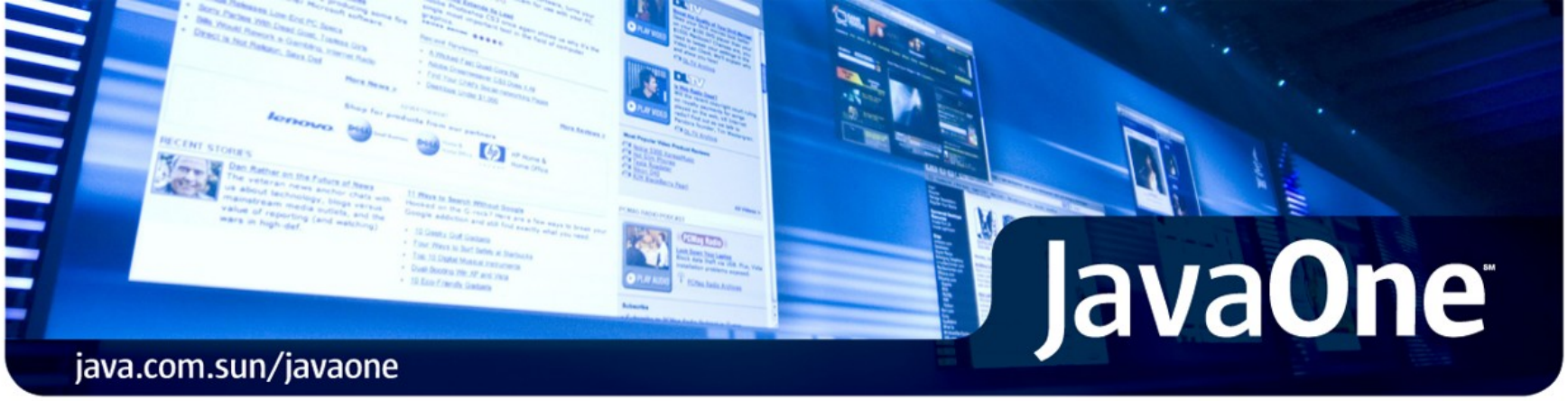


JRuby on Rails Deployment: What They Didn't Tell You

Nick Sieger, Staff Engineer, Sun Microsystems, Inc.

TS-6490

Java

Sun microsystems

Agenda

- Rails deployment background
- Mechanics of JRuby on Rails
- Preparations
- Packaging
- Post-deployment

The Concurrency Question

- Each Rails request must execute in isolation.

"Rails is not thread-safe? WTF?"

"Servlets have been multi-threaded since, like,1997!"

Context for Concurrency

Writing thread-safe code is hard

\+ Ruby's userland threads don't yield much benefit

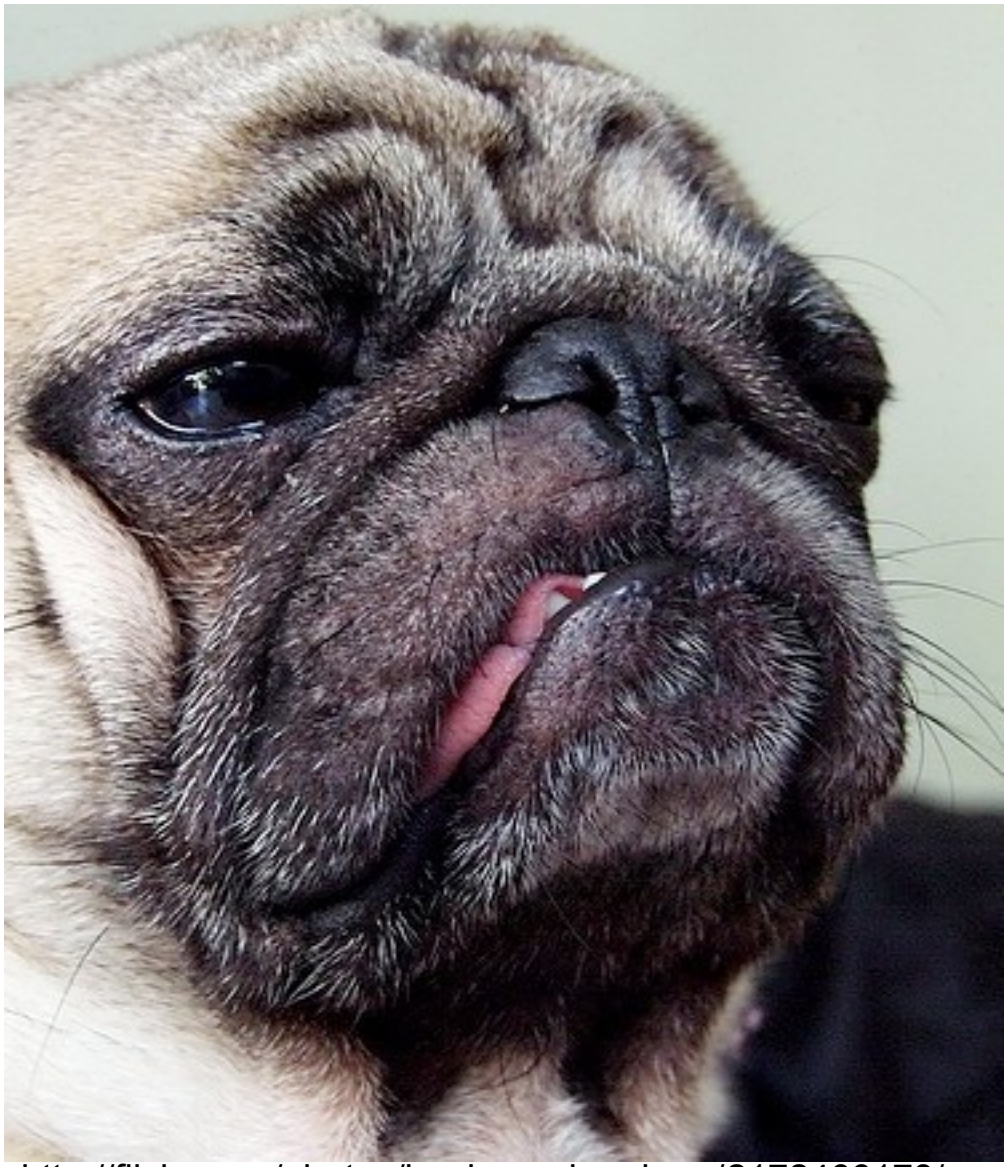
Not worth the effort

- Rails scales up by adding processes

CGI/FGCI

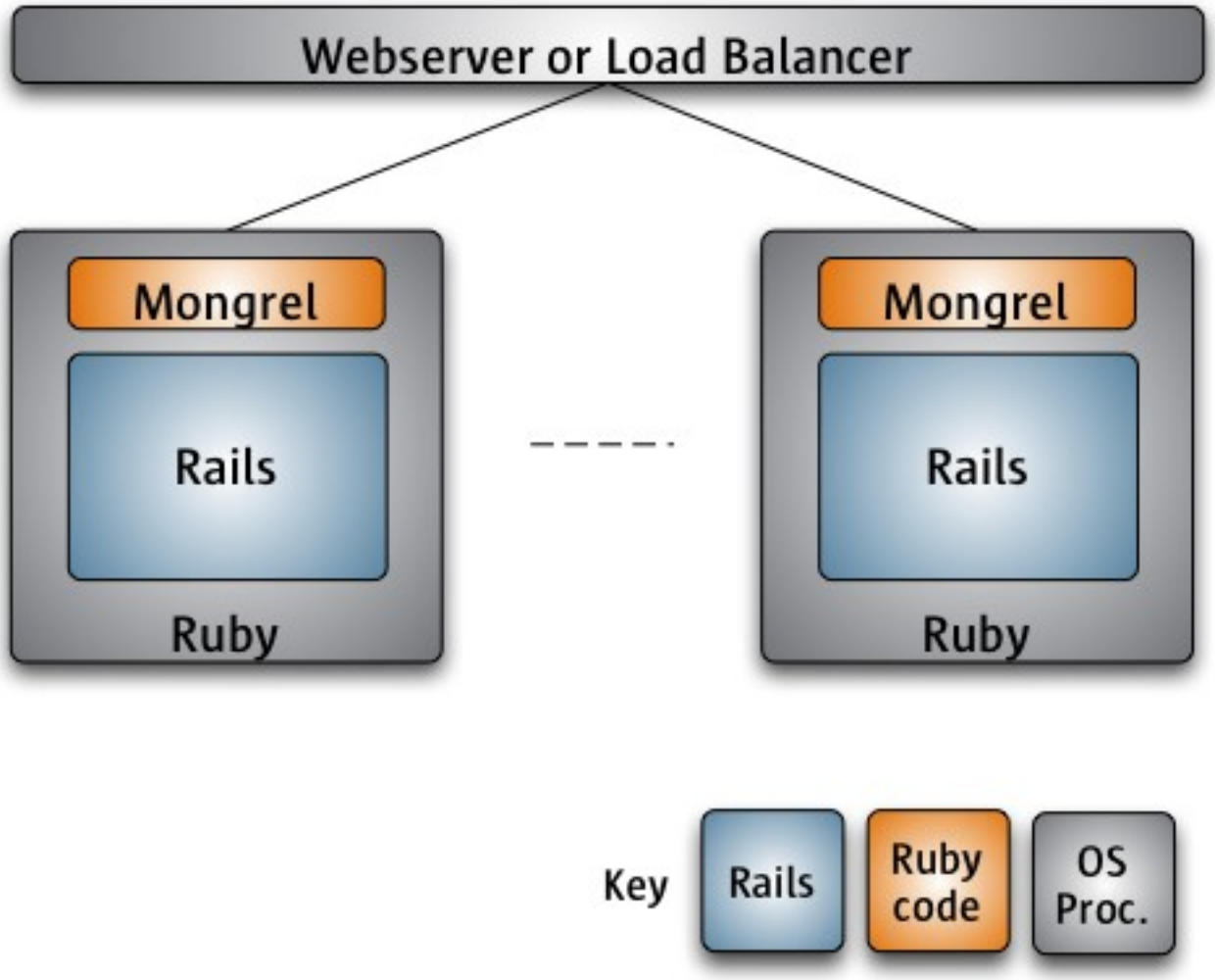
http://flickr.com/photos/counteragent/2190576349/

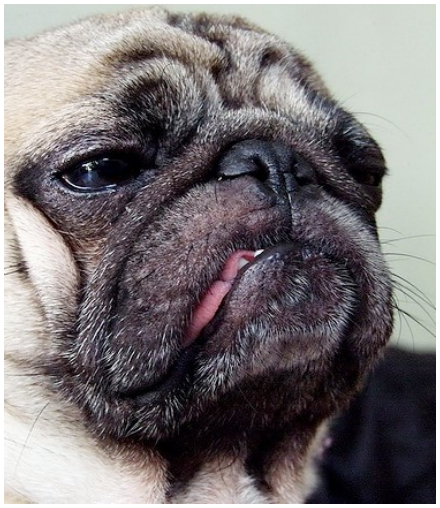
Mongrel



http://flickr.com/photos/bugbunnybambam/2172466178/

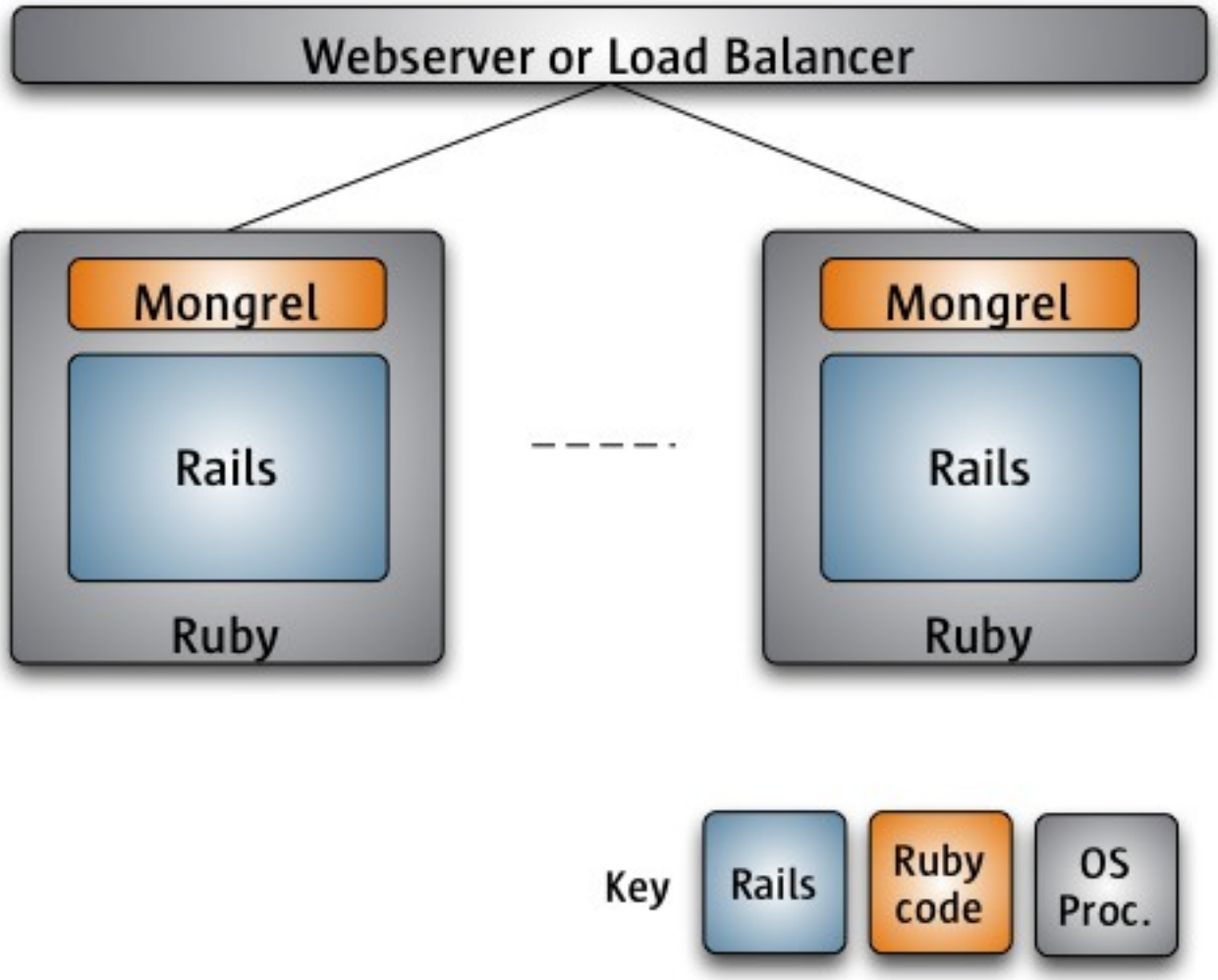
Mongrel Model



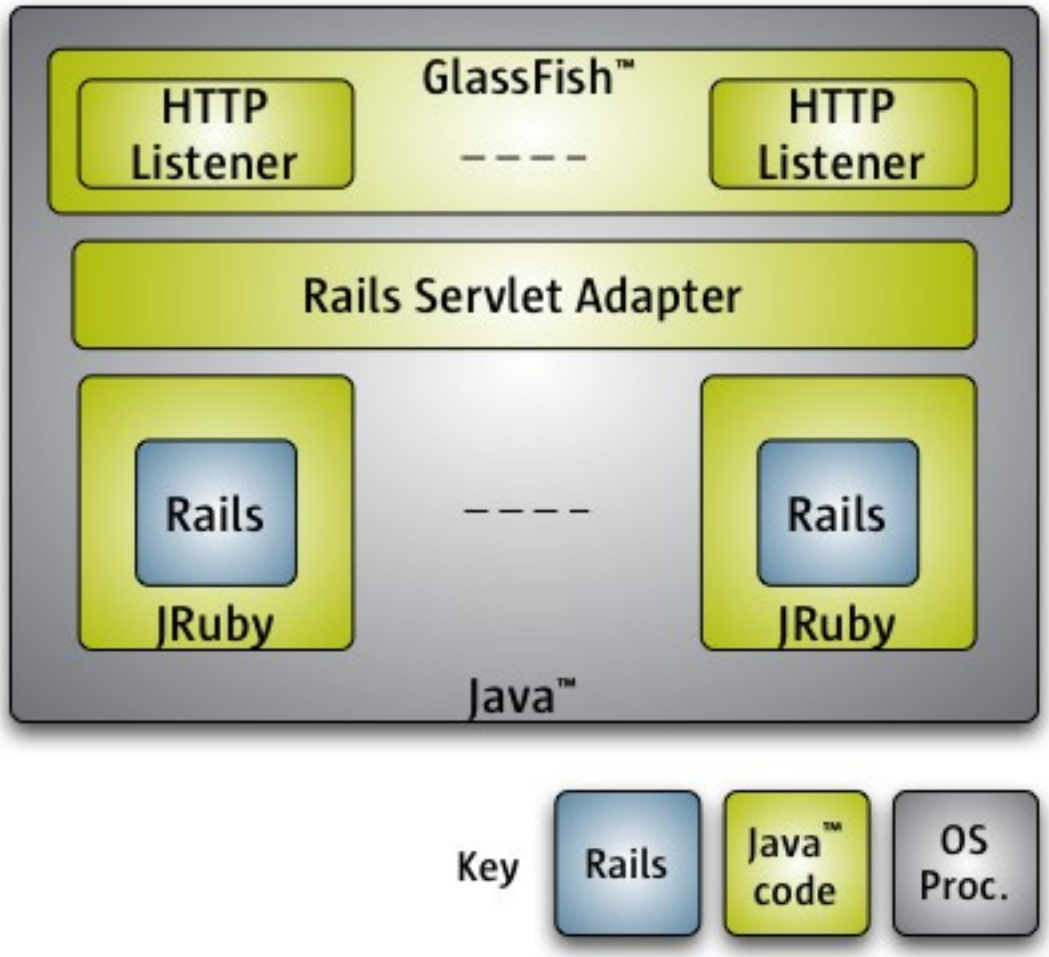
twitter =  * 180

http://highscalability.com/scaling-twitter-making-twitter-10000-percent-faster

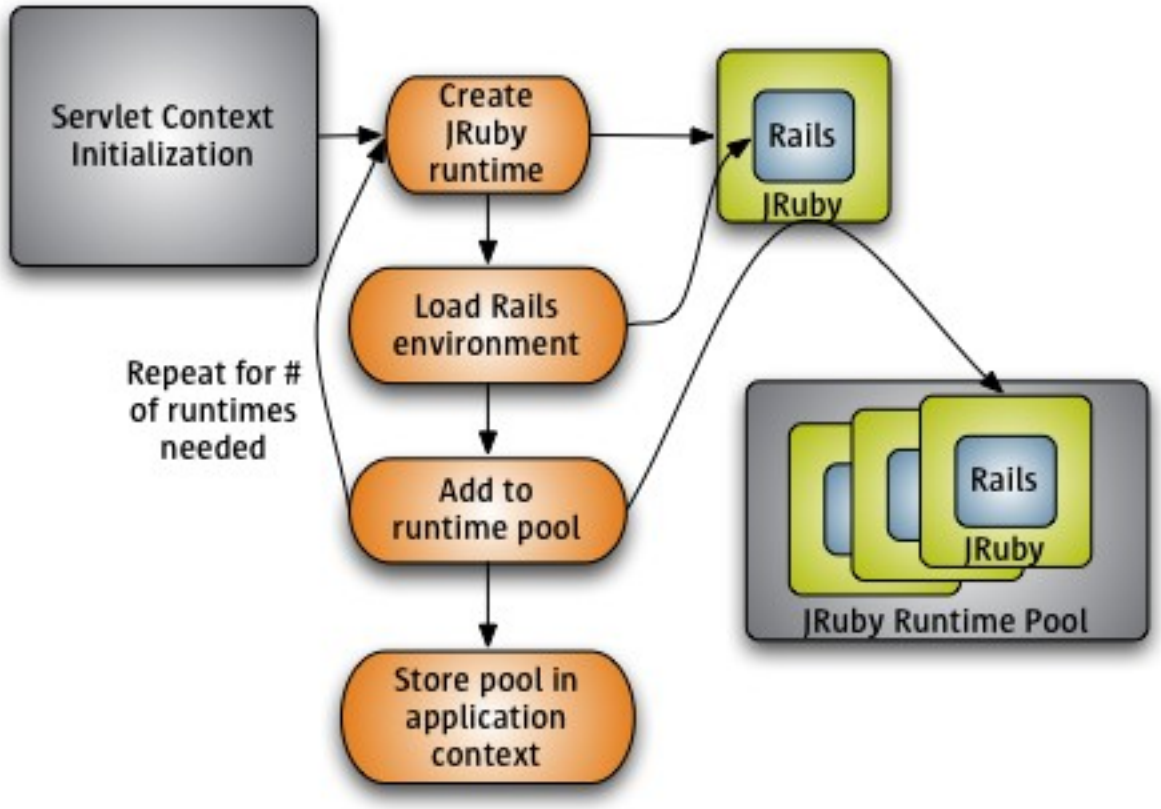
Instead of multiple processes...



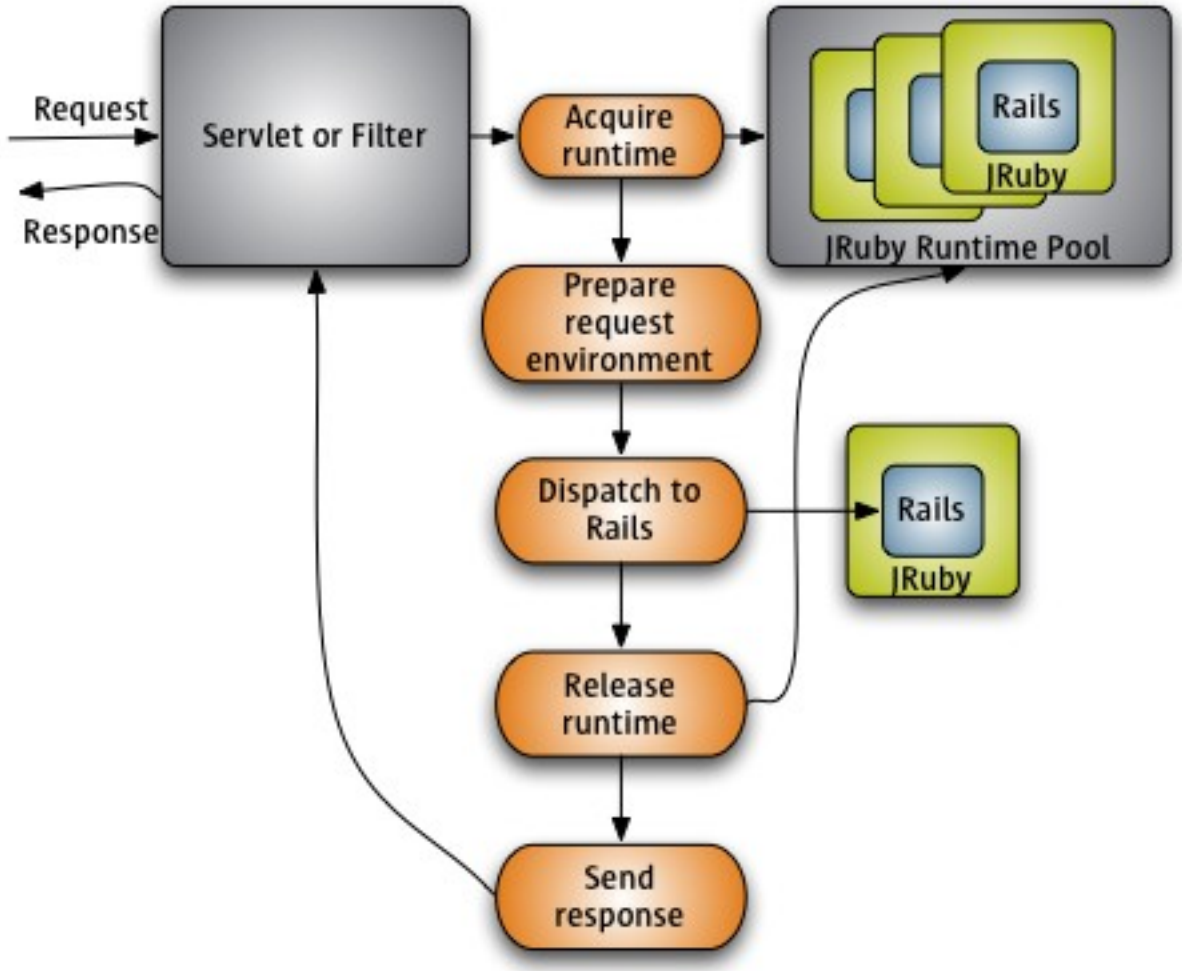
The Java Virtual Machine (JVM™):
All Ur Rails Are Belong To Us



Application Startup



Dispatching to Rails



Developing Your Rails Application

- TS-4806: JRuby on Rails: Web Development Evolved
- TS-5249: The NetBeans™ Ruby IDE: You Thought Rails Development Was Fun Before
- Both yesterday

Database Connectivity

- Install the ActiveRecord Java DataBase Connectivity (JDBC™) adapter gem (mysql)

```
$ jruby -S gem install activerecord-jdbcmysql-adapter
Successfully installed activerecord-jdbc-adapter-0.8
Successfully installed jdbc-mysql-5.0.4
Successfully installed activerecord-jdbcmysql-adapter-0.8
3 gems installed
Installing ri documentation for activerecord-jdbc-adapter-0.8...
Installing ri documentation for jdbc-mysql-5.0.4...
Installing ri documentation for activerecord-jdbcmysql-adapter-
0.8...
Installing RDoc documentation for activerecord-jdbc-adapter-0.8...
Installing RDoc documentation for jdbc-mysql-5.0.4...
Installing RDoc documentation for activerecord-jdbcmysql-adapter-
0.8...
```

Configure the Database

```
# config/database.yml
<% jdbc = defined?(JRUBY_VERSION) ? 'jdbc' : '' %>
development:
  adapter: <%= jdbc %>mysql
  encoding: utf8
  database: testapp_development
  username: root
  password:
  socket: /tmp/mysql.sock

# same for test/production...
```

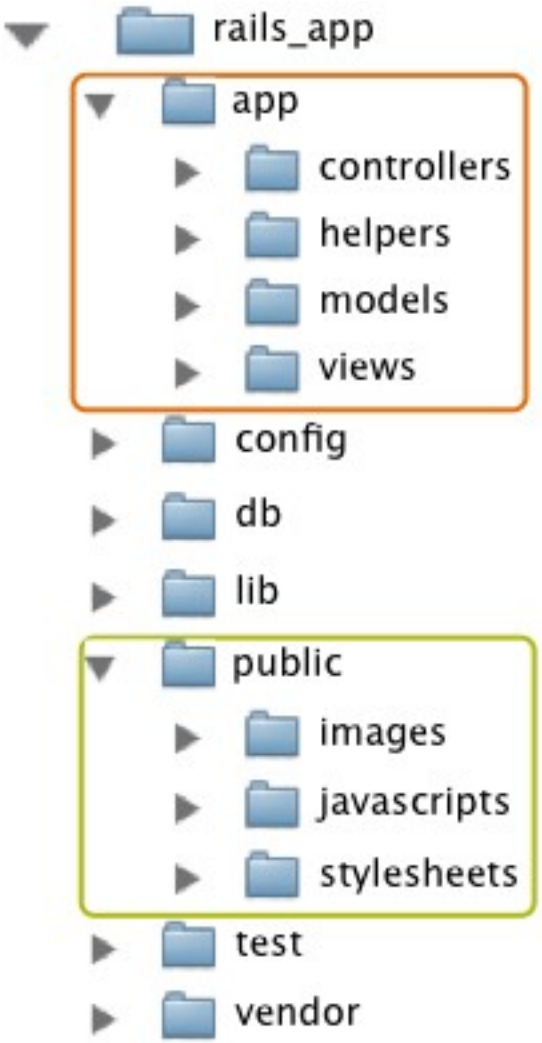
Examine dependencies

- Gems and libraries with native code == FAIL on JRuby
- Look for replacements/equivalents:
  - Mongrel, Hpricot OK
  - RMagick, ImageScience => ImageVoodoo
  - OpenSSL => JRuby-OpenSSL gem
  - Ruby/LDAP => JRuby/LDap
  - json => json_pure

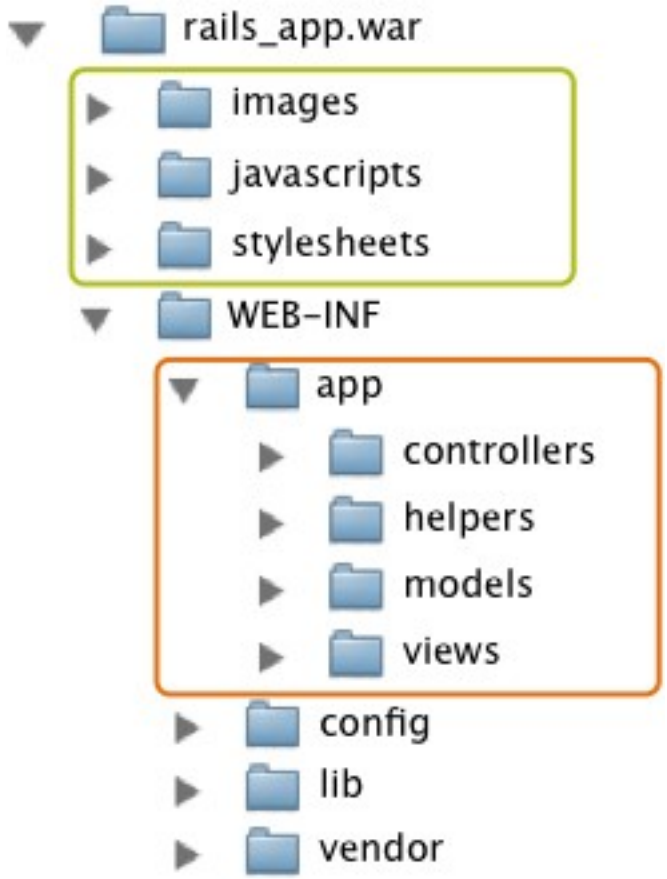
Try It!

```
$ jruby -S gem install mongrel
Successfully installed mongrel-1.1.3-java
1 gem installed
Installing ri documentation for mongrel-1.1.3-java...
Installing RDoc documentation for mongrel-1.1.3-java...
$ jruby script/server
=> Booting Mongrel (use 'script/server webrick' to force WEBrick)
=> Rails application starting on http://0.0.0.0:3000
=> Call with -d to detach
=> Ctrl-C to shutdown server
** Starting Mongrel listening at 0.0.0.0:3000
** Starting Rails with development environment...
** Rails loaded.
** Loading any Rails specific GemPlugins
** Signals ready.  TERM => stop.  USR2 => restart.  INT => stop (no restart).
** Rails signals registered.  HUP => reload (without restart).  It might not work
well.
** Mongrel 1.1.3 available at 0.0.0.0:3000
** Use CTRL-C to stop.
```

Application Layout (Rails)



Application Layout (WAR)



Using Warbler

- Installed as a gem (`jruby -S gem install warbler`)
- JRuby and servlet adapter bits included
- Automates war configuration

Installing Warbler as a Plugin

```
$ jruby -S warble pluginize
/Users/nicksieger/Projects/jruby/trunk/jruby/bin/jruby -S gem unpack warbler
Unpacked gem: '/Users/nicksieger/Projects/rails/testapp/vendor/plugins/warbler-
0.9.5'
$ jruby -S rake -T | grep war
rake war                                # Create testapp.war
rake war:app                            # Copy all application files into the ...
rake war:clean                          # Clean up the .war file and the stagi...
rake war:gems                           # Unpack all gems into WEB-INF/gems
rake war:jar                            # Run the jar command to create the .war
rake war:java_classes                   # Copy java classes into the .war
rake war:java_libs                      # Copy all java libraries into the .war
rake war:public                         # Copy all public HTML files to the ro...
rake war:webxml                         # Generate a web.xml file for the webapp
```

Configuring Warbler

- Generate config file
  - `jruby -S warble config` **-or-**
  - `jruby script/generate warble`
  - `=> config/warble.rb`

```
# Warbler web application assembly configuration file
Warbler::Config.new do |config|
  # ...

  # Gems to be packaged in the webapp.  ...
  config.gems += ["activerecord-jdbcmysql-adapter",
    "jruby-openssl"]
  config.gems["rails"] = "2.0.2"

  # ...
end
```

Runtime Pool Size

- config/warble.rb:

```
Warbler::Config.new do |config|
  # ...

  # Control the pool of Rails runtimes
  config.webxml.jruby.min.runtimes = 2
  config.webxml.jruby.max.runtimes = 4
end
```

Logging

```
if defined?(JRUBY_VERSION) && defined?($servlet_context)
  # Logger expects an object that responds to #write and #close
  device = Object.new
  def device.write(message)
    $servlet_context.log(message)
  end
  def device.close; end

  # Make these accessible to wire in the log device
  class << RAILS_DEFAULT_LOGGER
    public :instance_variable_get, :instance_variable_set
  end

  old_device = RAILS_DEFAULT_LOGGER.instance_variable_get "@log"
  old_device.close rescue nil
  RAILS_DEFAULT_LOGGER.instance_variable_set "@log", device
end
```

Session handling

- Rails sessions != Servlet sessions

```
config.action_controller.session_store = :java_servlet_store

# ...

class CGI::Session::JavaServletStore
  def initialize(session, options) end
  def restore; end
  def update; end
  def close; end
end
```

Database: JNDI

```
production:
  adapter: <%= jdbc %>mysql
  jndi: jdbc/testapp_production
  encoding: utf8
  database: testapp_production
  username: root
  password:
  socket: /tmp/mysql.sock
```

Database connections

- Use JNDI DataSource with a connection pool
- But Rails doesn't close connections by default (!)

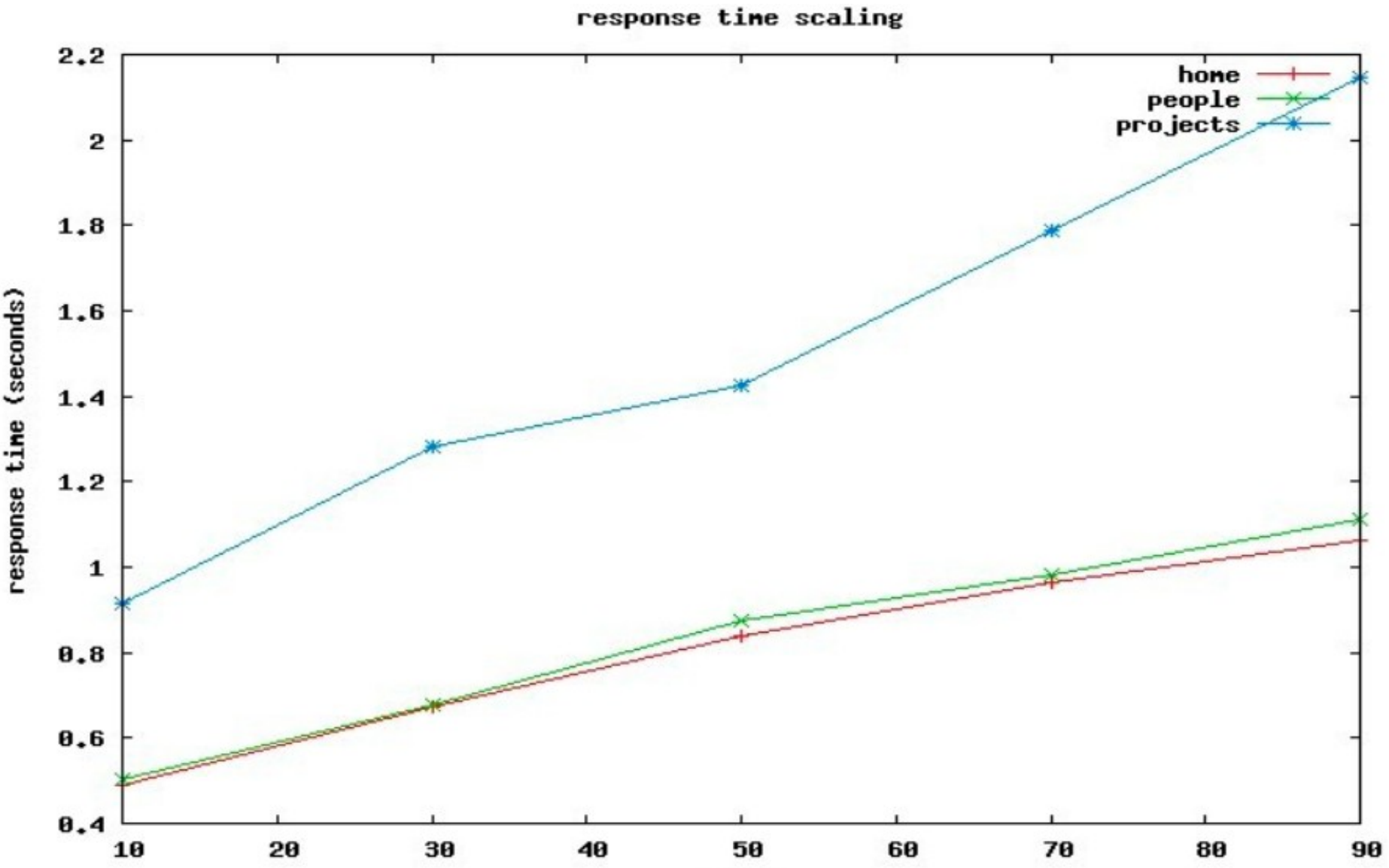
```
# config/initializers/close_connections.rb
if defined?($servlet_context)
  require 'action_controller/dispatcher'
  ActionController::Dispatcher.after_dispatch do
    ActiveRecord::Base.clear_active_connections!
  end
end
```

Caching and File stat'ing

- Ensure view caching is enabled (default in Rails 2.0.2)
  - `config.action_view.cache_template_loading = true`
- Avoid asset ID timestamp checks with RAILS_ASSET_ID
  - `ENV['RAILS_ASSET_ID'] = "r#{source_revision}"`
- Ensure full-page cache directory points to root of WAR
  - `config.action_controller.page_cache_directory`
  - `Rails.public_path` coming in 2.1

Performance...

...is a developing story.


response time scaling
response time (seconds)
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
10
20
30
40
50
60
70
80
90
home
people
projects

Precompiling Rails

- Compile Ruby code to bytecode before execution
- System property: -Djruby.compile.mode=FORCE
- Leverage existing tools for profiling, troubleshooting

```
class ErrorsController < ApplicationController
  def index
    java.lang.Thread.dumpStack
  end
end
```

```
java.lang.Exception: Stack trace
at java.lang.Thread.dumpStack(Thread.java:1176)
at
ruby...testapp.app.controllers.errors_controller.index__1
  (.../app/controllers/errors_controller.rb:3)
at ruby...action_controller.base.perform_action__66
  (.../gems/1.8/gems/actionpack-
2.0.2/lib/action_controller/base.rb:1158)
  ...
at ruby...action_controller.base.process__25
  (.../gems/1.8/gems/actionpack-
2.0.2/lib/action_controller/base.rb:388)
at ruby...action_controller.dispatcher.handle_request__18
  (.../gems/1.8/gems/actionpack-
2.0.2/lib/action_controller/dispatcher.rb:171)
at ruby...action_controller.dispatcher.__rescue_5
  (.../gems/1.8/gems/actionpack-
2.0.2/lib/action_controller/dispatcher.rb:115)
```

Introducing JRuby-Rack

- http://wiki.jruby.org/wiki/JRuby_Rack
- 0.9 -- First public release today!
- Also released – Warbler 0.9.9 – bundles jruby-rack 0.9

Introducing JRuby-Rack

- Adapter from servlets to Rack
  - Rack == Ruby web server abstraction layer
  - http://rack.rubyforge.org/
- Eliminate manual setup
- Runs Rails, Merb, any Rack-compatible framework
- Seamless integration with non-Rails servlet components