About the speaker

Michael 'manveru' Fellinger
m.fellinger@gmail.com
http://manveru.net

Ramaze?

A modular and easy to use web application framework.

Web application framework

Quoting Wikipedia:

A software framework that is designed to support the development of dynamic websites, Web applications and Web services.

The framework aims to alleviate the overhead associated with common activities used in Web development.

Come again?

Ramaze is your new employee, stealing your work.

Easy to use?

```
1  require 'ramaze'
2
3  class MainController < Ramaze::Controller
4    def index
5      "Hello, World!"
6    end
7  end
8
9  Ramaze.start
```

Ramaze

```ruby
require 'ramaze'

class MainController < Ramaze::Controller
  helper :cache
  cache :index

  def index
    "Number: #{rand * 100}<br />
    <a href='/'>refresh</a>
    <a href='/invalidate'>invalidate cache</a>"
  end

  def invalidate
    action_cache.delete '/index'
    redirect '/'
  end
end

Ramaze.start
```

Ramaze

Principles

- KISS (Keep It Super Simple)
- Modular design
- Minimal dependencies
- Documentation & Examples
- Open development
- Full BDD (Behaviour Driven Design)

Keeping it simple

YAGNI: You ain't gonna need it

How Ramaze gets its KISS

Modular design?

Use what you want and how you want.

Even the most essential parts of Ramaze can easily be replaced or modified
without losing the advantage of the whole framework.

Dependencies

Rack

```
1 require 'rack'
2
3 hello = lambda do |env|
4   [200, {}, 'Hello, World!']
5 end
6
7 options = { :Host => 'localhost', :Port => 8080 }
8 Rack::Handler::Mongrel.run(hello, options)
```

Documentation

Constantly striving to have 100% of source code documented well.

Examples

Essential!

Don't even try to get people using your code without showing them examples.

Open development

Everybody is welcome to contribute.

How open development works

- Identify issue
- Send patch or description to mailing list or talk about it on our IRC channel
- Wait a bit and be (r)amazed

Contribution policy

- Learn how to make patches:
  `darcs record; darcs send`
- Patch is applied without objections
- Send us your ssh public key
- Push your patches to ramaze@ramaze.net:ramaze

Ramaze

Strong community

Our friendly and widely distributed community is always eager to help you.

Community communicates

- IRC
  irc.freenode.net - #ramaze
- Mailing list
  http://groups.google.com/group/ramaze
- Homepage
  http://ramaze.net

Ramaze

Community matters

“All bugs are fixed within 48 hours of reporting.”

Yet another framework?

Initially an experiment to reimplement Nitro

Every other web framework at that time forced you to follow its way or was too minimal.

The more, the merrier!

Deployment options

- CGI
- FastCGI
- LiteSpeed
- Mongrel
- SCGI
- WEBrick
- Ebb
- Evented Mongrel
- Swiftiplied Mongrel
- Thin
- Whatever comes along and fits into the Rack

Templating engines

- Amrita2
- Builder
- Erubis
- Ezamar
- Haml
- Liquid
- Markaby
- RedCloth
- Remarkably
- Sass
- Tagz
- Tenjin
- XSLT

Ezamar

```
<div class="userlist">
  <?r @users.each do |user| ?>
    #{user.name}
  <?r end ?>
</div>
~
~
```

Show me more

Ramaze ships with ~25 examples to satisfy you.
There are many more in the wiki.

```ruby
require 'ramaze'
require 'coderay'

class MainController < Ramaze::Controller
  layout '/layout'
  engine :None

  def index
    Dir['code/*.rb'].map{|file|
      content = File.read(file)
      CodeRay.scan(content, :ruby).
        html(:line_numbers => :inline,
             :wrap => :div)
    }.join("<br />\n")
  end
end

Ramaze.start :template_root => __DIR__
```

Current issues

- Growing codebase
  How to deal with more code from more people
- Too many specs?
  Takes almost 1 minute to run whole suite
- Still “obscure”
  RubyFools helps there
- Versioning

The future

Pushing the Ruby frontier, first class support for Ruby 1.9, JRuby and Rubinius

Even more extensive Documentation, Tutorials, Screencasts, HowTo, Examples.

Constant refactor, there is always something to improve.

Metadata

- Around 6k LoC
- Any ORM is OK
- Specs in Bacon
- ~2100 patches
- ~4 patches/day
- 23 direct contributors
- 14 templating engines supported
- 22% of patches contributed
- Runs on
  - Ruby >= 1.8.5
  - Jruby

Ramaze

History

- 29.09.2006 – First implementation
- 04.10.2006 – Darcs repository initialized
- 17.02.2007 – Build on Rack
- 11.05.2007 – Version 0.1.0
- 20.11.2007 – Version 0.2.0
- Today – Ramaze 0.3.9.5

Ramaze

Where to get Ramaze

Web	: http://ramaze.net
gem	: install ramaze
Darcs	: get http://darcs.ramaze.net/ramaze
nightly	: http://gems.ramaze.net

Where to find help

Web : http://ramaze.net
IRC : irc://irc.freenode.net/ramaze
Mailing list : http://groups.google.com/group/ramaze

Fin.

```ruby
require 'sequel'
require 'ramaze'

DB = Sequel.sqlite
class User < Sequel::Model
  set_schema{ varchar :name; text :description }
  create_table
end

class MainController < Ramaze::Controller
  helper :form

  def index
    form_for User
  end
end

Ramaze.start
```