
```

<http://wiki.ramaze.net/Screencasts>

Hullo Web (four)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters Sequel
- Forms
 - Parameters
- An ORM
 - Data stuff

```
# model/user.rb
require 'sequel'
Sequel::Model.plugin( :schema )
DB = Sequel.sqlite('sql_test.db', ... )

unless DB.table_exists?( :user )
  DB.create_table :user do
    set_primary_key :id
    String        :name
    : :
  end #create_table
end #table exists

class User < Sequel::Model( :user )
  : :
end
```

<http://wiki.ramaze.net/Screencasts>

Hullo Web (four)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters Sequel
- Forms
 - Parameters
- An ORM
 - Data stuff

```
# model/user.rb
require 'sequel'
Sequel::Model.plugin( :schema )
DB = Sequel.sqlite('sql_test.db', ... )

unless DB.table_exists?( :user )
  DB.create_table :user do
    set_primary_key :id
    String        :name
    : :
  end #create_table
end #table exists

class User < Sequel::Model( :user )
  : :
end
```

<http://wiki.ramaze.net/Screencasts>

Hullo Web (four)



Our Turn ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters Sequel
- Forms
 - Parameters
- An ORM
 - data stuff

```
# model/user.rb
: :
class User < Sequel::Model( :user )
  # Add a class method
  def User.add( name, passw, email, ... )
  def User.get_user( withName )
  end # User class
```

<http://wiki.ramaze.net/Screencasts>



Hullo Web



Recap ...

- Controller(s)
 - Add structure
- Templates
 - Layouts
- Actions
 - Parameters
- Forms
- Sequel ORM
 - data stuff

```

project\ ..... netbeans project
spec\ ..... bacon behaviour spec
src\
|   ... start.rb
+-- controller\
|   .... controller.rb
+-- layout\
|   ... defaultt.rb
+-- model\
|   .... user.rb
+-- view
|   .... details.xhtml
|   .... index.xhtml
|   .... list.xhtml
|   .... new.xhtml
|   .... otherstuff.xhtml

```

<http://wiki.ramaze.net/Screencasts>



What's In It For Me?

- Simple Development
 - 60% - 20% development
- Simple Maintenance
 - 40% - 80% software costs
- More fun and less of the dull stuff

<http://www.netbeans.org/features/ruby/index.html>

References and links

- CSS Reference
http://www.w3schools.com/css/css_reference.asp
- Dynamic Capabilities (wikipedia)
http://en.wikipedia.org/wiki/Dynamic_capabilities
- Heroku
<http://heroku.com>
- Introduction to Camping
<http://files.meetup.com/280030/Introduction%20to%20Camping.pdf>
- Journey into Ramaze
<http://book.ramaze.net>
- Lorem Ipsum
<http://www.lipsum.com>
- The Mythical Man Month
http://en.wikipedia.org/wiki/The_Mythical_Man-Month
- Model-View-Controller pattern
<http://www.enode.com/x/markup/tutorial/mvc.html>
- Xx
<http://www.>
- Ramaze
<http://ramaze.net>
- Carlson, L. & Richardson, L (2006) "**Ruby Cookbook**", O'Reilly
- Ruby Language
<http://www.ruby-language.com>
- Silicon Valley Tarot
<http://www.svtarot.com>

for interest

- A New Look at Test-Driven Development
http://blog.daveastels.com/files/BDD_Intro.pdf
- Accidental Technologist
<http://accidentaltechnologist.com>
- Advanced Topics in Programming Languages
- Fast, lightweight, cheap message passing in Java
<http://www.youtube.com/watch?v=37NaHRE0Sqw>
- Hello World Collection
<http://www.roesler-ac.de/wolfram/hello.htm>
- Instinct – BDD for Java
<http://code.google.com/p/instinct>

finis