The New Face of JRuby

Nick Sieger
Engine Yard, Inc.
nsieger@engineyard.com
www.engineyard.com
@nicksieger
blog.nicksieger.com

JavaZone
September 9-10th in Oslo, Norway

Nick

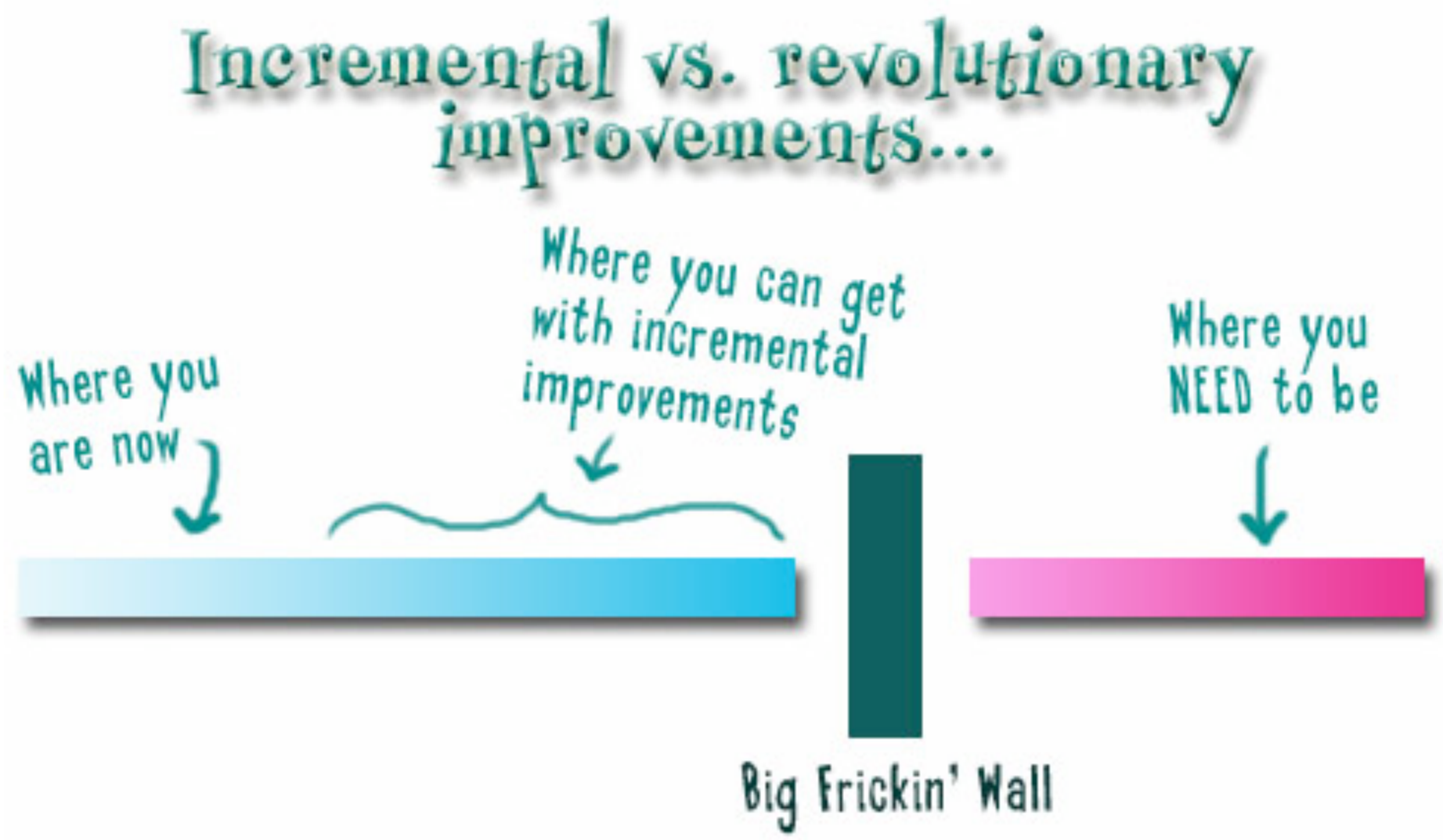
- works on JRuby at Engine Yard
- started using JRuby ~5 years ago
- passion: staying ahead of status quo

Slides

http://bit.ly/nicksieger-javazone

Why Ruby?

(Or, why any language that is not Java?)



http://headrush.typepad.com/photos/uncategorized/2007/04/06/incremental1.jpg

External Systems Integration

DSLs

Configuration,
Glue, Wiring

Dynamic Layer
Application code

Stable Layer
Java, statically typed
Legacy libraries

Develop, Deploy
Build, Test, Monitor

Nick's

Ola's

Nick's

http://olabini.com/blog/2008/01/language-explorations/

Layer	Stable	Dynamic
Language	Java	Ruby
Feel	Ceremony	Essence
Resilience to Change	infrequent	frequent
Dependency Load	Fewer external dependencies	More external dependencies
Development Velocity	Slower	Faster
Flexibility	Hardened	Malleable

Ruby

- dynamic, strongly typed
- everything is an object
- programmer-friendly
- focus on readability, productivity
- thriving community outside of Java

Java

```java
package demo;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Ruby

```ruby
puts "Hello World"
```

Java

```java
package demo;

public class Point {
    private int dimension;
    private int[] coordinates;

    public Point(int x, int y) {
        dimension = 2;
        coordinates = {x, y};
    }

    public void setDimension(int newDimension) {
        dimension = newDimension;
    }

    public int getDimension() {
        return dimension;
    }

    public void setCoordinates(int[] newCoordinates) {
        coordinates = newCoordinates;
    }

    public int[] getCoordinates() {
        return coordinates;
    }
}
```

Ruby

```ruby
class Point
  attr_accessor :dimension, :coordinates

  def initialize(*coordinates)
    @dimension = coordinates.length
    @coordinates = coordinates
  end
end
```

Java

```
FileInputStream input = null;
FileOutputStream output = null;
try {
    try {
        input = new FileInputStream("/tmp/src.txt");
    } catch (IOException io1) {
        throw new RuntimeException("couldn't open input file", io1);
    }
    try {
        output = new FileOutputStream("/tmp/dest.txt");
    } catch (IOException io2) {
        throw new RuntimeException("couldn't open output file", io2);
    }

    byte[] buf = new byte[8192];
    int numBytesRead = 0;
    while ((numBytesRead = input.read(buf)) != -1) {
        output.write(buf, 0, numBytesRead);
    }
} catch (IOExeption io) {
    throw new RuntimeException("couldn't read/write or something", io);
} finally {
    try {
```

```
        try {
            input = new FileInputStream("/tmp/src.txt");
        } catch (IOException io1) {
            throw new RuntimeException("couldn't open input file", io1);
        }
        try {
            output = new FileOutputStream("/tmp/dest.txt");
        } catch (IOException io2) {
            throw new RuntimeException("couldn't open output file", io2);
        }

        byte[] buf = new byte[8192];
        int numBytesRead = 0;
        while ((numBytesRead = input.read(buf)) != -1) {
            output.write(buf, 0, numBytesRead);
        }
    } catch (IOException io) {
        throw new RuntimeException("couldn't read/write or something", io);
    } finally {
        try {
            if (input != null) {
                input.close();
            }
        } catch (IOException ignored1) { }
        try {
            if (output != null) {
                output.close();
            }
        } catch (IOException ignored2) { }
    }
```

Java

Ruby

```ruby
File.open("/tmp/src.txt") do |input|
  File.open("/tmp/dest.txt", "w") do |output|
    output << input.read
  end
end
```

JRuby

- Ruby 1.8-compatible runtime in Java
- 3+ full-time contributors
- Full interpreted and compiled modes
- Active community

JRuby 1.4

- New 1.4 release early October
- Reified java classes
- Annotation support
- Early invokedynamic support
- Improved Ruby 1.8.7 and 1.9 support

JRuby:

Hello Swing

Java Integration (JI)

JI: Ruby-like Methods

Java

```
Locale.US
System.currentTimeMillis()
locale.getLanguage()
date.getTime()
date.setTime(0)
file.isDirectory()
```

Ruby

```
Locale::US
System.current_time_millis
locale.language
date.time
date.time = 0
file.directory?
```

JI: Ruby Conversions

- primitives: numbers, strings, booleans, nil
- collections:
  - arrays -> lists
  - hashes -> maps
- attempt “principle of least surprise”

JI: Ruby Conversions

```
ruby_array = [10, 5, 1]
java.util.Collections.sort(ruby_array)
ruby_array
# => [1, 5, 10]
```

JI: Java Extensions

```
h = java.util.HashMap.new
h["key"] = "value"
h["key"]
  # => "value"
h.get("key")
  # => "value"
h.each {|k,v| puts k + ' => ' + v}
  # key => value
h.to_a
  # => [["key", "value"]]
```

JI: Java Extensions

```
module java::util::Map
  include Enumerable

  def each(&block)
    entrySet.each { |pair| block.call([pair.key, pair.value]) }
  end

  def [](key)
    get(key)
  end

  def []=(key,val)
    put(key,val)
    val
  end
end
```

JI: Helpers

```
["a", "b", "c"].to_java :string
  # => new String[] {"a", "b", "c"}
```

```
import java.io.FileOutputStream
stream = FileOutputStream.new("/tmp/hello.txt")
stream.write("Hello".to_java_bytes)
stream.close
```

JI: Interface Conversion

- what happens if...

```
package java.util.concurrent;
public class Executors {
    // ...
    public static Callable callable(Runnable r) {
        // ...
    }
}
```

JI: Interface Conversion

```
class SimpleRubyObject
end

import java.util.concurrent.Executors
callable = Executors.callable(SimpleRubyObject.new)
callable.call
```

JI: Interface Conversion

```
class SimpleRubyObject
end

import java.util.concurrent.Executors
callable = Executors.callable(SimpleRubyObject.new)
callable.call
```

JI: Interface Conversion

```
class SimpleRubyObject
end

import java.util.concurrent.Executors
callable = Executors.callable(SimpleRubyObject.new)
callable.call

# => undefined method `run' for #<SimpleRubyObject:0xfd5428>
(NoMethodError)
```

JI: Interface Conversion

```
class SimpleRubyObject
end

import java.util.concurrent.Executors
callable = Executors.callable(SimpleRubyObject.new)
callable.call

# => undefined method `run' for #<SimpleRubyObject:0xfd5428>
(NoMethodError)

class SimpleRubyObject
  def run
    puts "hi"
  end
end
callable.call
# => hi
```

JI: Closure Conversion

- taking that last example further...

```
import java.util.concurrent.Executors
callable = Executors.callable { puts "hi" }
callable.call
# => hi


button.add_action_listener do |event|
  event.source.text = "Pressed!"
end
```

From Ruby

To Java

Rake

- Build DSL in Ruby
- Tasks, prerequisities
- File- and pattern-based rules
- Extensible – it's just Ruby
- Clean fun!

Rake

```
desc "Compile the code"
task "compile" => "build" do |t|
  files = FileList["src/**/*.java"]
  cpath = FileList["lib/*.jar"].join(':')
  sh "javac -d #{t.prerequisites.first} -classpath #{cpath} #{files}"
end

desc "Create a jar file of the compiled code"
task "jar" => "compile" do
  sh "jar cf greeter.jar -C build ."
end
```

Rake + Ant

```
desc "Compile the code"
task "compile" => "build" do |t|
  ant.javac :srcdir => "src", :destdir => "build" do
    ant.classpath do
      FileList["lib/*.jar"].each do |jar|
        ant.pathelement :path => jar
      end
    end
  end
end

desc "Create a jar file of the compiled code"
task "jar" => "compile" do
  ant.jar :destfile => "greeter.jar", :basedir => "build"
end
```

Maven

```
<plugin>
  <groupId>org.jruby.plugins</groupId>
  <artifactId>jruby-rake-plugin</artifactId>
  <executions>
    <execution>
      <id>tests</id>
      <phase>test</phase>
      <goals><goal>rake</goal></goals>
      <configuration>
        <args>default</args>
      </configuration>
    </execution>
  </executions>
</plugin>
```

Route around Maven with Rake!

RSpec

- http://rspec.info
- BDD tool in Ruby (unit test analogue)
- Clean, readable, executable examples
- Use it to test Java!

RSpec

```
package org.jruby.rack;

import javax.servlet.Filter;

public class RackFilter implements Filter {
    public void doFilter(ServletRequest request,
            ServletResponse response, FilterChain chain)
            throws IOException, ServletException {
        // ...
        chain.doFilter(/* modified */ request, response);
        // ...
    }
}
```

RSpec

```
import org.jruby.rack.RackFilter

describe RackFilter do
  before :each do
    @request, @response = mock("request"), mock("response")
    @chain = mock("filter chain")
    @filter = RackFilter.new
  end

  it "should dispatch to the filter chain" do
    @chain.should_receive(:doFilter).and_return do |_, resp|
      resp.setStatus(200)
    end
    @response.should_receive(:setStatus).with(200)
    @filter.doFilter(@request, @response, @chain)
  end
end
```

Cucumber

Behaviour Driven Development
with elegance and joy

JMX

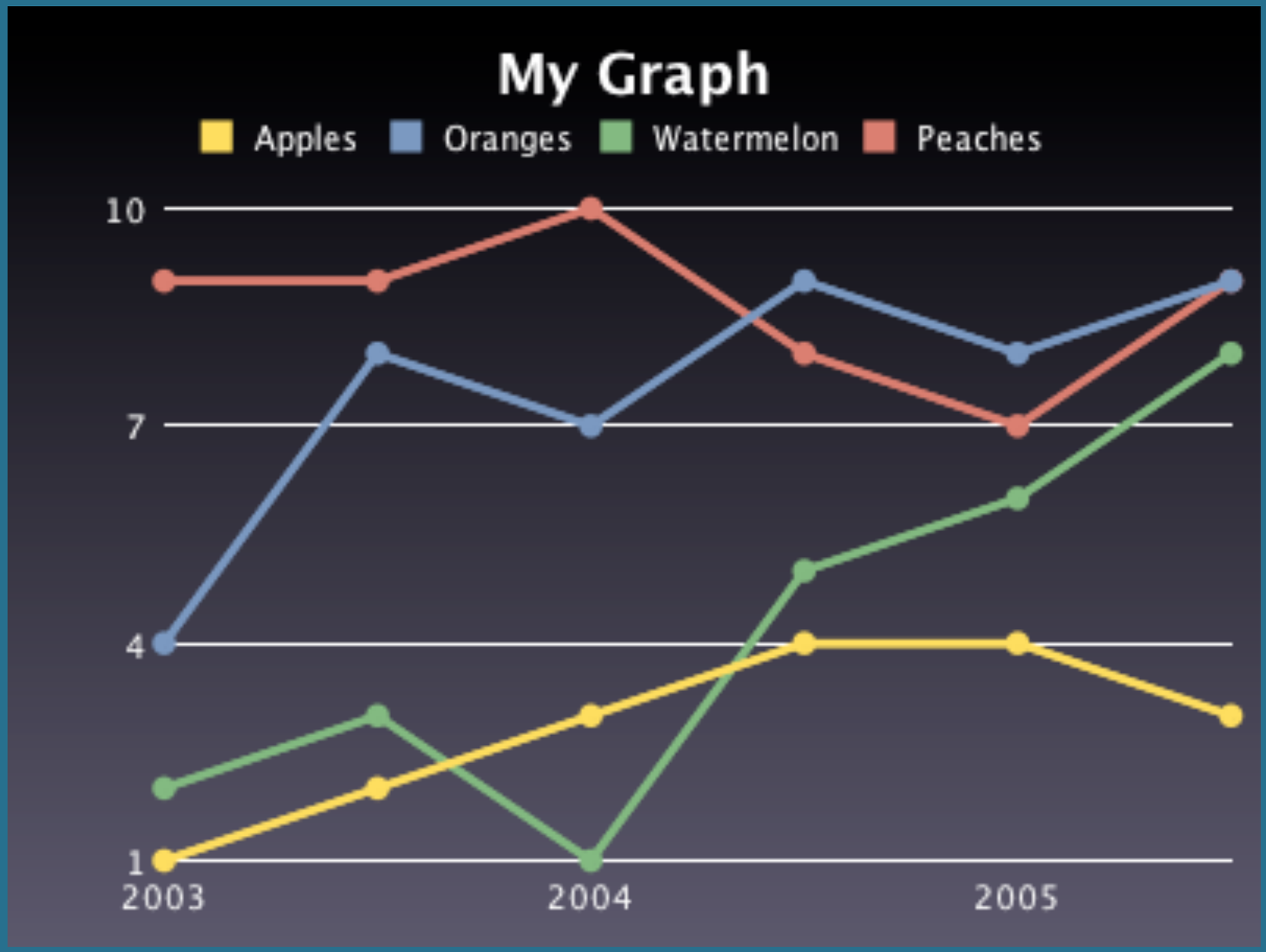
```
$ jruby -S gem install jmx
```

```
require 'rubygems'
require 'jmx'

server = JMX.connect(:host => "localhost", :port => 8686,
  :user => "...", :password => "...")
memory_bean = server["java.lang:type=Memory"]
puts memory_bean.heap_memory_usage.used
```

Gruff Graphs

```
require 'gruff'

g = Gruff::Line.new("400x300")

g.title = "My Graph"
g.data("Apples", [1, 2, 3, 4, 4, 3])
g.data("Oranges", [4, 8, 7, 9, 8, 9])
g.data("Watermelon", [2, 3, 1, 5, 6, 8])
g.data("Peaches", [9, 9, 10, 8, 7, 9])

g.labels = {0 => '2003', 2 => '2004', 4 => '2005'}

File.open("simple.png", "w") {|f| f << g.to_blob}
```

Gruff Graphs



JMX + Gruff

From Java

To Ruby

JRuby-in-a-Jar

- Complete Ruby distribution as a jar
- Including Ruby stdlib, Rubygems, Rake, RSpec
- Available in Maven:

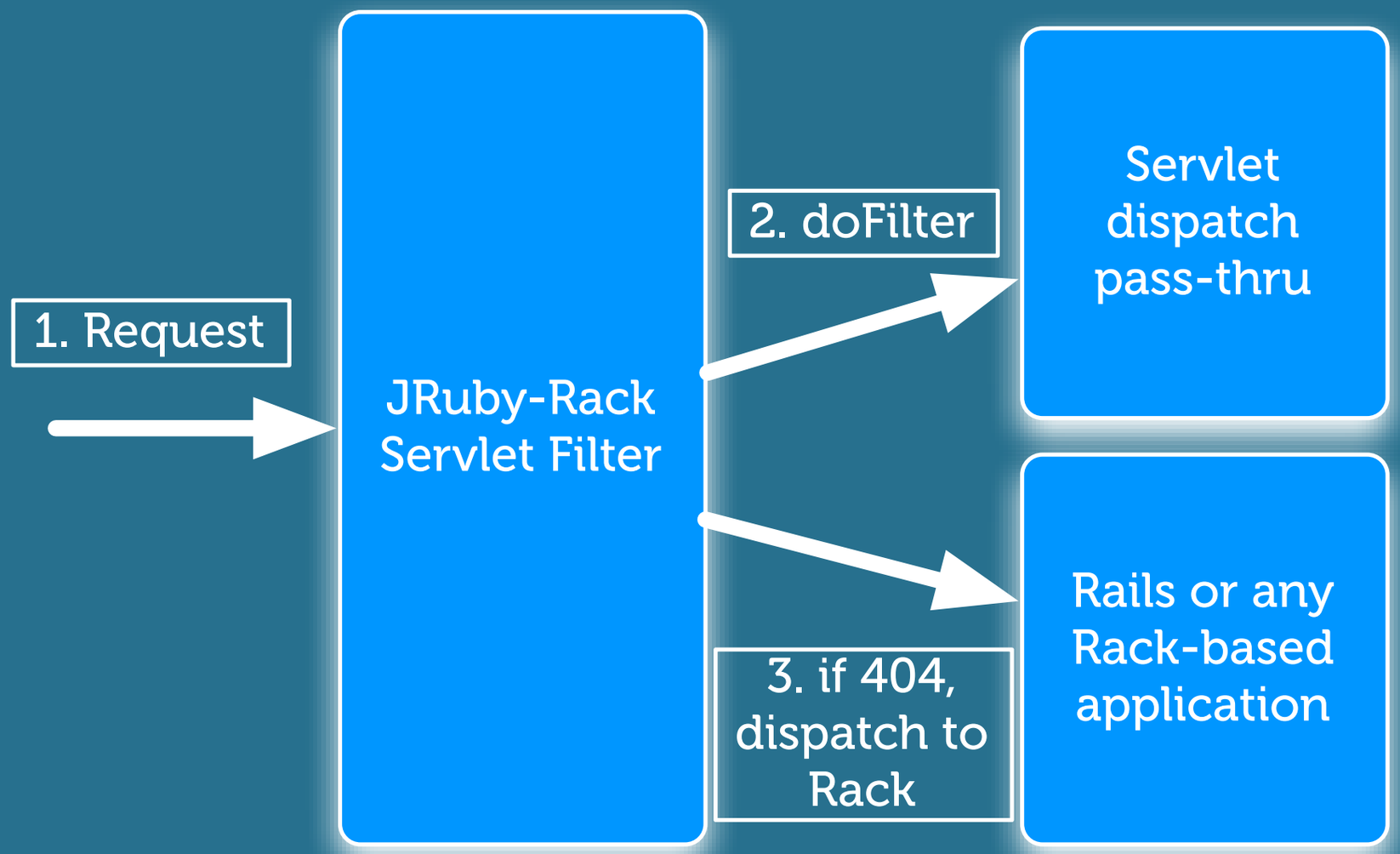
```
<dependency>
  <groupId>org.jruby</groupId>
  <artifactId>jruby-complete</artifactId>
  <version>1.3.1</version>
</dependency>
```

JSR-223

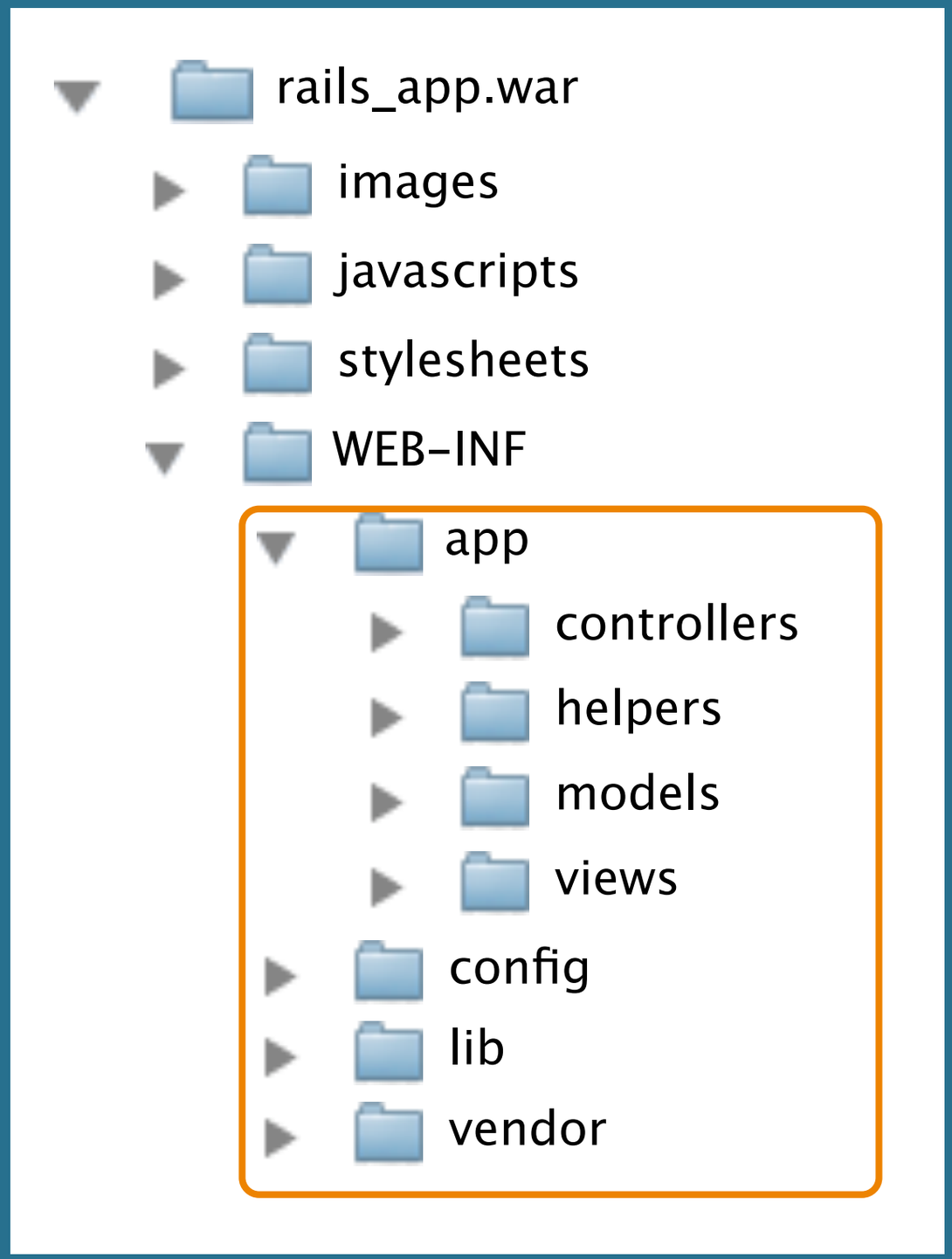
```
// import javax.script.*;

ScriptEngineManager factory = new ScriptEngineManager();
ScriptEngine engine = factory.getEngineByName("jruby");
engine.put("name", "JavaZone");
engine.eval("puts 'Hello ' + $name");
```

http://java.sun.com/developer/technicalArticles/scripting/jruby/

JRuby-Rack


1. Request
JRuby-Rack
Servlet Filter
2. doFilter
Servlet
dispatch
pass-thru
3. if 404,
dispatch to
Rack
Rails or any
Rack-based
application

JRuby-Rack


rails_app.war
images
javascripts
stylesheets
WEB-INF
app
controllers
helpers
models
views
config
lib
vendor

JRuby-Rack

```
<web-app>
  <filter>
    <filter-name>RackFilter</filter-name>
    <filter-class>org.jruby.rack.RackFilter</filter-class>
  </filter>

  <filter-mapping>
    <filter-name>RackFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

  <listener>
    <listener-class>
      org.jruby.rack.rails.RailsServletContextListener
    </listener-class>
  </listener>
</web-app>
```

Reified Classes

- Define class, method, parameter annotations
- Define method signatures
- become_java!
- Creates and loads a Java class

JUnit

```
require 'junit'

class MyJUnitTest
  include JUnit

  Before()
  def setup
    @fixture = "foo"
  end

  Test()
  def equals
    assert_equals "bar", @fixture
  end
end
```

Hibernate

```
require 'hibernate'

class Event
  include Hibernate::Model
  hibernate_attr :title => :string, :created => :date
  hibernate!
end
```

Jersey

Jersey

```
package ruby;

import javax.ws.rs.*;

@Path("/helloworld")
public class HelloWorldResource {
    @GET
    @Produces("text/plain")
    public String getMessage() {
        return "Hello World";
    }

    @GET
    @Produces("text/plain")
    @Path("{id}")
    public String getPersonalMessage(@PathParam("id") String message) {
        return "Hello " + message;
    }
}
```

Ruby-Jersey

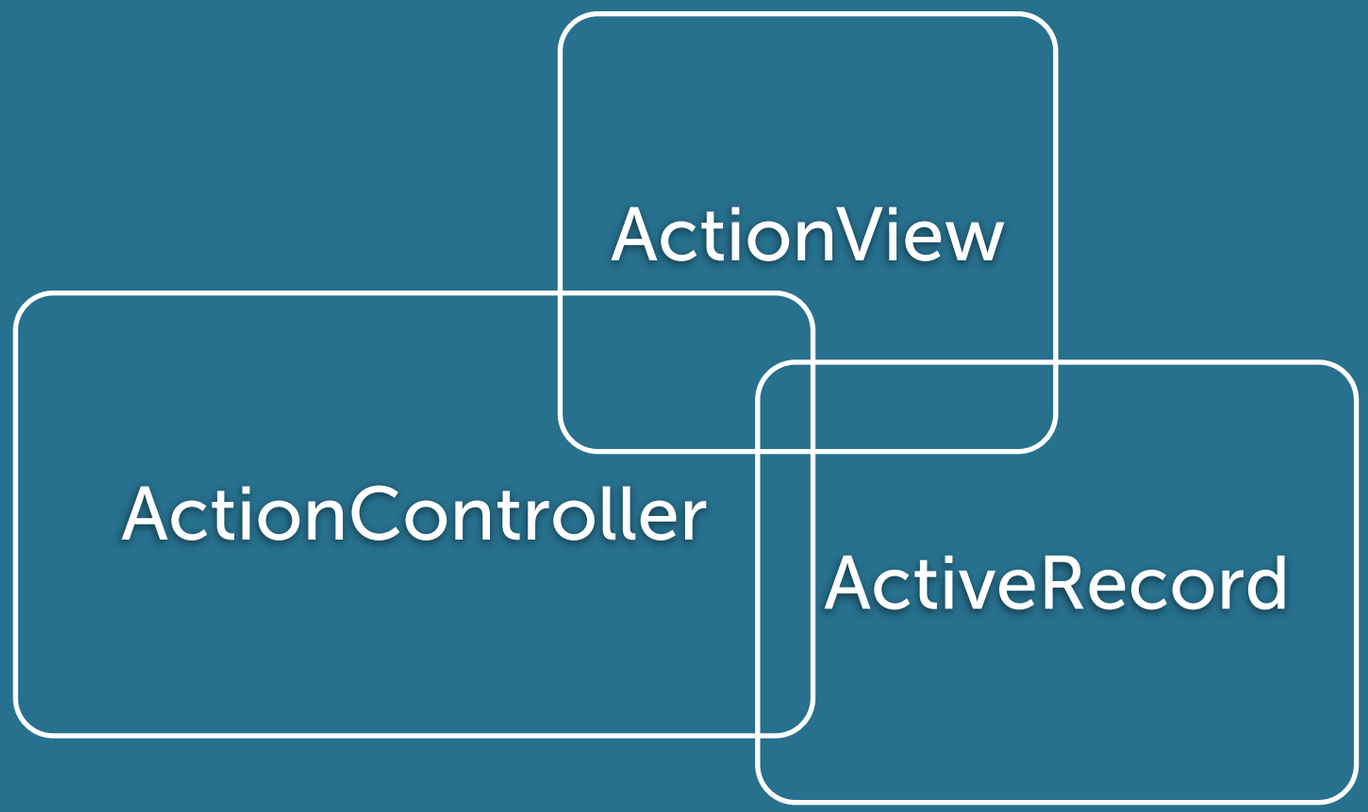
```
require 'ruby-jersey'

class Hello < RubyJersey::Resource
  GET()
  Produces("text/plain")
  Returns(java.lang.String)
  def hello
    "Hello Ruby Jersey!"
  end

  GET()
  Path("{id}")
  Produces("text/plain")
  Returns(java.lang.String)
  Parameters(java.lang.String)
  ParameterAnnotations(PathParam("id"))
  def personal_hello(msg)
    "Hello #{msg}!"
  end
end
```

Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/

Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/

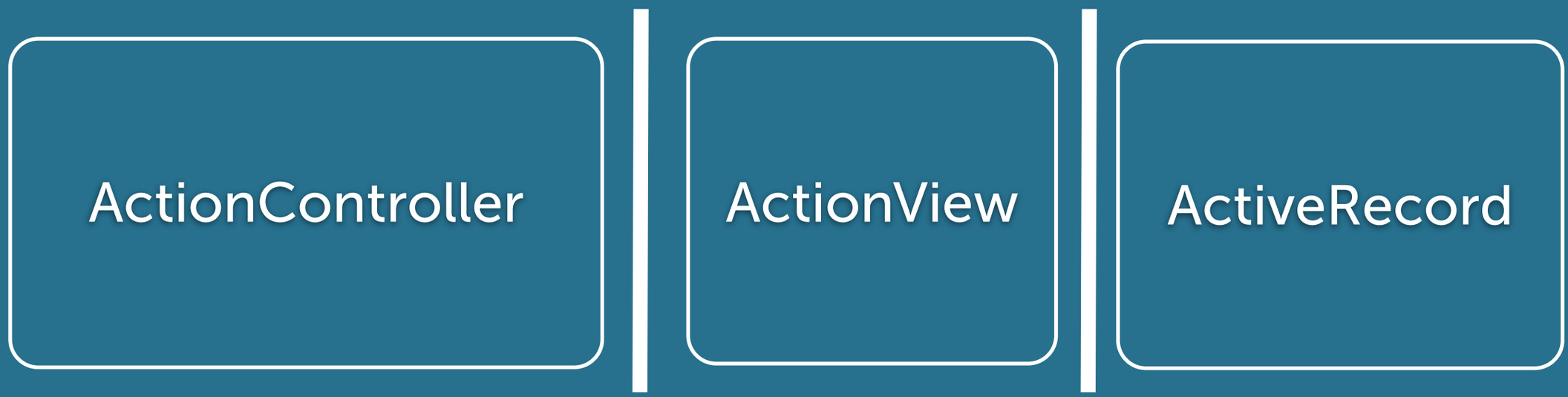


Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/

Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/



Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/

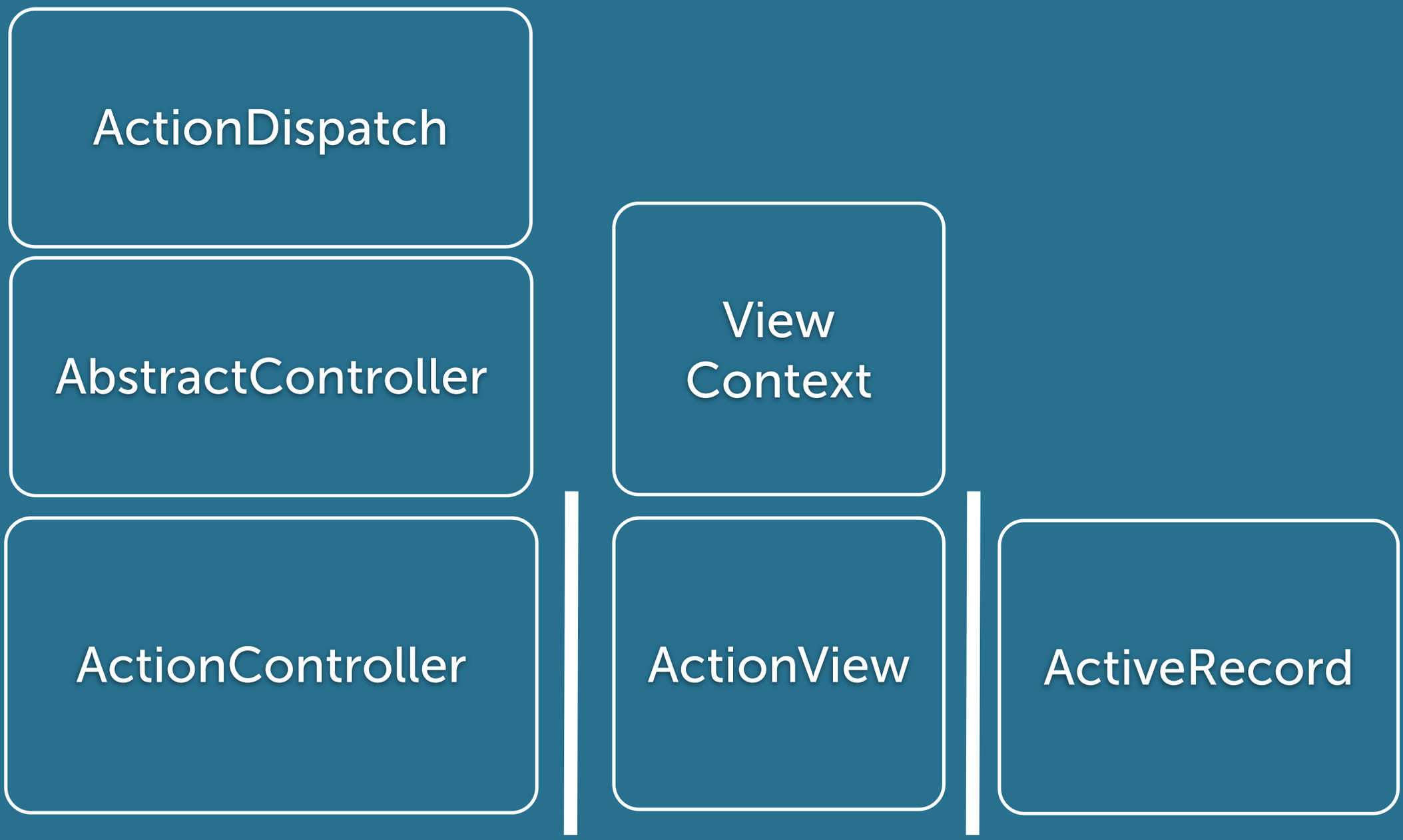
ActionDispatch

AbstractController

ActionController | ActionView | ActiveRecord

Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/



Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/

ActionDispatch

AbstractController

ActionController

View
Context

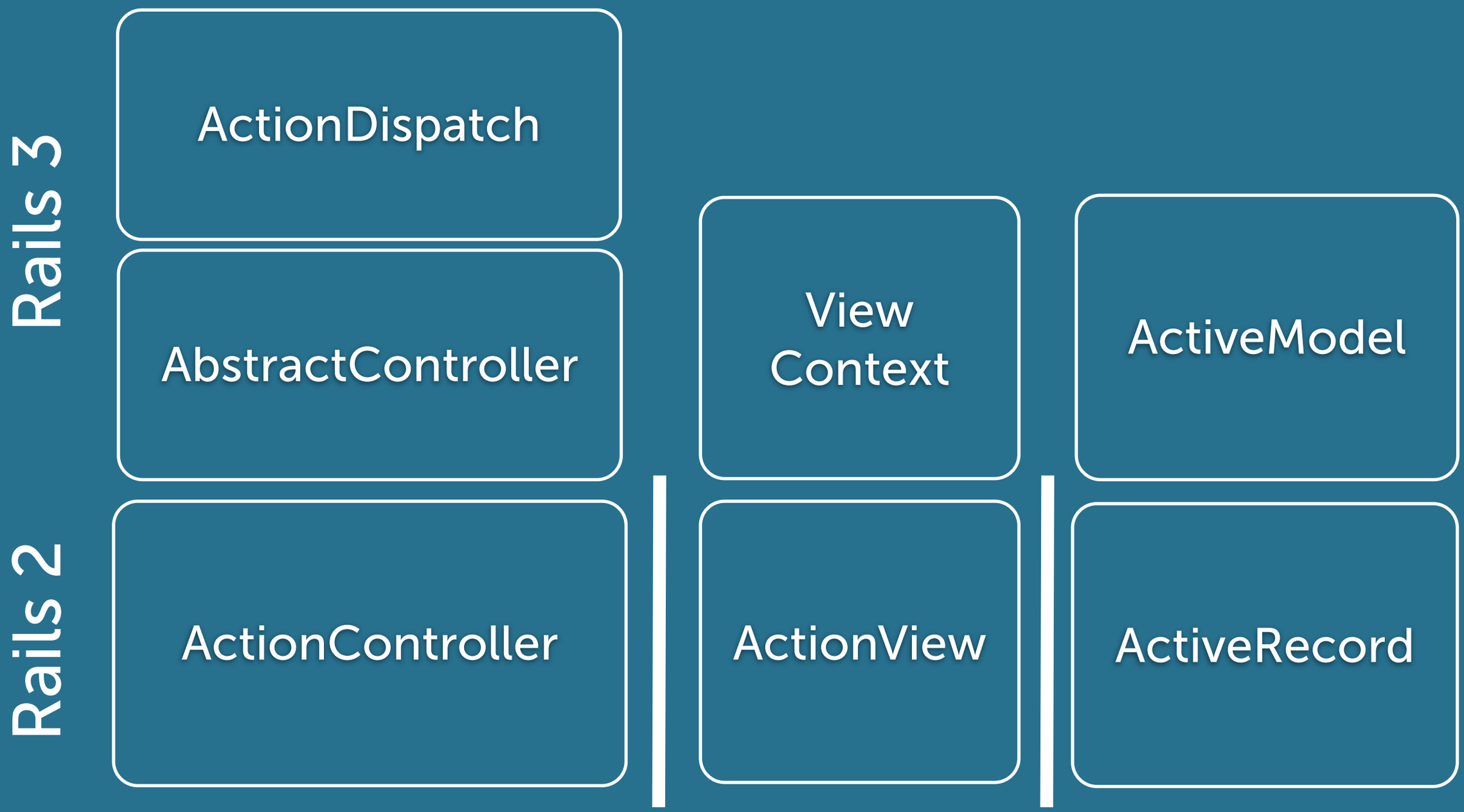
ActionView

ActiveModel

ActiveRecord

Rails 3.0

http://yehudakatz.com/2009/07/19/rails-3-the-great-decoupling/



Recapitulation

- Ruby => Java
  - Scripting
  - Build
  - Test
  - Deploy
  - DSLs
- Java => Ruby
  - Embedding
  - Application code
  - Java interop
  - DSLs

Go Try It!

Stickers!

Resources

- http://www.jruby.org/
- http://github.com/nicksieger
  - advent-jruby, ruby-jersey, jibernate, jruby-rack, rake-ant
- http://rspec.info, http://cukes.info