

play ▶

at

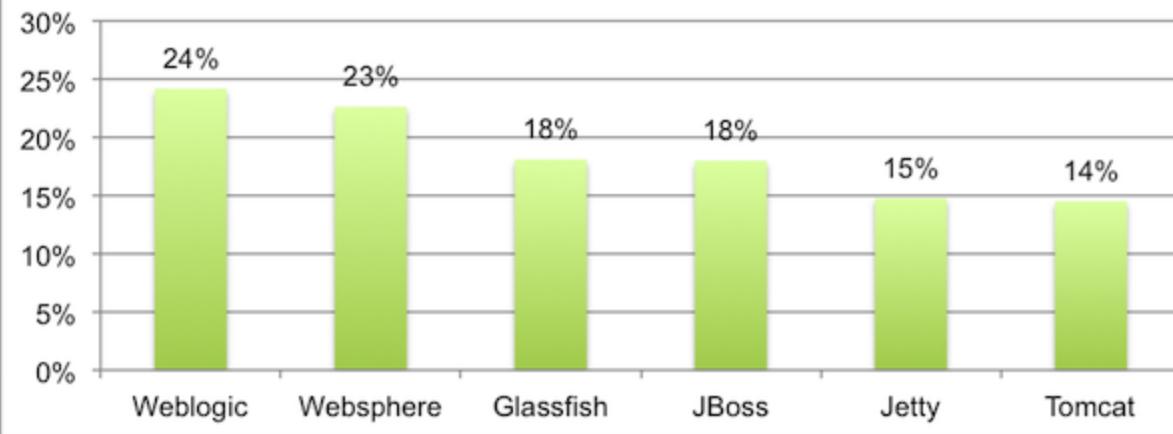


Productivity and Performance at Scale

You are a developer using Java to
build web services.

This is your life:

Percent of coding time spent redeploying



Waiting. [1]

```
network — jbrikman@ela4:/export/apps/cws-tomcat/i001/logs — ssh — 204x58
bash          jbrikman@ela4-a...ckend/i001/logs      bash          jbrikman@ela4-a...omcat/i001/logs
[jbrikman@ela4logs]$ tail -f catalina.out
May 28, 2013 7:06:28 AM org.apache.tomcat.util.http.Parameters processParameters
INFO: Invalid chunk starting at byte [207] and ending at byte [207] with a value of [null] ignored
2013/05/28 07:04:25.649 ERROR [ResponseGenerator] [http-12288-121] [cws] [GnchBVd4ILM0tfz6/vHEnv=] Uncaught exception
java.lang.reflect.InvocationTargetException
at sun.reflect.GeneratedMethodAccessor498.invoke(Unknown Source)
at com.linkedin.container.rpc.filter.FilterChainProxy.doinvoke(FilterChainProxy.java:58)
at org.xeril1.util.reflect.AbstractInvocationHandler.invoke(AbstractInvocationHandler.java:63)
at $Proxy21.read(Unknown Source)
at sun.reflect.GeneratedMethodAccessor68.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:597)
at org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:309)
at org.springframework.aop.framework.JdkDynamicAopProxy.invoke(JdkDynamicAopProxy.java:196)
at $Proxy20.read(Unknown Source)
at sun.reflect.GeneratedMethodAccessor62.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:597)
at org.springframework.remoting.support.RemoteInvocation.invoke(RemoteInvocation.java:285)
at org.springframework.remoting.support.DefaultRemoteInvocationExecutor.invoke(DefaultRemoteInvocationExecutor.java:38)
at org.springframework.remoting.support.RemoteInvocationBasedExporter.invoke(RemoteInvocationBasedExporter.java:78)
at org.springframework.remoting.support.RemoteInvocationBasedExporter.invokeAndCreateResult(RemoteInvocationBasedExporter.java:114)
at org.springframework.remoting.httpinvoker.HttpInvokerServiceExporter.handleRequest(HttpInvokerServiceExporter.java:74)
at com.linkedin.container.rpc.http.RpcServlet.processRequest(RpcServlet.java:217)
at com.linkedin.container.rpc.http.RpcServlet.service(RpcServlet.java:139)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:820)
at org.mortbay.jetty.servlet.ServletHolder.handle(ServletHolder.java:511)
at org.mortbay.jetty.servlet.ServletHandler.handle(ServletHandler.java:401)
at org.mortbay.jetty.security.SecurityHandler.handle(SecurityHandler.java:216)
at org.mortbay.jetty.servlet.SessionHandler.handle(SessionHandler.java:182)
at org.mortbay.jetty.handler.ContextHandler.handle(ContextHandler.java:766)
at org.mortbay.jetty.webapp.WebAppContext.handle(WebAppContext.java:450)
at com.linkedin.enveb.ContextBasedHandlerImpl.handle(ContextBasedHandlerImpl.java:87)
at com.linkedin.enveb.MapBasedHandlerImpl.handle(MapBasedHandlerImpl.java:453)
at org.mortbay.jetty.handler.HandlerCollection.handle(HandlerCollection.java:114)
at org.mortbay.jetty.handler.HandlerWrapper.handle(HandlerWrapper.java:152)
at org.mortbay.jetty.Server.handle(Server.java:326)
at org.mortbay.jetty.HttpConnection.handleRequest(HttpConnection.java:542)
at org.mortbay.jetty.HttpConnection$RequestHandler.content(HttpConnection.java:945)
at org.mortbay.jetty.HttpParser.parseNext(HttpParser.java:756)
at org.mortbay.jetty.HttpParser.parseAvailable(HttpParser.java:218)
at org.mortbay.jetty.HttpConnection.handle(HttpConnection.java:404)
at java.lang.reflect.Method.invoke(Method.java:597)
... 19 more
Caused by: java.lang.NullPointerException
at org.springframework.remoting.support.RemoteInvocationUtils.fillInClientStackTraceIfPossible(RemoteInvocationUtils.java:47)
at org.springframework.remoting.support.RemoteInvocationResult.recreate(RemoteInvocationResult.java:115)
at org.springframework.remoting.support.RemoteInvocationBasedAccessor.recreateRemoteInvocationResult(RemoteInvocationBasedAccessor.java:85)
at org.springframework.remoting.httpinvoker.HttpInvokerClientInterceptor.invoke(HttpInvokerClientInterceptor.java:148)
at com.linkedin.container.rpc.RpcServiceImporter.invoke(RpcServiceImporter.java:148)
at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:172)
at org.springframework.aop.framework.JdkDynamicAopProxy.invoke(JdkDynamicAopProxy.java:202)
at $Proxy97.read(Unknown Source)
at sun.reflect.GeneratedMethodAccessor326.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:597)
at com.linkedin.util.MethodInvocationTrackerProxy.invoke(MethodInvocationTrackerProxy.java:104)
at $Proxy98.read(Unknown Source)
```

tail -f logs/catalina.out

filters.spring — frontier

```

<assembly.spring> <contentProvider.spring> <component.spring> <entitlements-lix.spring> <url-manager.properties> <url-manager.spring> <template.spring>

```

```

<bean id="frontierFacadeFactory" class="com.linkedin.frontier.filters.FrontierFacadeFactory">
    <property name="urlEncryptor" ref="urlEncryptor"/>
    <property name="urlManager" ref="urlManager"/>
    <property name="pageKeyContext" ref="pageKeyContext"/>
    <property name="mobileAgentDetector" ref="mobileAgentDetector"/>
    <property name="deviceDetector" ref="deviceDetector"/>
    <property name="client" ref="client"/>
    <property name="lixClient" ref="shadowLixClient"/>
</bean>
<lin:bean id="filter.requestFacade" class="com.linkedin.frontier.filters.FrontierRequestFacadeFilter">
    <property name="facadeFactory" ref="frontierFacadeFactory" />
    <property name="frontierContextConfig" ref="frontierContextConfig"/>
</lin:bean>
<lin:bean id="icFinder" class="com.linkedin.container.ic.impl.ICFinderBeanFactory"/>
<lin:bean id="filter.icInitFilter" class="com.linkedin.container.ic.filter.ICInitializeFilter">
    <property name="icFinder" ref="icFinder" />
</lin:bean>
<lin:bean id="filter.cookiesToICFilter" class="com.linkedin.container.ic.filter.CookiesToICFilter">
    <property name="icFinder" ref="icFinder" />
    <property name="isEnabled" value="#{filter.cookiesToICFilter.isEnabled}" />
</lin:bean>
<lin:bean id="filter.requestSecurityManager" class="org.xeril.wafwk.gui.core.RequestSecurityManagerFilter">
    <property name="requestSecurityManager" ref="requestSecurityManager"/>
</lin:bean>
<bean id="filter.uniqueCookieValue" class="org.xeril.wafwk.gui.core.filters.UniqueCookieValueFilter">
</bean>
<lin:bean id="filter.trackingGUIDFilter" class="com.linkedin.tracking.ic.filter.TrackingGUIDServletFilter">
    <property name="guidAccessor" ref="trackingGuidAccessor"/>
    <property name="key" value="#{webtrack.guid.tracking.key}"/>
</lin:bean>
<lin:bean id="trackingGuidAccessor" class="com.linkedin.tracking.client.impl.ic.InvocationContextTrackingGUIDAccessor">
    <property name="icFinder" ref="icFinder" />
</lin:bean>
<lin:bean id="realmCacheKeyProvider" class="com.linkedin.cache.impl.RealmCacheKeyProviderPrefixImpl">
    <property name="realmIDLen" value="0"/>
</lin:bean>
<bean id="allowedAuthTokenType" class="org.springframework.beans.factory.config.SetFactoryBean">
    <property name="sourceSet">
        <set>
            <value>GST</value>
            <value>LTM</value>
            <value>CAP</value>
            <value>CSM</value>
            <value>OAU</value>
        </set>
    </property>
</bean>
<lin:bean id="linkedInAuthService" class="com.linkedin.domain.authrealm.pub.linkedin.LinkedInAuthServiceImpl">
    <property name="authService" ref="authService"/>
    <property name="origin" ref="${com.linkedin.app.name}"/>

```

Line: 108 Column: 1 XML Soft Tabs: 2

XML soup

`org.springframework.aop.framework`

Class AbstractSingletonProxyFactoryBean

`java.lang.Object`

 └ `org.springframework.aop.framework.ProxyConfig`

 └ `org.springframework.aop.framework.AbstractSingletonProxyFactoryBean`

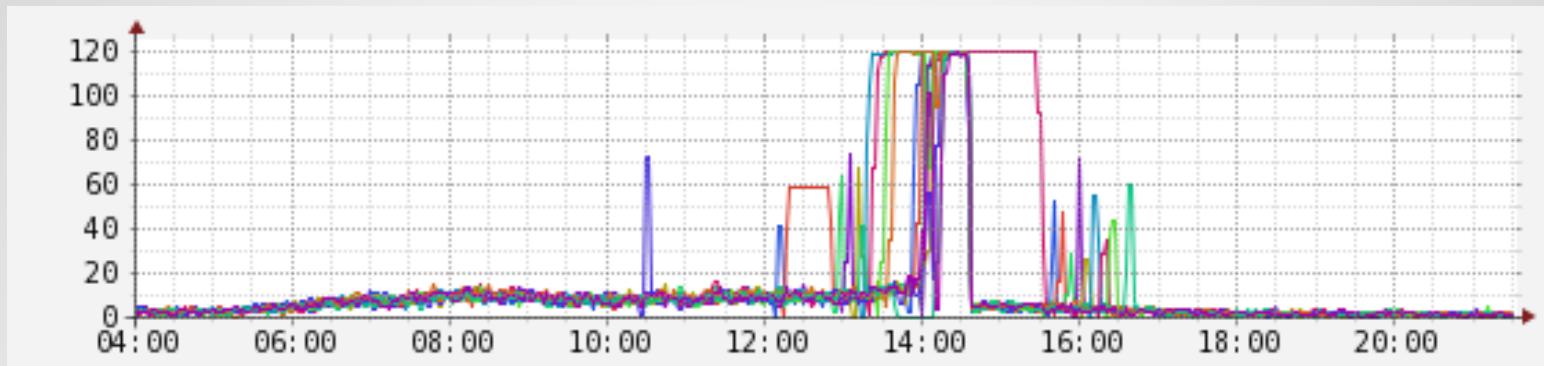
`public abstract class AbstractSingletonProxyFactoryBean`

`extends ProxyConfig`

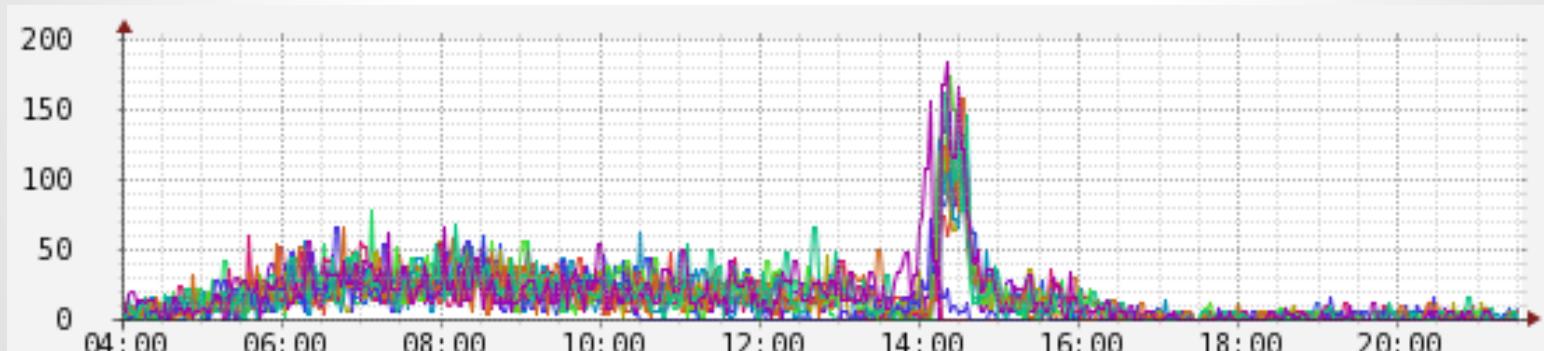
`implements FactoryBean, BeanClassLoaderAware, InitializingBean`

Convenient proxy factory bean superclass for proxy factory beans that create only singletons.

"Convenient proxy factory bean superclass for proxy factory beans that create only singletons." [2]

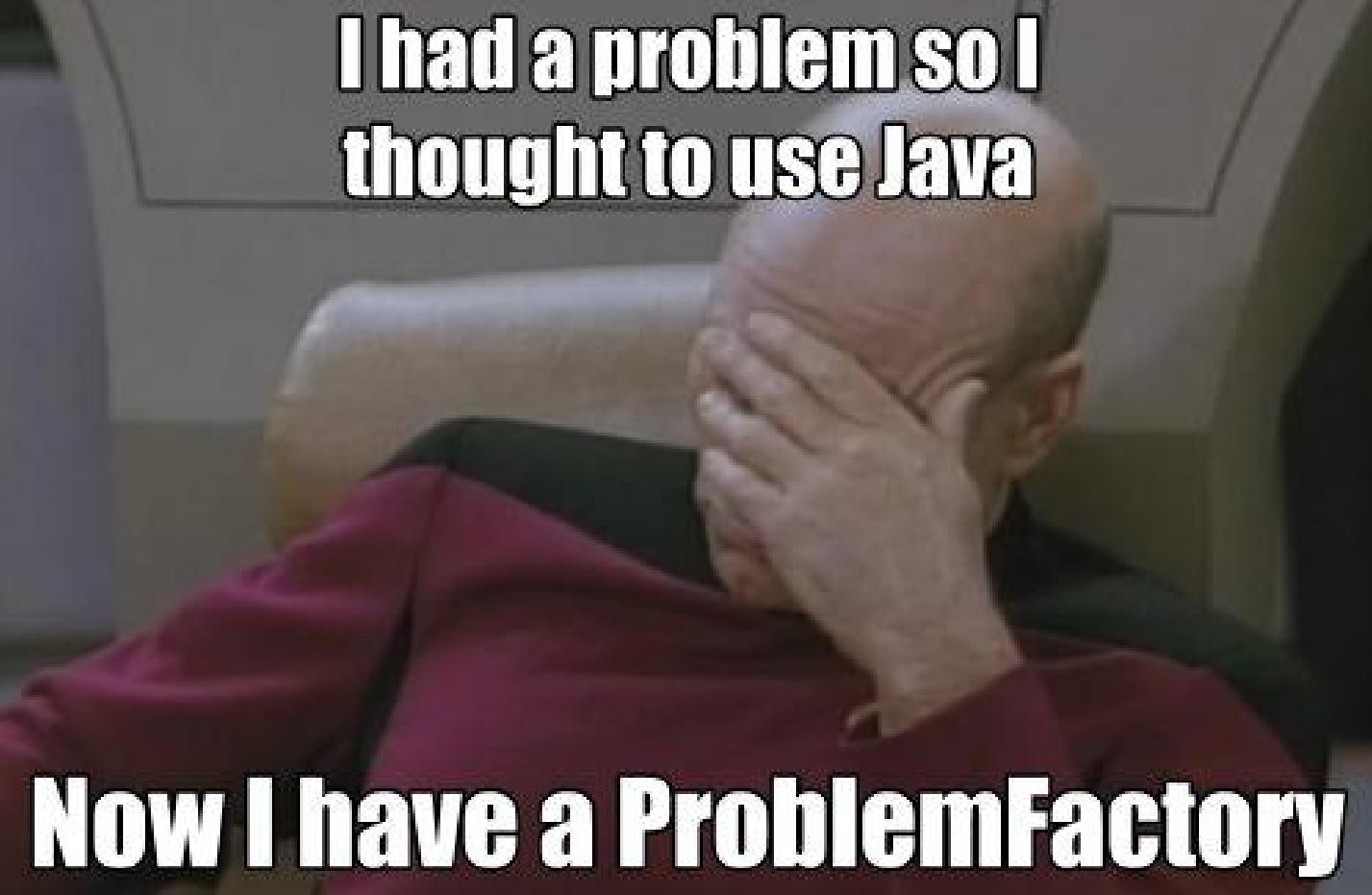


Thread pool usage



Latency

Thread pool hell



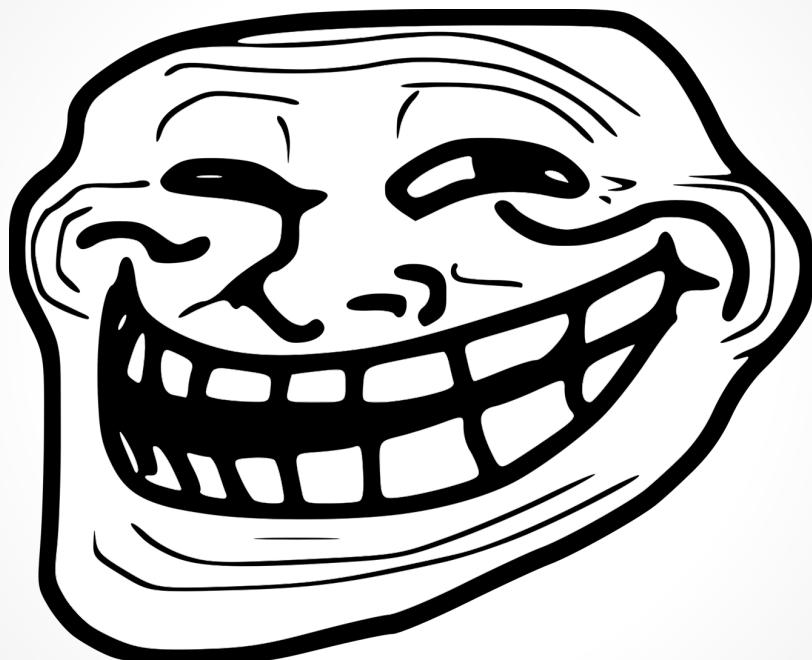
**I had a problem so I
thought to use Java**

Now I have a ProblemFactory

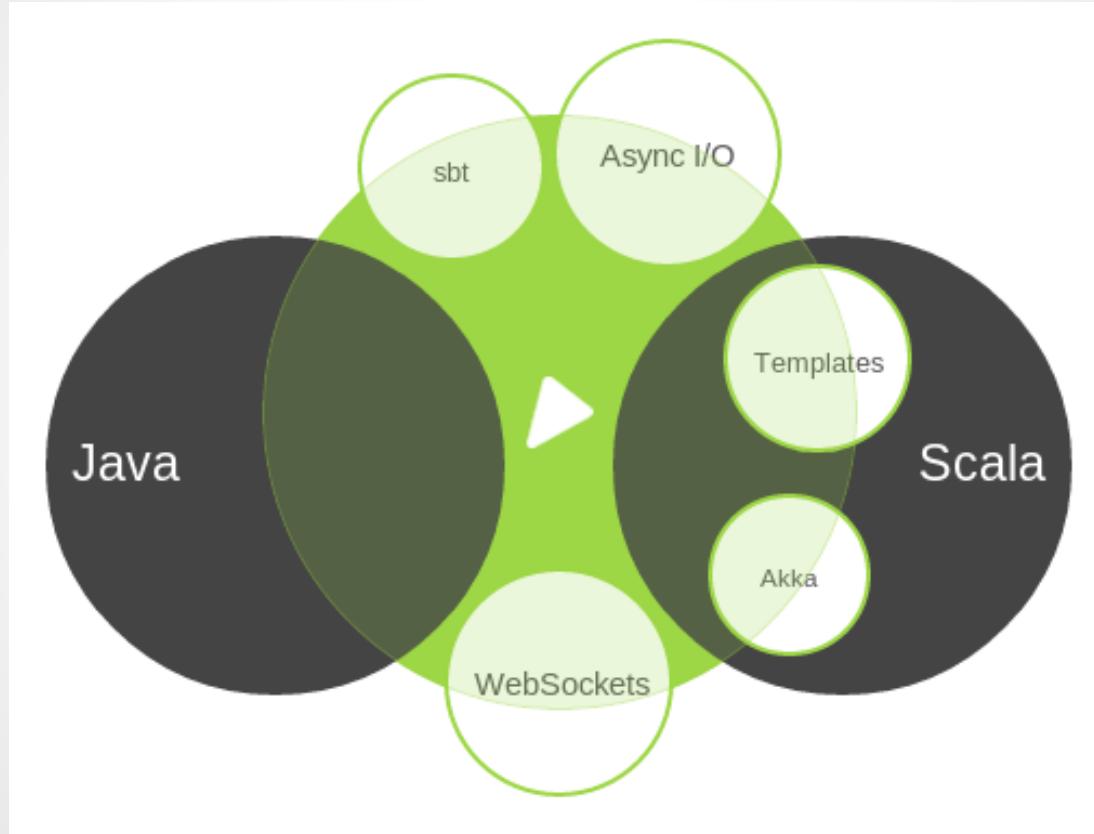
There is a better way.



Ruby on Rails!



Nah, just kidding. We have real work to do.



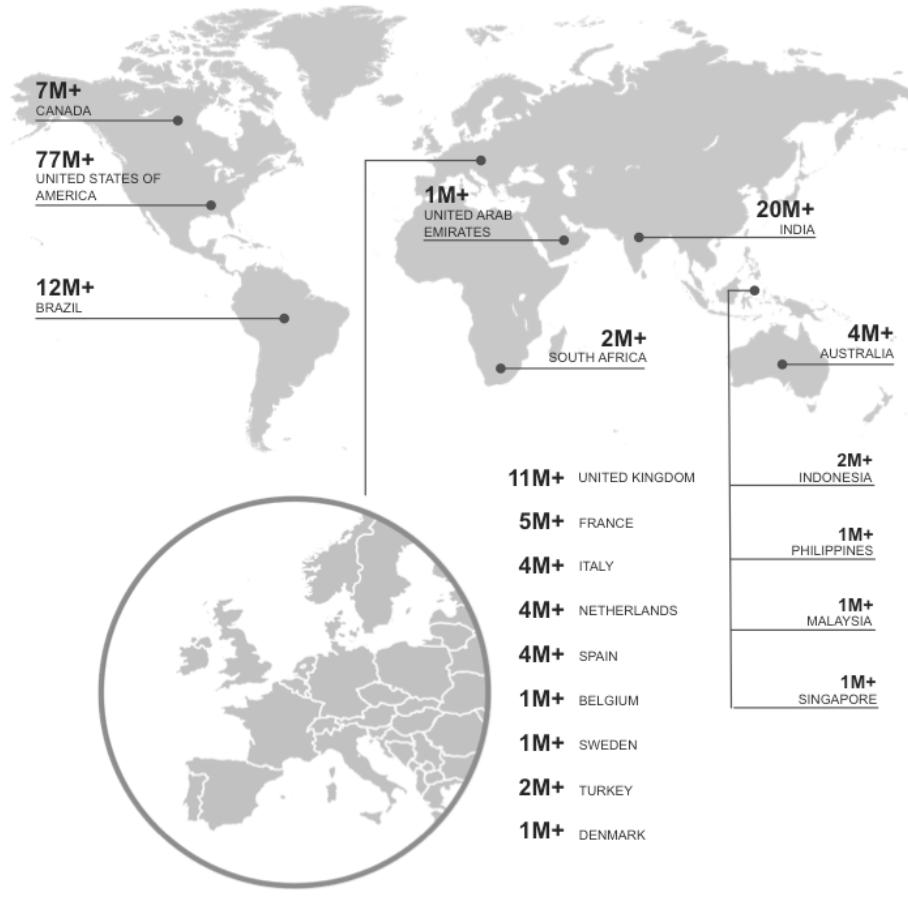
A modern web framework for Java and Scala



The world's largest professional network

225,000,000+

REGISTERED MEMBERS





at



We've been using Play in production for
more than 6 months [3]

A few apps built on Play

LinkedIn | Higher Education

https://www.linkedin.com/channels/education?trk=cha-other-cha

Search... Advanced

Home Profile Network Jobs Interests Premium Solutions Upgrade

Higher Education 208,358 followers ✓ Following



Michael Fertik, CEO at Reputation.com and Owner, Reputation.com
+ Follow (10,981)

Please Think Twice Before Getting Your Master's · May 28, 2013

124,783 450 495 2,277

Recently Posted

Most Recent | Most Popular

WSJ Wall Street Journal

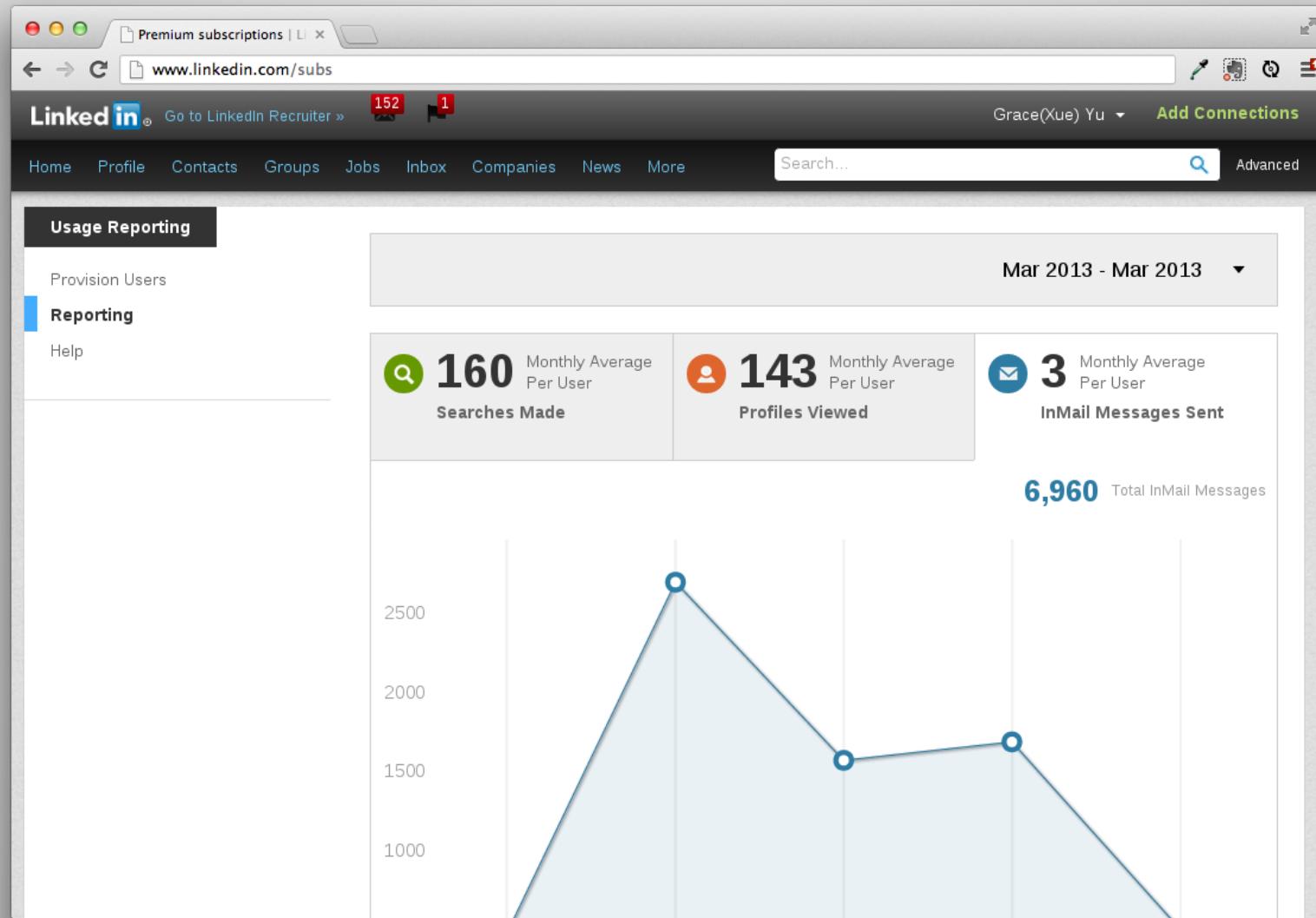
The Real Message for the Class of 2013

wsj.com Simpsons writer Rob LaZebnik imagines a commencement address: You're pampered, privileged and oversexed—but at least your employment prospects are dim.

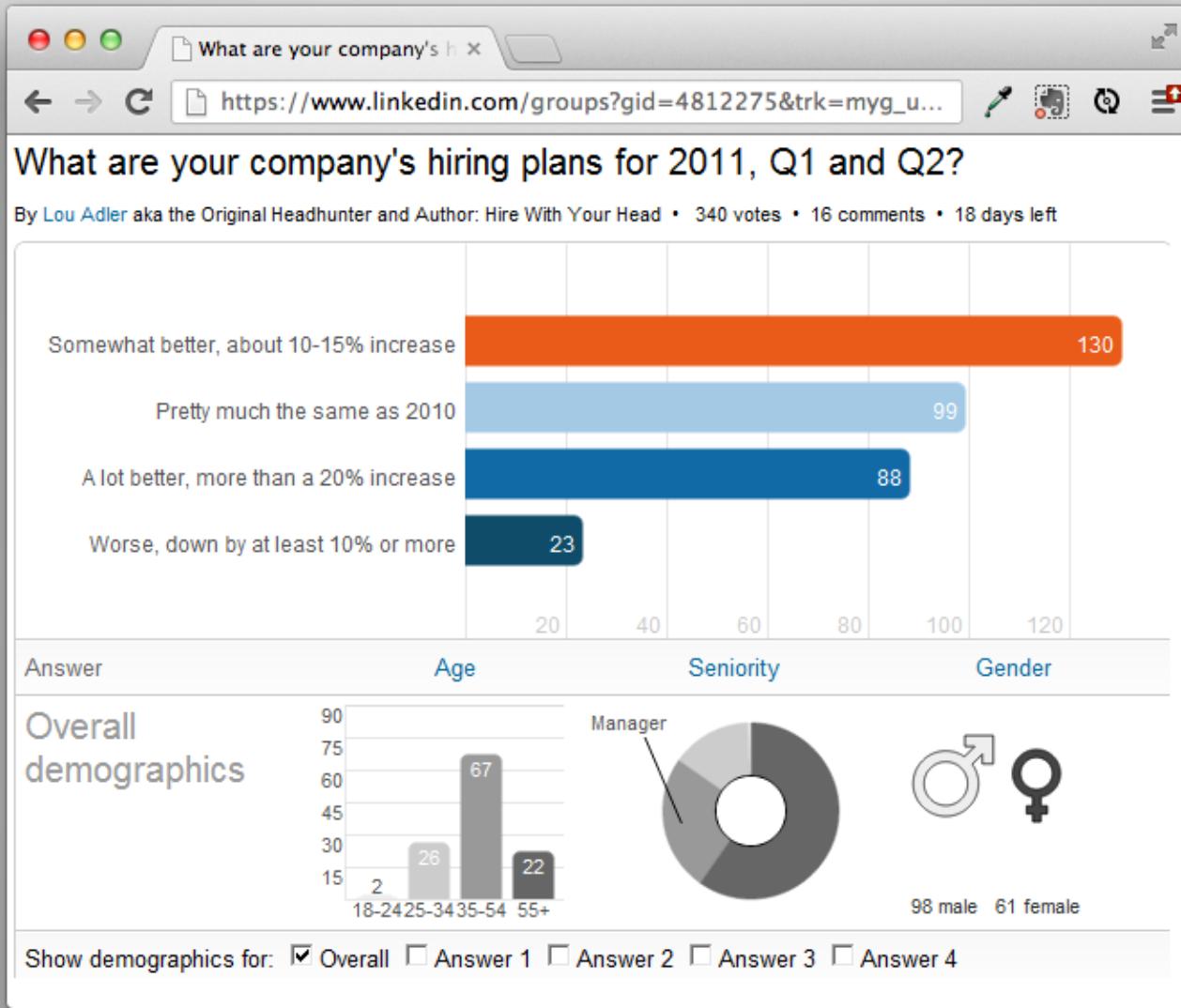
Other Channels

Editor's Picks Marketing Strategies Social Media Technology Best Advice Big Ideas & Innovation Business Travel Customer Service

Channels (frontend)



Premium Subscriptions (frontend)



Polls (frontend + backend)

REST API Search: companies

<https://www.linkedin-ei2.com/rest-search-frontend/apis/BizProfile/companies>

Rest.li Search Rest.li Docs

BizProfile Cluster / companies Resource

/companies

BizCompany collection

Key:
companiesId: long

Supports:
create get update batch_get

Finders:
admin email name

Actions:
getCompanyIdsFromEmailDomains areEmployeesOfCompany

Resource Documentation

Deco Link

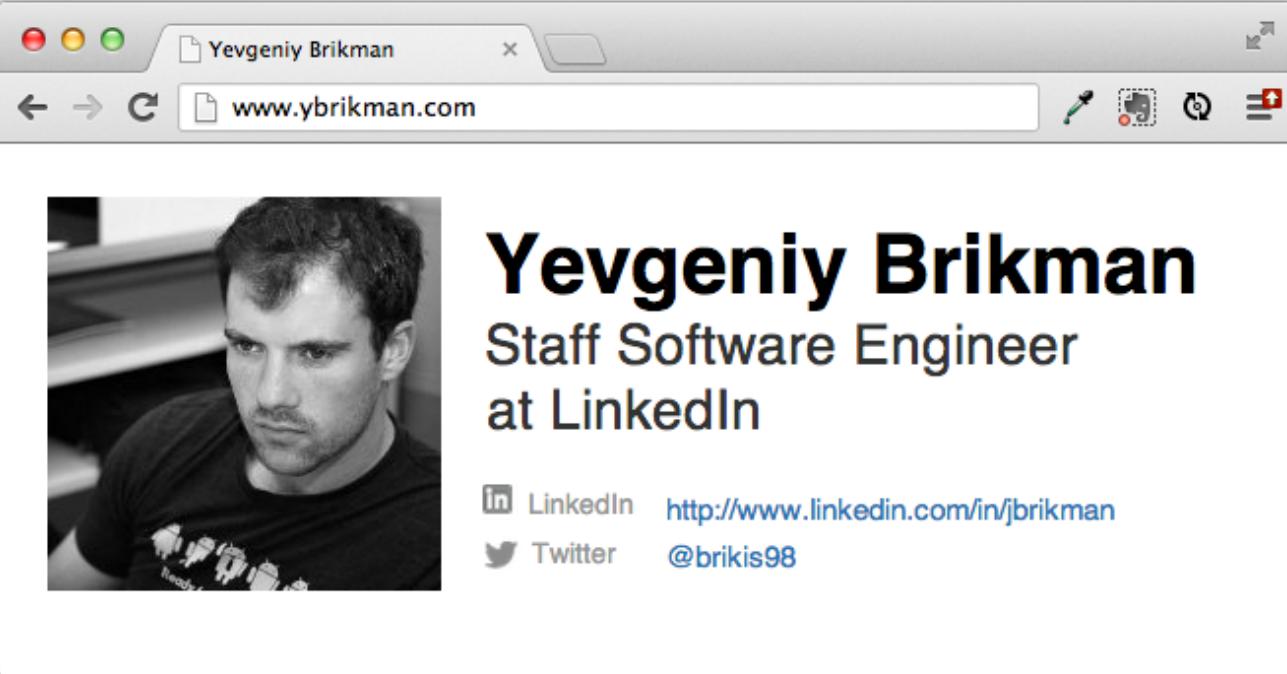
BizCompany Schema

Company Profile

Field Name	Type	Description
active	boolean, default="true"	
administrators	optional Urn[]	
allEmployeesAsAdmins	boolean, default="false"	
attributes	BizCompanyAttributes	This field contains boolean attributes represented in the 'attributes' bitmap of a

REST search (internal tool)

About me

A screenshot of a web browser window. The title bar says "Yevgeniy Brikman". The address bar shows the URL "www.ybrikman.com". The main content area features a black and white portrait photo of a man with dark hair and a beard, looking slightly to the left. To the right of the photo, the text reads "Yevgeniy Brikman" in large bold letters, followed by "Staff Software Engineer" and "at LinkedIn". Below this, there are two lines of social media links: "LinkedIn" with the URL "http://www.linkedin.com/in/jbrikman" and "Twitter" with the handle "@brikis98".

Yevgeniy Brikman
Staff Software Engineer
at LinkedIn

LinkedIn <http://www.linkedin.com/in/jbrikman>
Twitter @brikis98

Leading the Play project as part of LinkedIn's Service Infrastructure Team.
Also: hackdays, engineering blog, incubator, open source.

This talk is the story of building web services
at massive scale with Java and Scala...

... while maintaining performance,
reliability, and developer
productivity.

Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. Community

Outline

1. **Getting started with Play**
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. Community

A screenshot of a web browser window displaying the Play Framework download page at www.playframework.com/download. The page has a green header with the word "play" and navigation links for "Download", "Documentation", and "Get Involved". Below the header, there's a large green section titled "Download Play Framework" with social sharing icons. A central call-to-action button says "Latest official version" with a download icon, linking to "play-2.1.1.zip" from April 3, 2013, which is 145.2M in size. To the right, there are four numbered sections: "First steps" (with links to installation instructions, first app, and documentation), "Find help" (with links to mailing list, Stackoverflow, and professional support), "Contribute" (with links to code/contributors, get involved, and security reports), and "Try it live in the cloud" (with a note about CloudBees support). At the bottom left, there's a "Follow us for release announcements" button with a Twitter icon.

Download Play Framework

play

Download Documentation Get Involved

Ready to use packages for Linux, MacOS and Windows

Latest official version

play-2.1.1.zip Apr 03 2013 145.2M

Follow us for release announcements

Development versions

Previews of releases in the pipeline.

play-2.1.2-RC1.zip	May 16 2013	145.5M
--------------------	-------------	--------

Previous versions

Looking for older versions of Play? You can find some of our older releases below.

play-2.1.0.zip	Feb 06 2013	144.3M
play-2.0.4.zip	Oct 01 2012	96.6M

① First steps

Installation instructions
Build your first app
Full documentation

② Find help

Mailing list
Stackoverflow
Professional support

③ Contribute

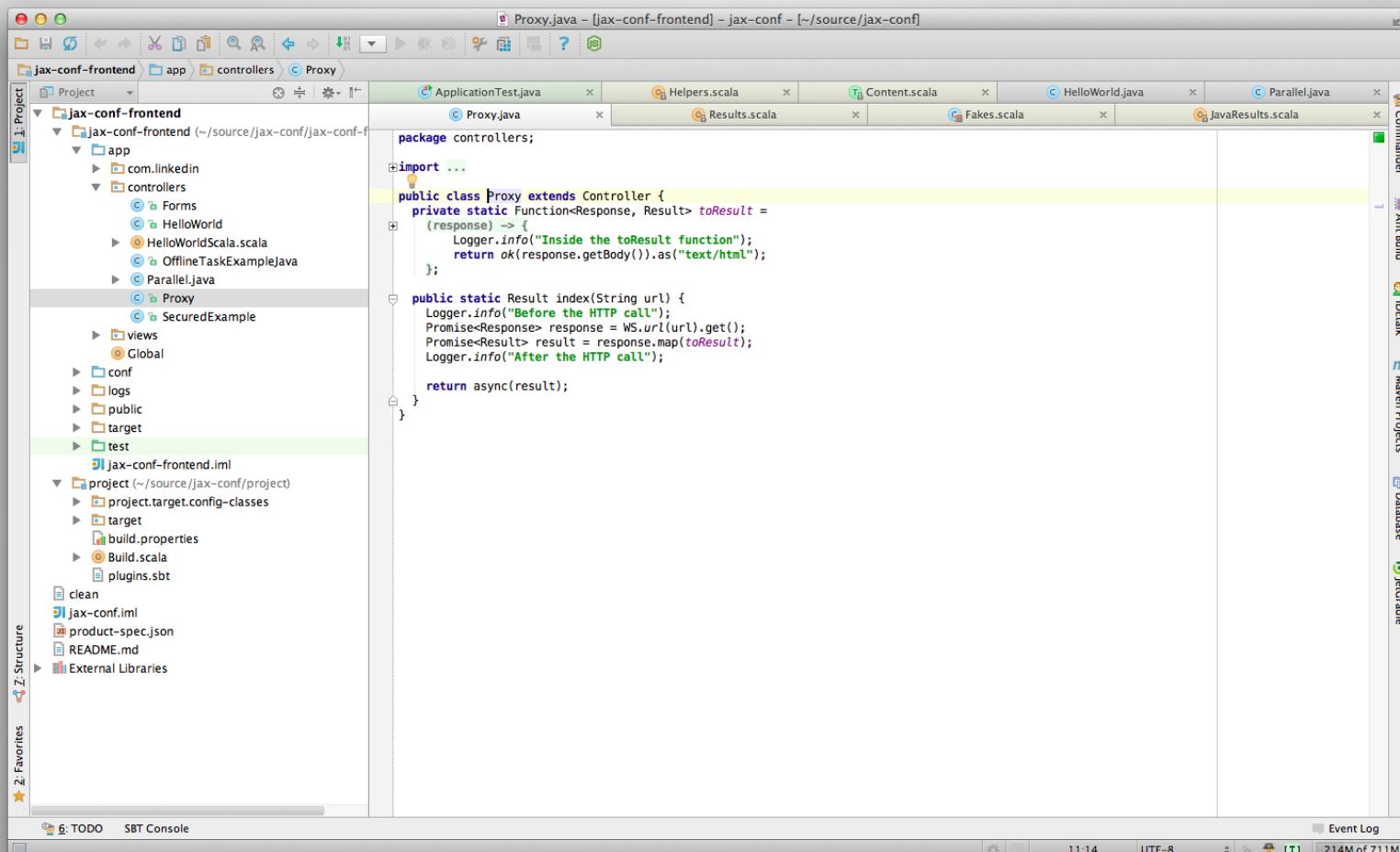
Code and contributors
Get involved
Security reports

④ Try it live in the cloud

CloudBees includes first-class support for running Play applications in the Cloud.

Download and install Play from
<http://www.playframework.com>

```
> play new my-app
```



> play idea
> play eclipse

```
jbrikman-mn:my-app jbrikman$ play run
[info] Loading project definition from /Users/jbrikman/source/my-app/project
[info] Set current project to my-app (in build file:/Users/jbrikman/source/my-app/)
--- (Running the application from SBT, auto-reloading is enabled) ---
[info] play - Listening for HTTP on /0:0:0:0:0:0:0:0%0:9000
(Server started, use Ctrl+D to stop and go back to the console...)

```

> play run

The screenshot shows a web browser window with the title "Welcome to Play 2.0". The address bar displays "localhost:9000". The main content area has a green header with the text "Your new application is ready." and a "Browse APIs" button. Below this, a message通知关于Play框架2.0.3发布的消息。接着是"Welcome to Play 2.0"的标题，下方有一段恭喜用户创建新应用的文字。一个黄色的提示框显示"Your are using Play 2.0.3"。随后是一个关于路由配置的段落，提到了conf/routes文件和Application.index动作。代码示例展示了如何在conf/routes文件中定义一个GET请求到根路径的规则。接着说明了控制器方法controllers.Application.index被调用。之后，通过一段文字和代码示例解释了如何处理HTTP请求并返回结果。最后，提到模板是在app/views/index.scala.html文件中定义的。

Welcome to Play 2.0

Your new application is ready.

Play framework 2.0.3 Final is out! Download it [here](#).

Welcome to Play 2.0

Congratulations, you've just created a new Play application. This page will help you in the few next steps.

Your are using Play 2.0.3

Why do you see this page?

The `conf/routes` file defines a route that tells Play to invoke the `Application.index` action whenever a browser requests the `/` URI using the GET method:

```
# Home page
GET      /           controllers.Application.index()
```

So Play has invoked the `controllers.Application.index` method:

```
public static Result index() = {
    return ok(index.render("Your new application is ready."));
}
```

An action method handles the incoming HTTP request, and returns the HTTP result to send back to the Web client. Here we send a `200 OK` response, using a template to fill its content.

The template is defined in the `app/views/index.scala.html` file and compiled as a standard Java class.

<http://localhost:9000>

app	→ Application sources
└ assets	→ Compiled asset sources
└ controllers	→ Application controllers
└ models	→ Application business layer
└ views	→ Templates
conf	→ Configurations files
└ application.conf	→ Main configuration file
└ routes	→ Routes definition
public	→ Public assets
└ stylesheets	→ CSS files
└ javascripts	→ Javascript files
└ images	→ Image files
project	→ sbt configuration files
└ Build.scala	→ Application build script
└ plugins.sbt	→ sbt plugins
lib	→ Unmanaged libraries
dependencies	
logs	→ Standard logs folder
target	→ Generated stuff
test	→ Unit or functional tests

Application layout

Outline

1. Getting started with Play
2. **Make a change and reload**
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. Community

```
public class HelloWorld extends Controller {  
  
    public static Result index() {  
        return ok("Hello World");  
    }  
}
```

app/controllers/HelloWorld.java

Create a new controller and action

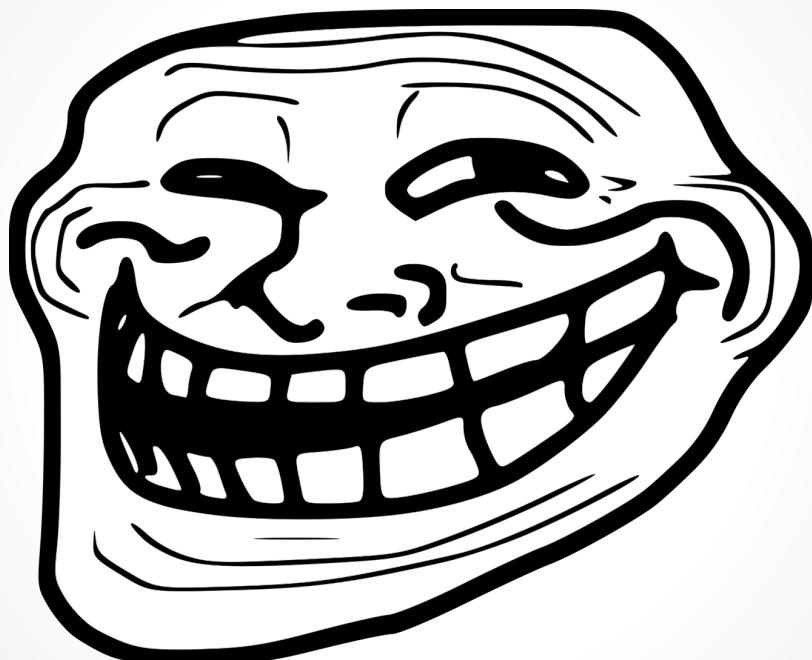
Don't worry about the use of static. Yes, Play supports IOC. Using static (and other shortcuts) lets me keep the examples simple.

```
GET      /hello      controllers.HelloWorld.index()
```

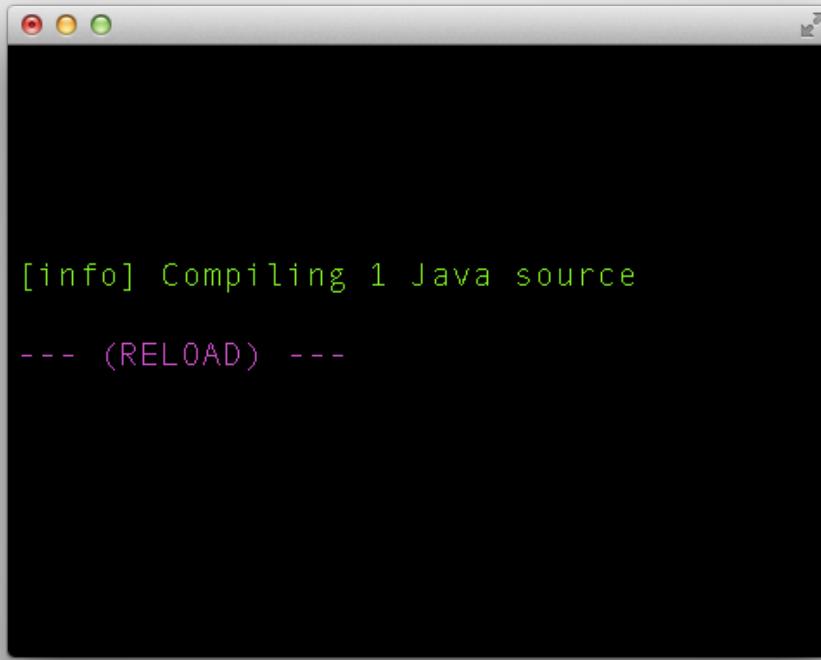
conf/routes

Expose the controller/action at a URL

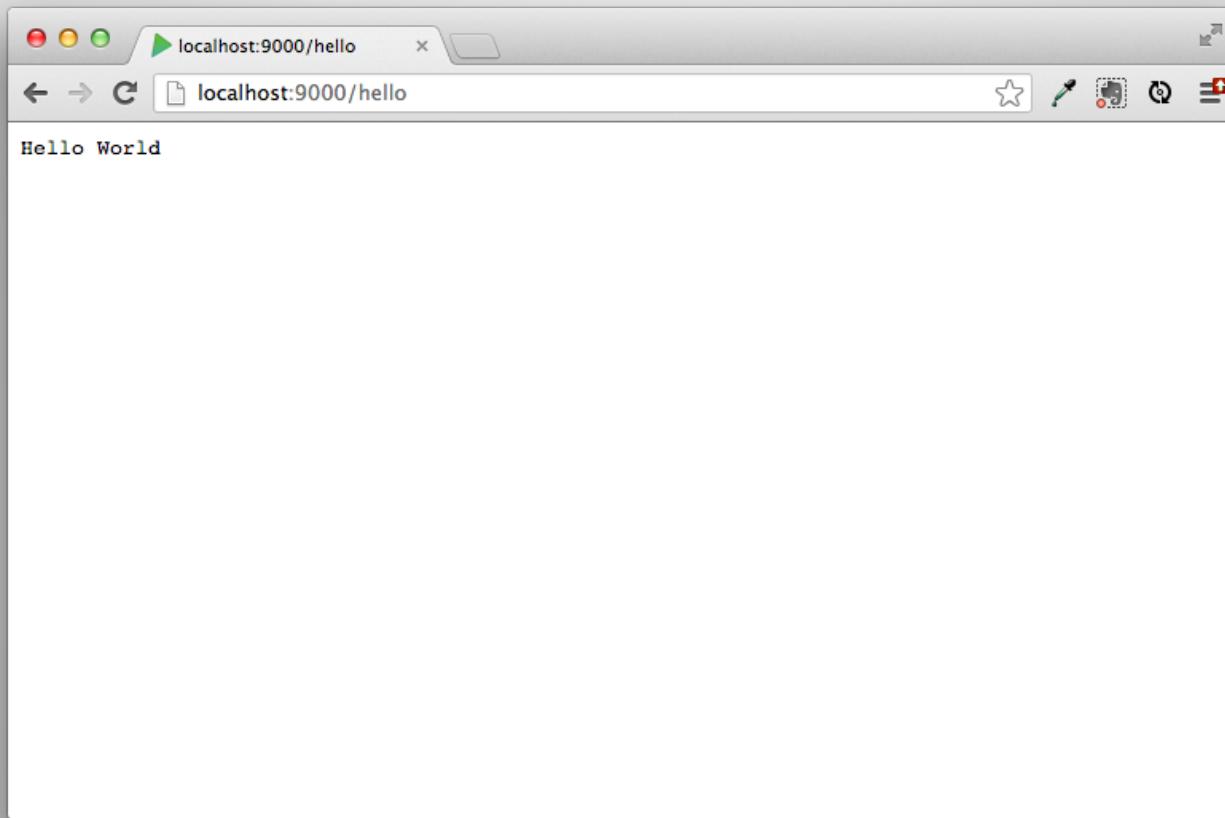
Now, restart the server.



Nah, just kidding. Refresh the page.



Woohoo, hot reload!



<http://localhost:9000/hello>

```
public class HelloWorld extends Controller {  
  
    public static Result index(String name) {  
        return ok("Hello " + name);  
    }  
}
```

app/controllers/HelloWorld.java

Add a parameter

```
GET /hello controllers.HelloWorld.index( name)
```

conf/routes

Read the parameter from the query string

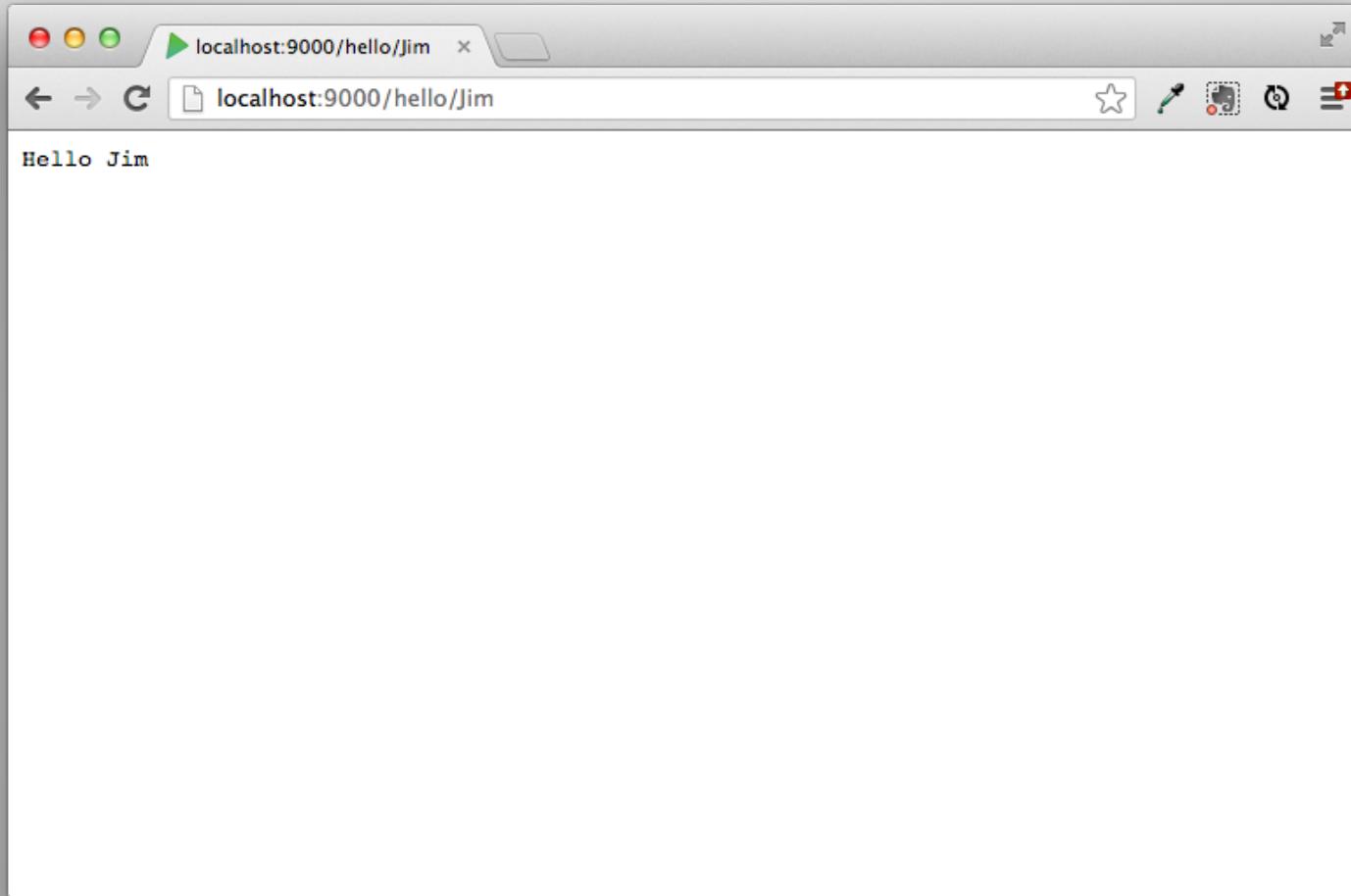


<http://localhost:9000/hello?name=Jim>

```
GET /hello/:name controllers.HelloWorld.index(name)
```

conf/routes

Read the parameter from the URL instead



<http://localhost:9000/hello/Jim>

```
public class HelloWorld extends Controller {  
  
    public static Result index(String name, int age) {  
        return ok("Hello " + name + " you are " + age +  
                 " years old");  
    }  
}
```

app/controllers/HelloWorld.java

Add another parameter, this time an int

```
GET /hello/:name/ :age controllers.HelloWorld.index(name: String, age: Int)
```

conf/routes

Add the parameter. Note the type checking!



<http://localhost:9000/hello/Jim/28>

```
@(name: String, age: Int)

<html>
  <head></head>
  <body>
    
    <p>
      Hello <b>@name</b>, you are <b>@age</b> years old
    </p>
  </body>
</html>
```

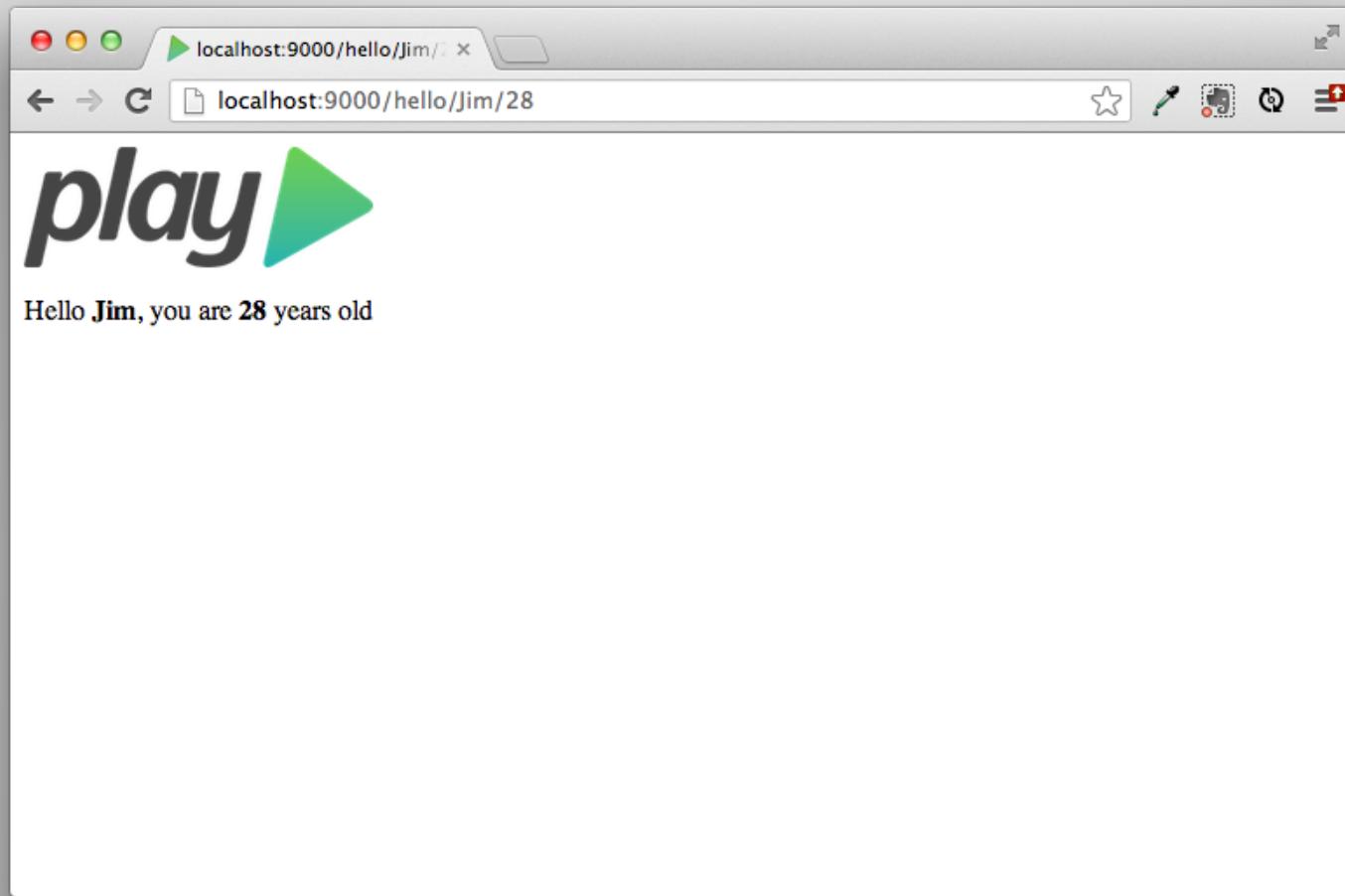
app/views/hello.scala.html

Add a view

```
public class HelloWorld extends Controller {  
  
    public static Result index(String name, int age) {  
        return ok(views.html.hello.render(name, age));  
    }  
}
```

app/controllers/HelloWorld.java

Render the view from the controller



<http://localhost:9000/hello/Jim/28>

```
location = "JaxConf"
```

app/conf/application.conf

How about some config?

```
public class HelloWorld extends Controller {  
  
    public static Result index(String name, int age) {  
        String location = getConfig().getString("location");  
        return ok(views.html.hello.render(name, age,  
location));  
    }  
  
    private static Configuration getConfig() {  
        return Play.application().configuration();  
    }  
}
```

app/controllers/HelloWorld.java

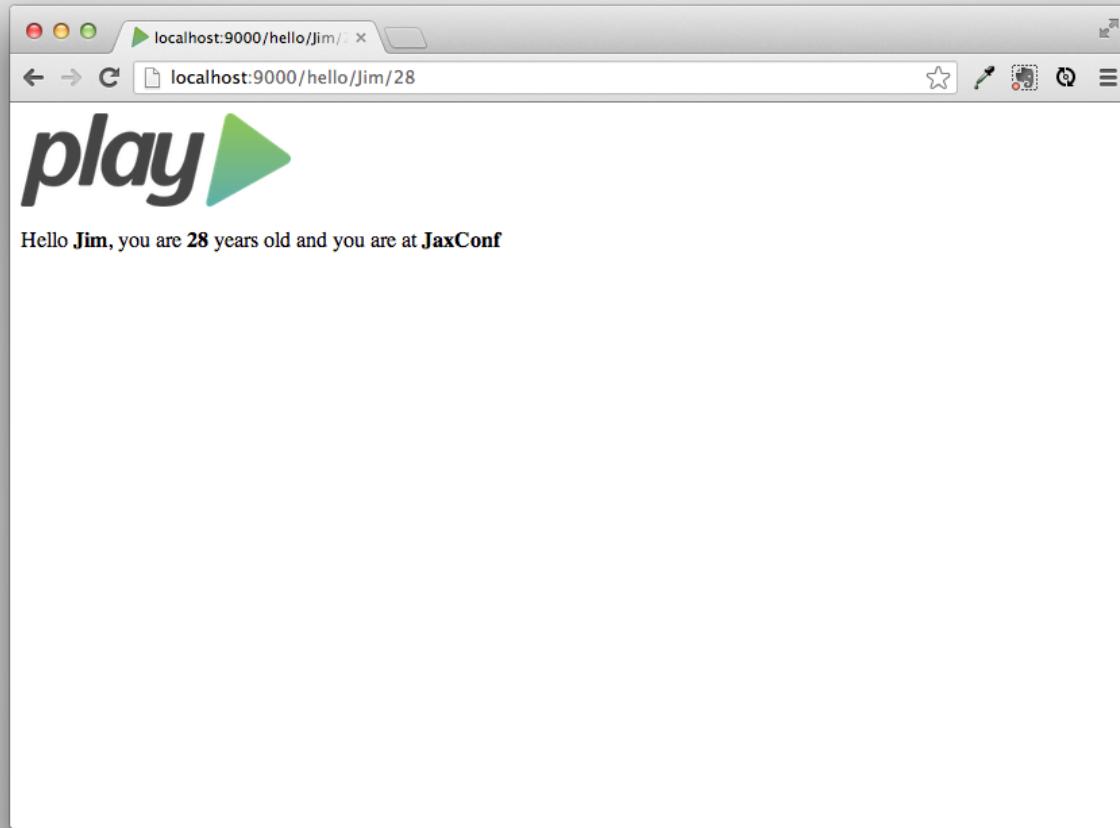
Read the config and pass it to the view

```
@(name: String, age: Int, location: String)

<html>
  <head></head>
  <body>
    
    <p>
      Hello <b>@name</b>, you are <b>@age</b> years old
      and you are at <b>@location</b>
    </p>
  </body>
</html>
```

app/views/hello.scala.html

Update the view

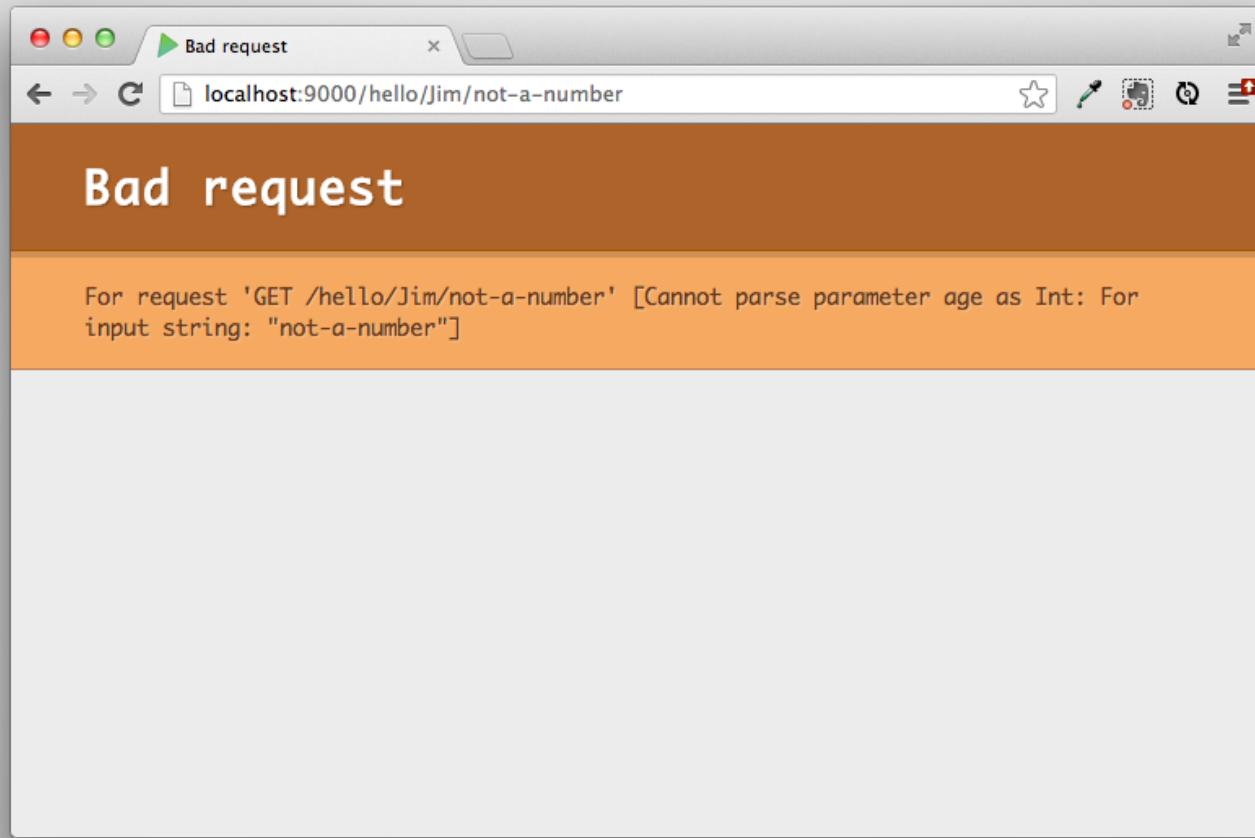


Refresh the page. Config hot reloads too!

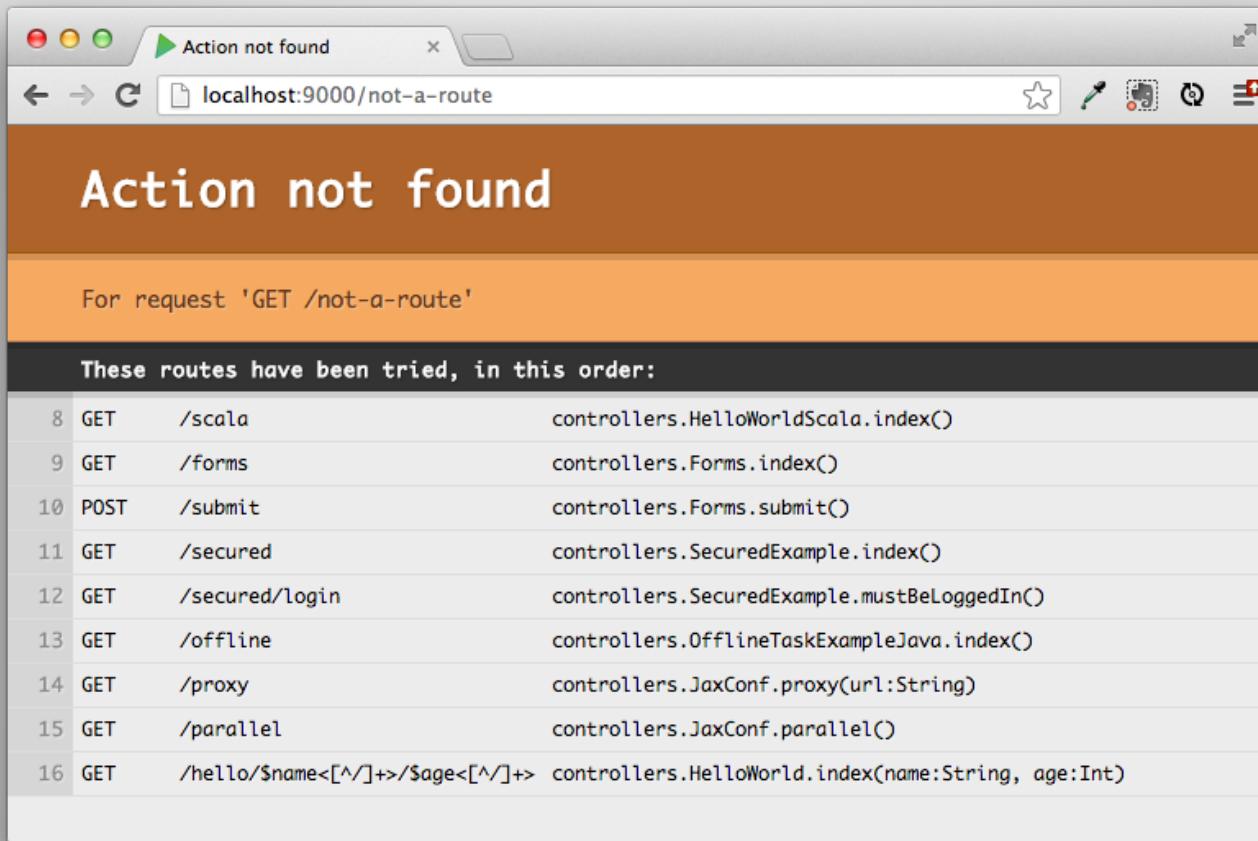
Outline

1. Getting started with Play
2. Make a change and reload
3. **Error handling**
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. Community

In dev mode, Play shows error messages
right in the browser.



Parameter type checking



A helpful 404 page

A screenshot of a web browser window titled "Compilation error". The address bar shows "localhost:9000/hello/Jim/28". The main content area has a red header with the text "Compilation error". Below this, a pink section displays the error message "';' expected". A black section below it provides context: "In /Users/jbrikman/source/jax-conf/jax-conf-frontend/app/controllers/HelloWorld.java at line 10." The source code is listed in a monospaced font:

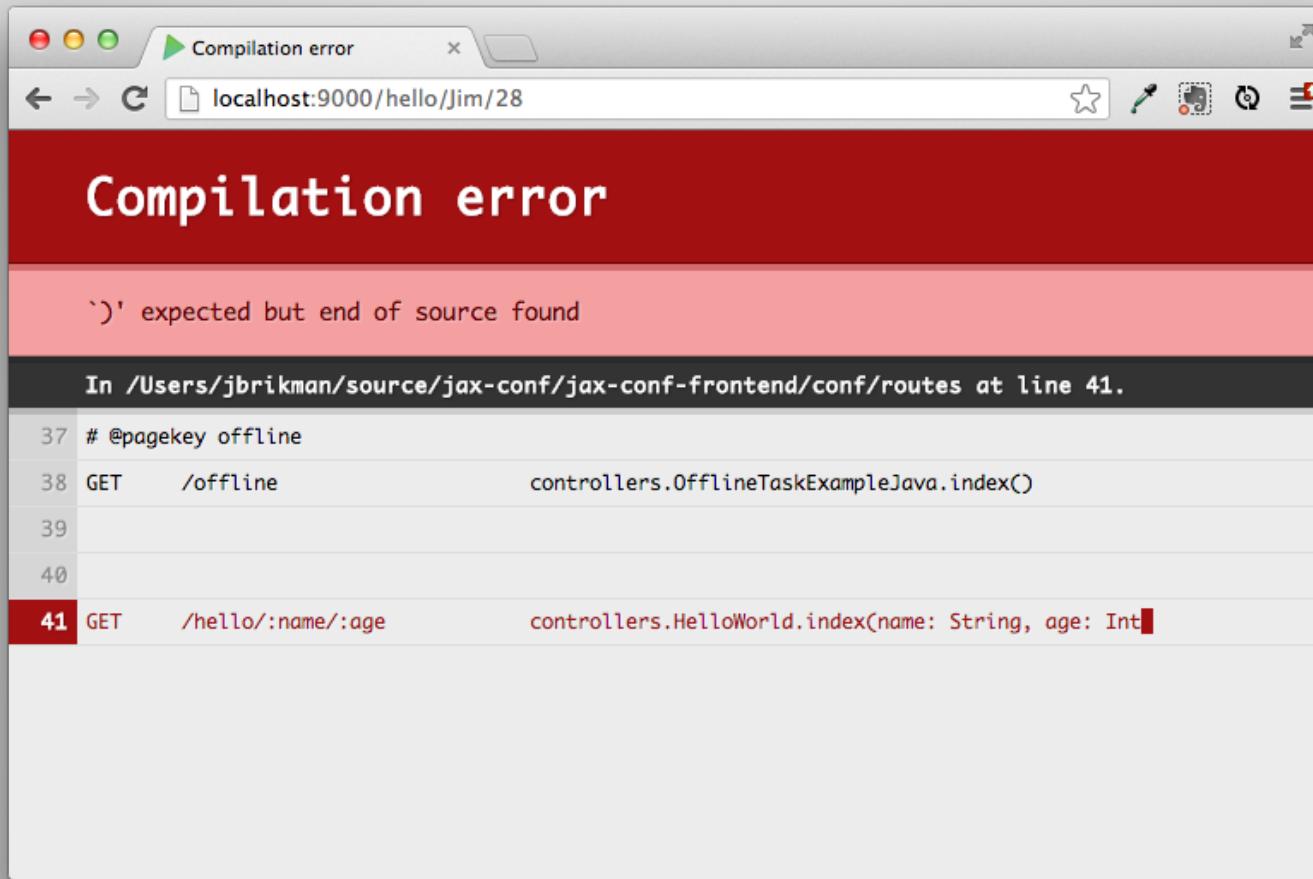
```
6 public class HelloWorld extends Controller  
7 {  
8     public static Result index(String name, int age)  
9     {  
10        return ok("Hello " + name + " you are " + age + " years old")  
11    }  
12 }
```

Compile errors show problematic source code
in the browser

A screenshot of a web browser window titled "Compilation error". The address bar shows "localhost:9000/hello/Jim/28". The main content area has a red header with the text "Compilation error". Below it, a pink section displays the error message "not found: value typo". A black section below that provides context: "In /Users/jbrikman/source/jax-conf/jax-conf-frontend/app/views/hello.scala.html at line 8.". The code editor shows the following Scala template code:

```
4 <head></head>
5 <body>
6   
7   <p>
8     Hello <b>@typo</b>, you are <b>@age</b> years old
9   </p>
10  </body>
11 </html>
```

Views are compiled as well



The routes file too

Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. **Threaded vs. evented**
5. Non-blocking I/O
6. Scala
7. Performance
8. Community



Most people are used to threaded servers

```
void doGet(HttpServletRequest req, HttpServletResponse res) {  
    // Apache HttpClient  
    HttpClient client = new HttpClient();  
    GetMethod method = new GetMethod("www.example.com/");  
  
    // executeMethod is a blocking, synchronous call  
    int statusCode = client.executeMethod(method);  
    System.out.println("Response " + statusCode);  
}
```

MyServlet.java

Threaded servers assign one thread per request and use **blocking** I/O



Evented servers are gaining popularity

```
var callback = function(data) {
  console.log("Response: " + data);
};

var options = {
  hostname: 'www.google.com',
  path: '/upload'
};

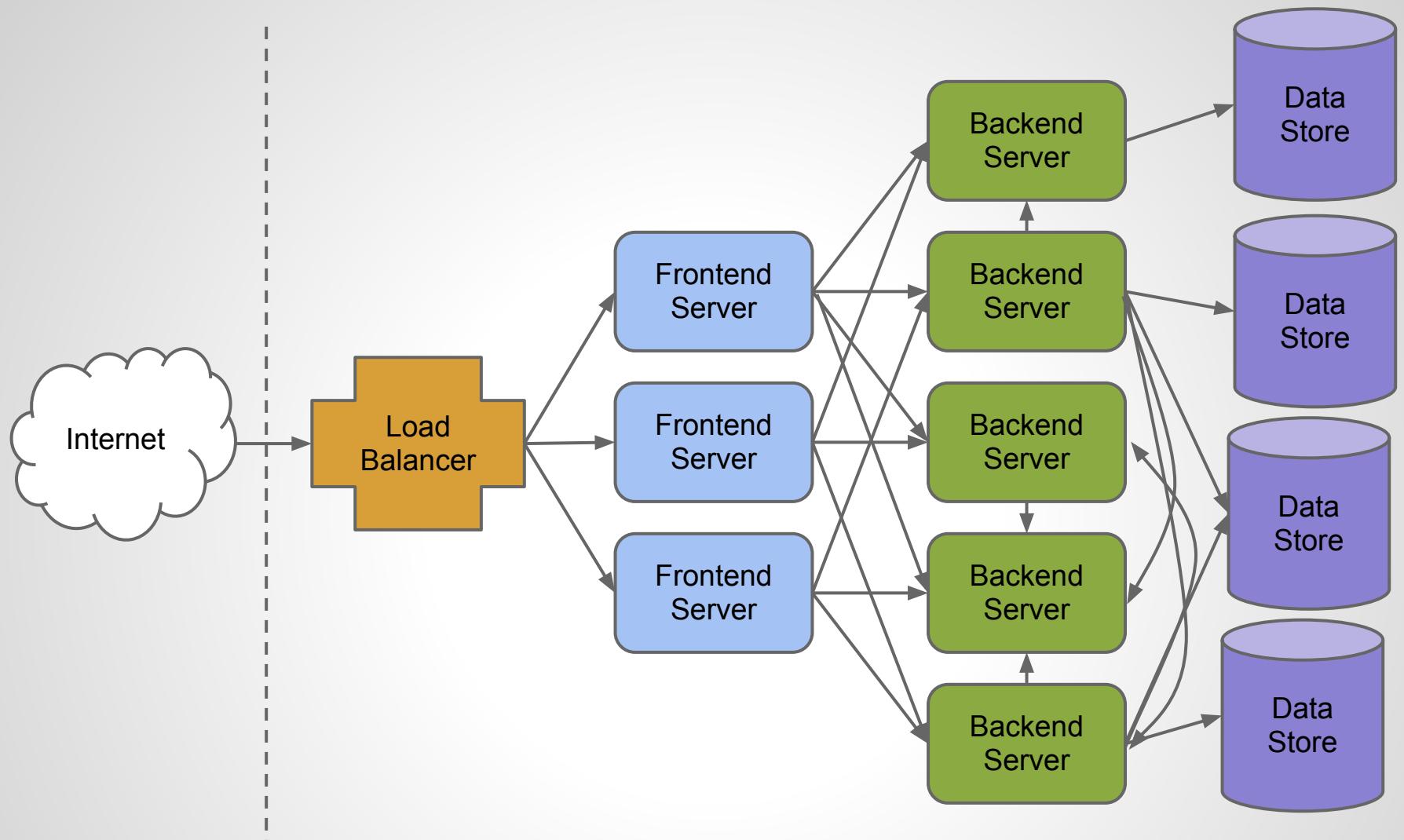
// Non-blocking HTTP call
http.request(options, callback);

console.log('This line may execute before the callback!');
```

MyNodeApp.js

Evented servers have one thread/process per CPU core and use **non-blocking** I/O

Why threaded vs. evented matters for
LinkedIn



LinkedIn uses a Service Oriented Architecture

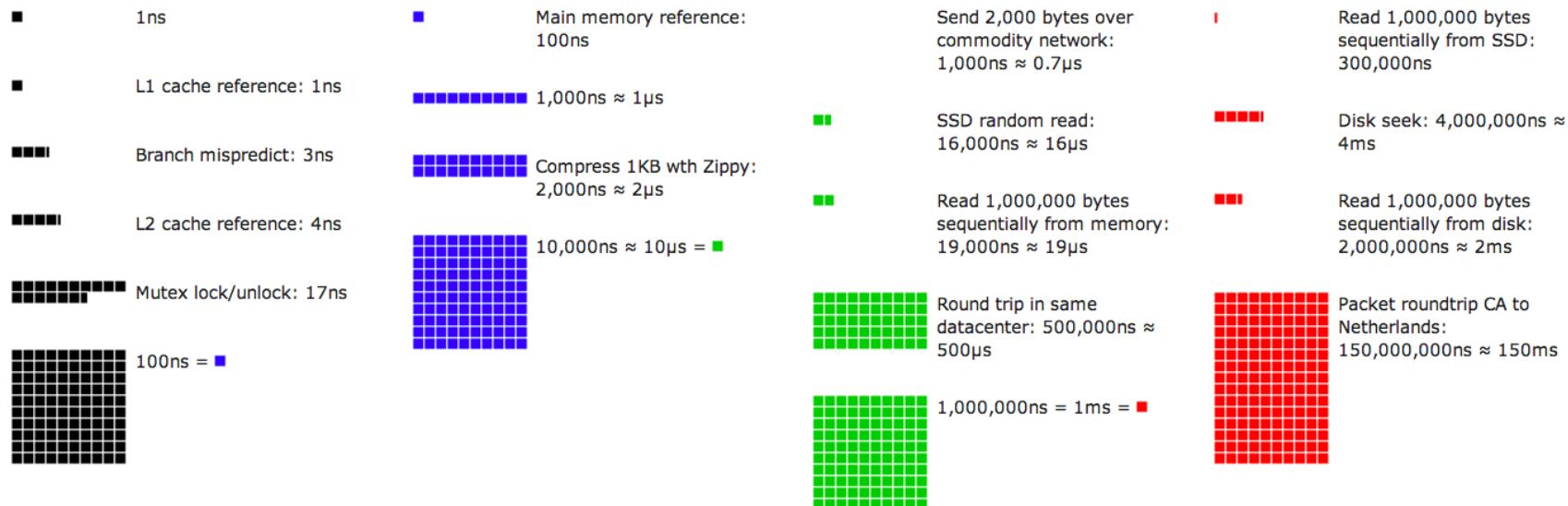
```
void doGet(HttpServletRequest req, HttpServletResponse res) {  
    // Call a number of backend services to get data  
    Profile profile = profileSvc.getProfile();  
    Company company = companySvc.getCompany();  
    Skills skills = skillsSvc.getSkills();  
}
```

MyServlet.java

Our services spend most of their time **waiting** for data from other services and data stores

Latency Numbers Every Programmer Should Know

2012

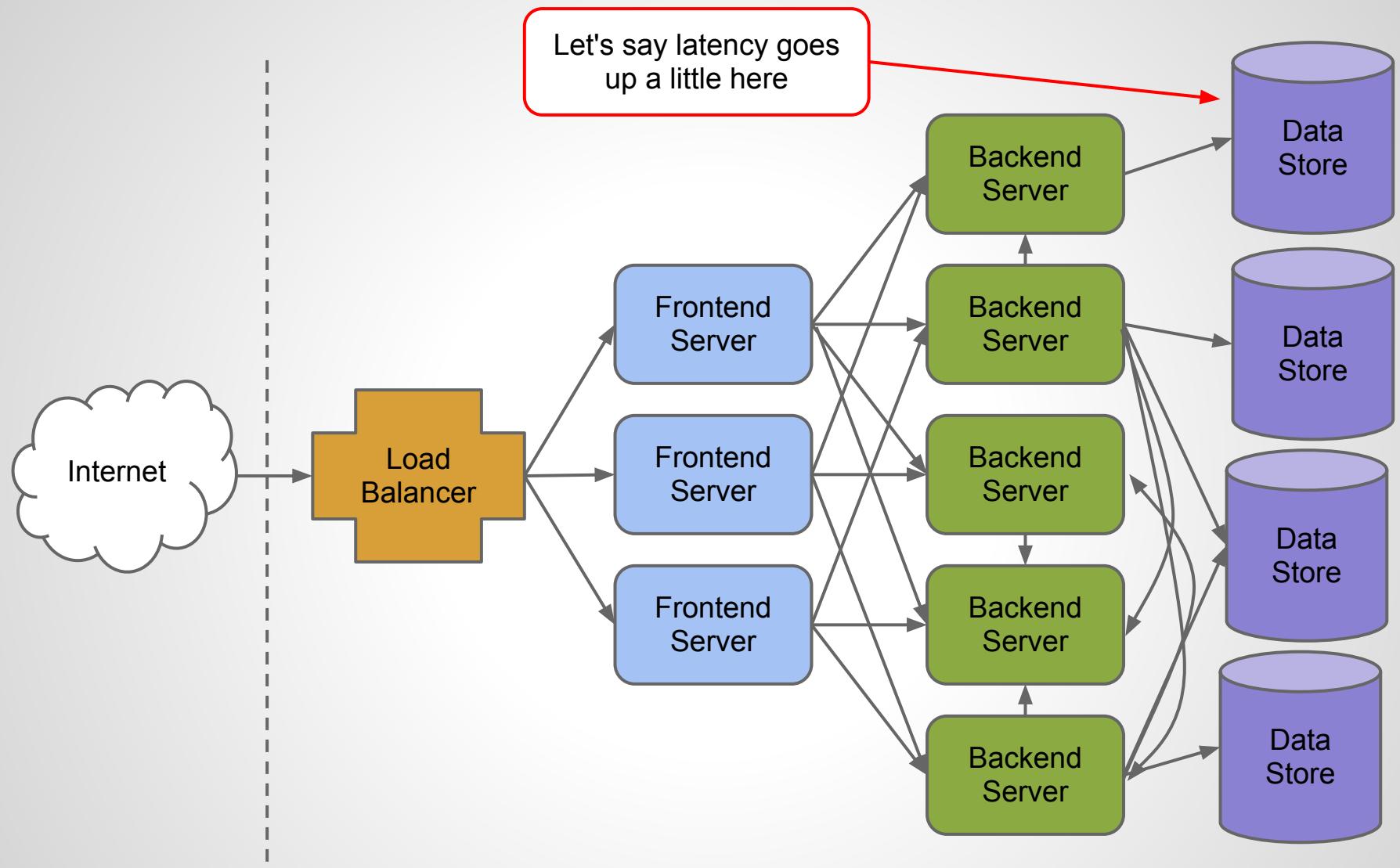


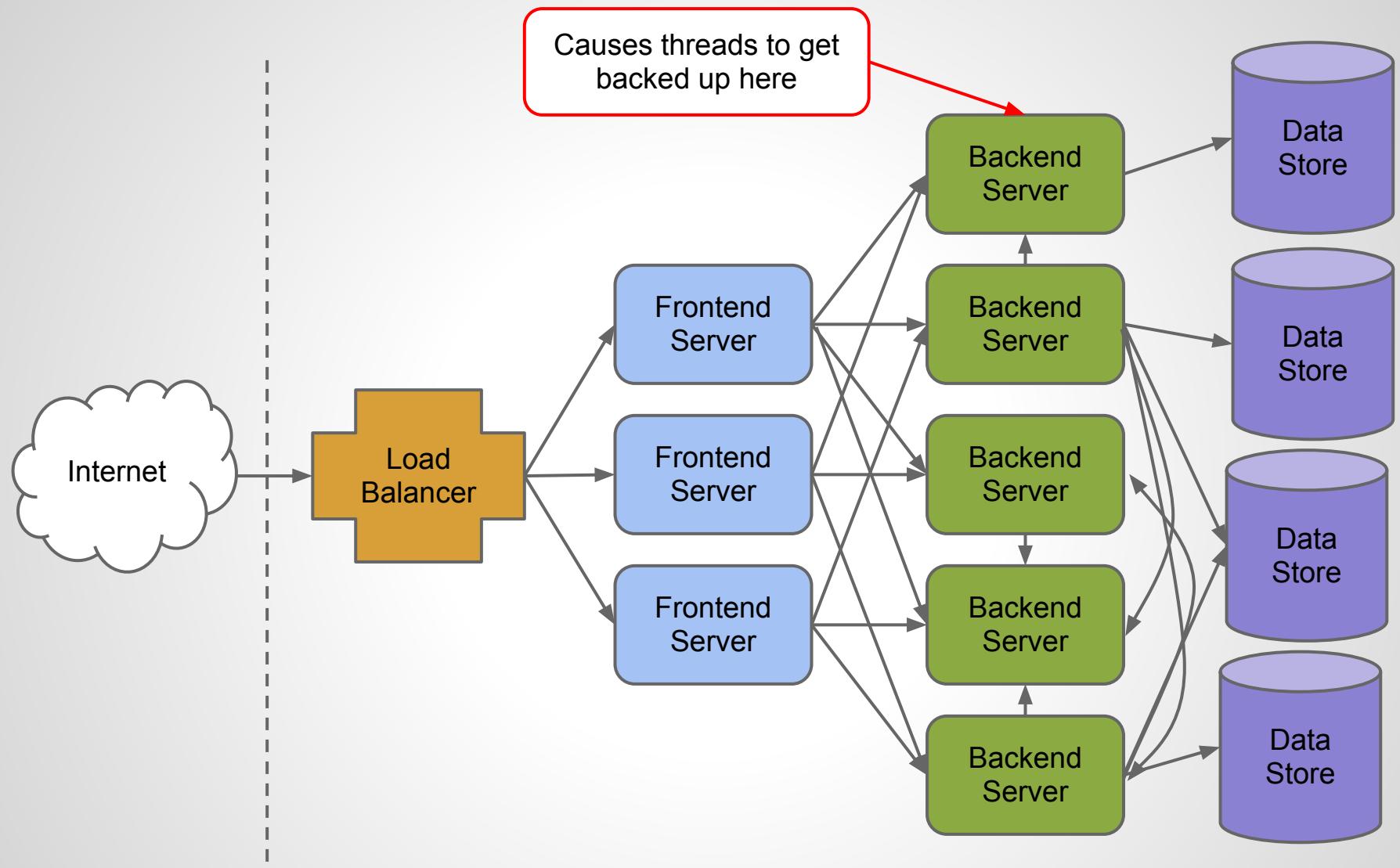
I/O is very expensive [4]

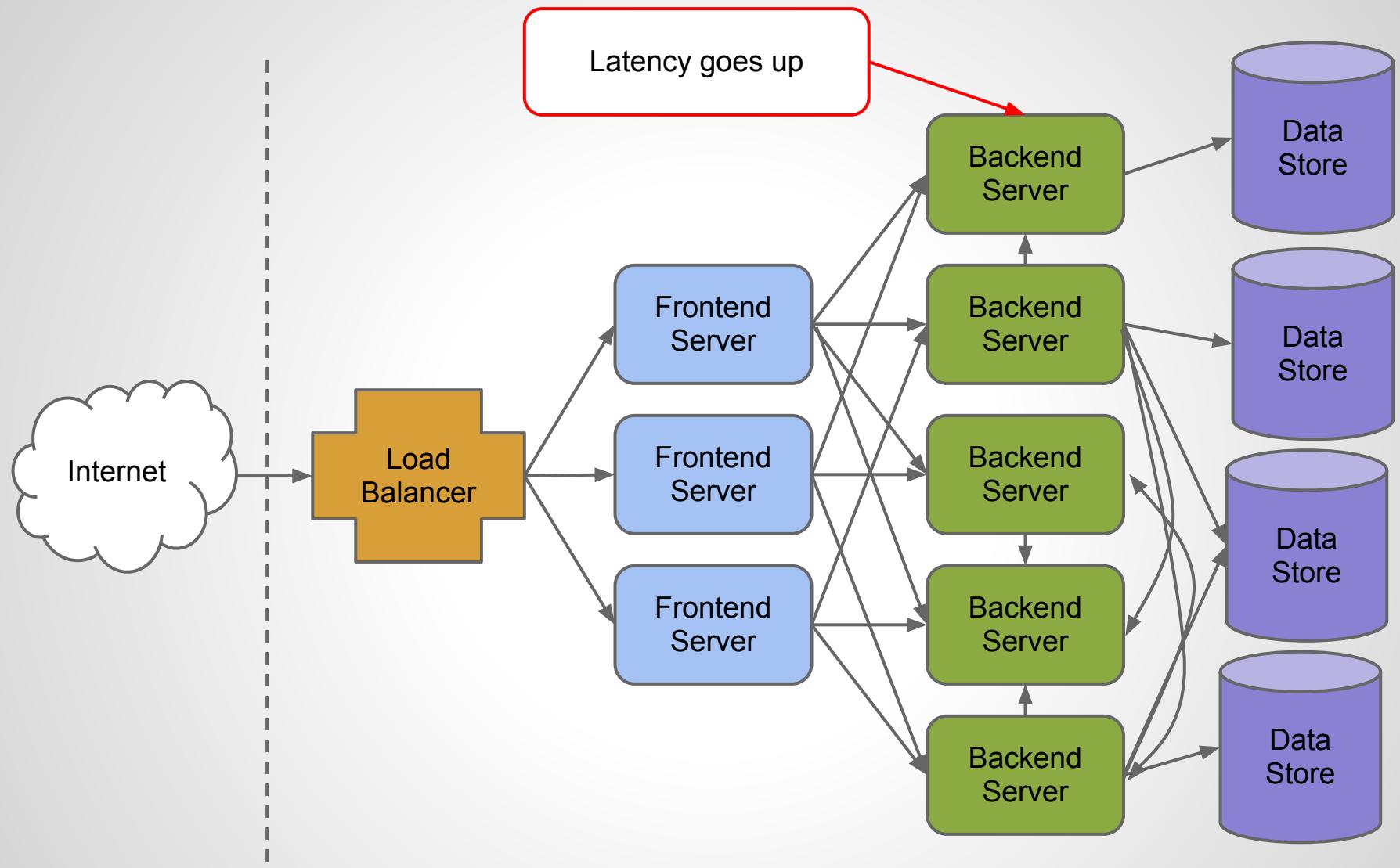
In a threaded server, threads spend most of
the time **idle**, waiting on I/O

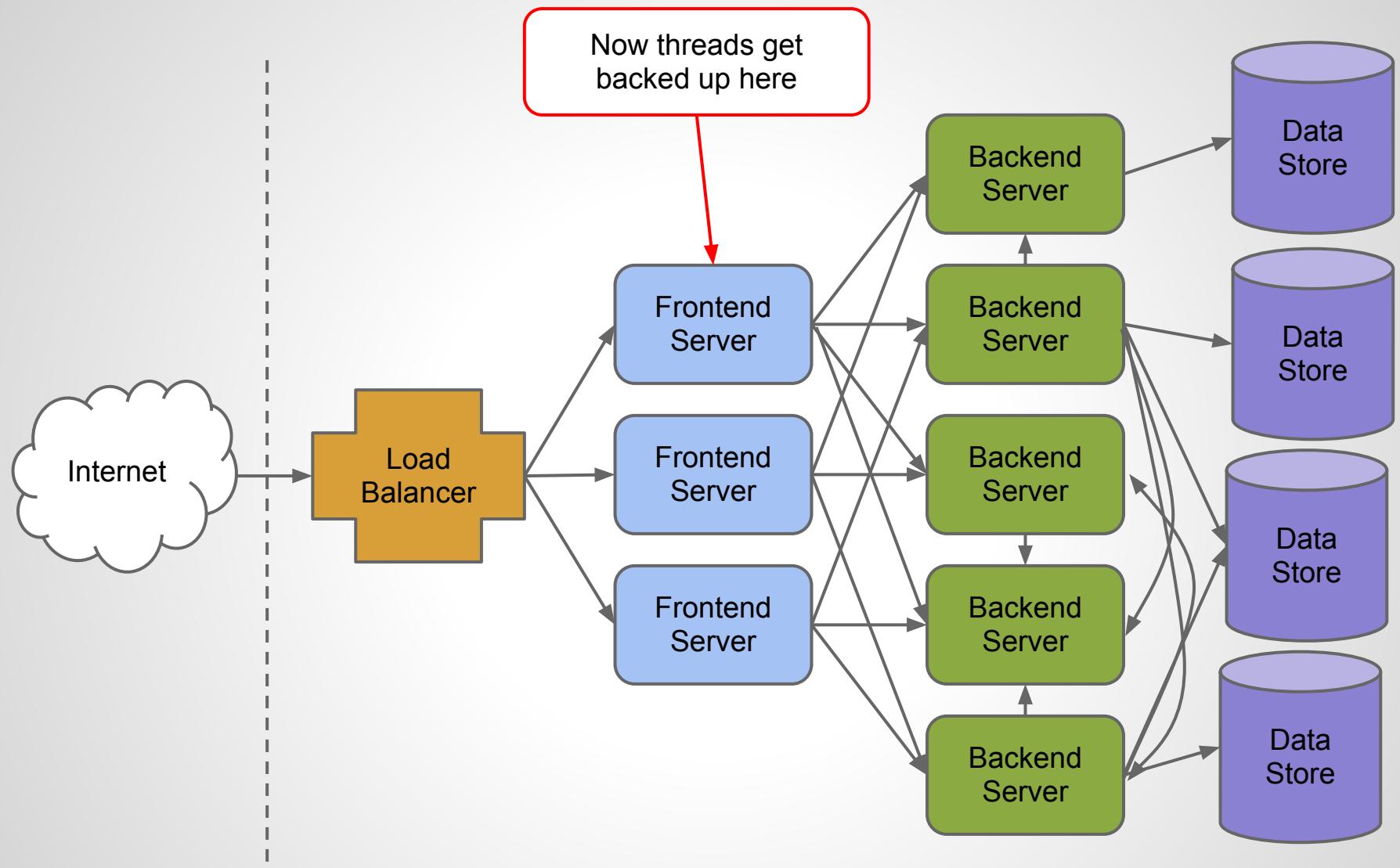
Threading dilemma

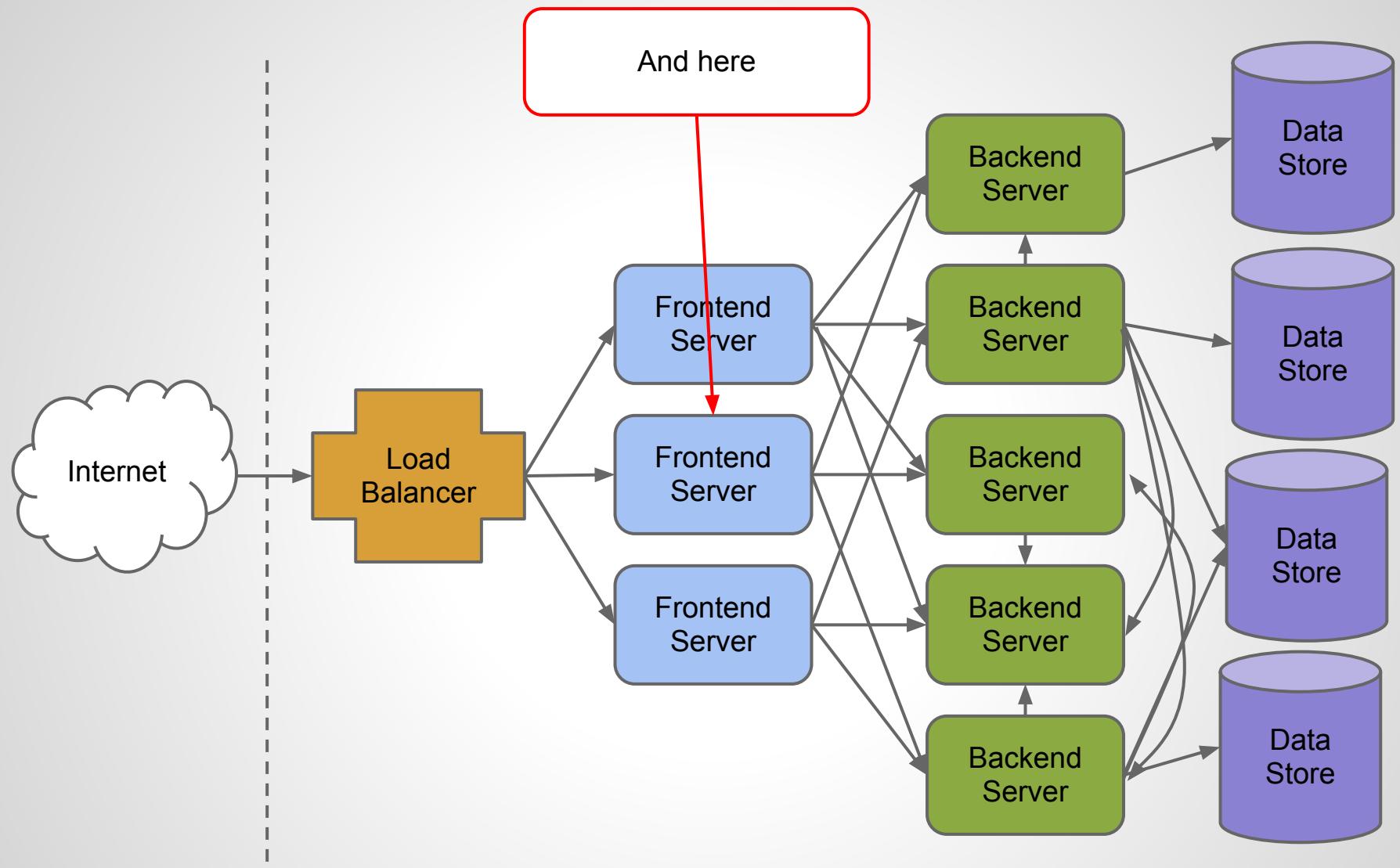
1. Creating new threads on the fly is expensive:
 - a. Use a thread pool
2. Too many threads in the thread pool:
 - a. Memory overhead
 - b. Context switching overhead
3. Too few threads in the thread pool:
 - a. Run out of threads, latency goes up
 - b. Sensitive to downstream latency!

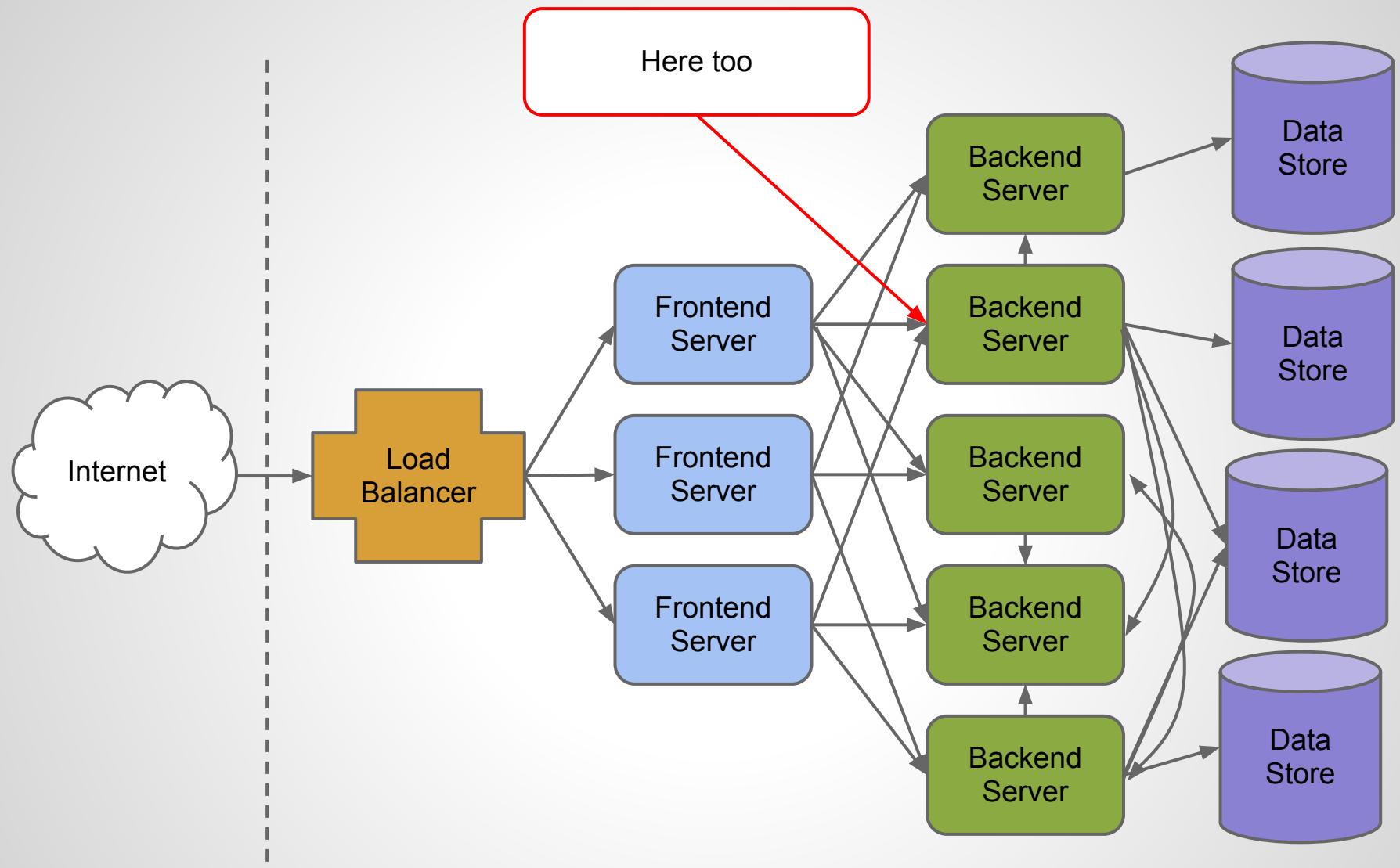


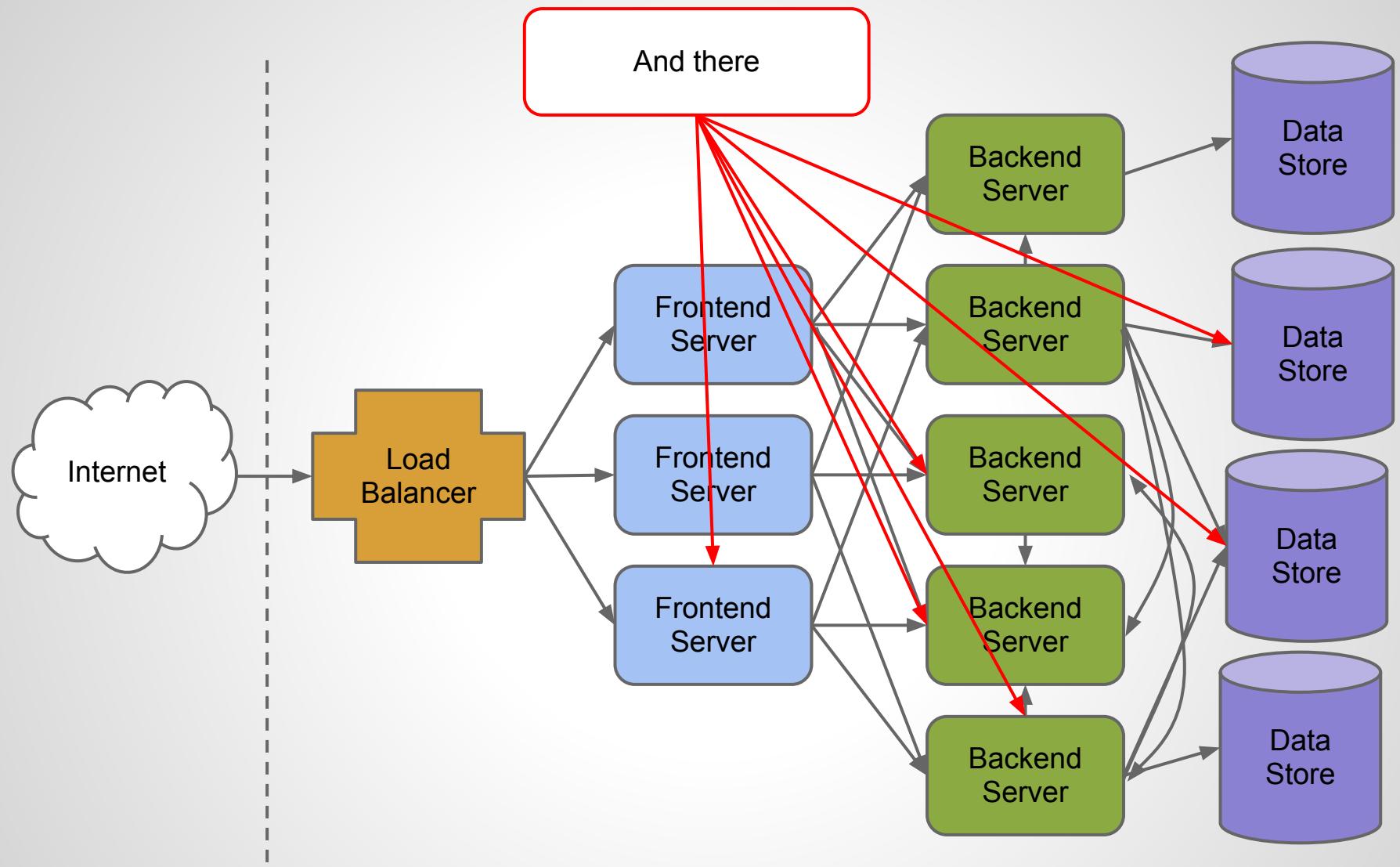












And there



And... the site is down.

This is thread pool hell [5]



Play is built on top of Netty, so it supports non-blocking I/O [6]

NIO benefits

1. No sensitivity to downstream slowness
2. Easy to parallelize I/O
3. Supports many concurrent and long-running connections, enabling:
 - a. WebSockets
 - b. Comet
 - c. Server-Sent Events

Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. **Non-blocking I/O**
6. Scala
7. Performance
8. Community

```
public class Proxy extends Controller {  
  
    public static Result index(String url) {  
        // Non blocking HTTP call  
        Promise<Response> responsePromise = WS.url(url).get();  
  
        // How do we turn a Promise into a Play Result?  
    }  
}
```

app/controllers/Proxy.java

Use Play's built in `WS` library to make a non-blocking HTTP call

A Promise<T> will *eventually* contain the value T (or an error)

```
public static class AsyncResult implements Result {  
    private final Promise<Result> promise;  
  
    public AsyncResult(Promise<Result> promise) {  
        this.promise = promise;  
    }  
}
```

(Play Framework source code)

Play has a built-in subclass of Result called
AsyncResult that takes a Promise<Result>

```

public class Proxy extends Controller {
    public static Result index(String url) {
        // Non blocking HTTP call
        Promise<Response> response = WS.url(url).get();

        // Transform asynchronously into a Play Result
        Promise<Result> result = response.map(toResult);

        return async(result);
    }

    // A function that can transform a Response into a Result
    private static Function<Response, Result> toResult =
        new Function<Response, Result>() {
            public Result apply(Response response) {
                return ok(response.getBody()).as("text/html");
            }
        };
}

```

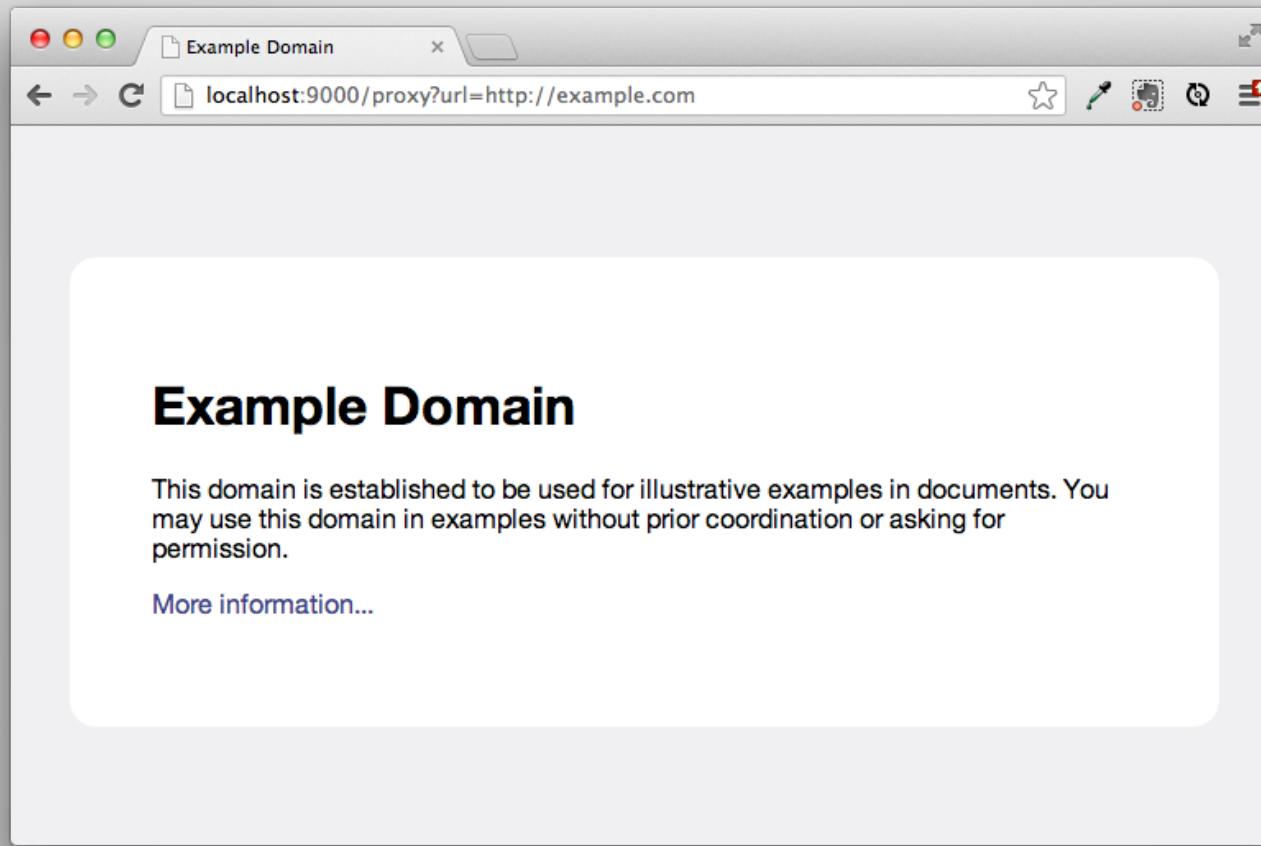
app/controllers/Proxy.java

We can use the map method to turn a
 Promise<Response> into a Promise<Result>

```
GET      /proxy      controllers.Proxy.index(url)
```

conf/routes

Let's add this endpoint to the routes file



<http://localhost:9000/proxy?url=http://example.com>

We just built a completely
non-blocking proxy!

```

public class Proxy extends Controller {
    public static Result index(String url) {
        Logger.info("Before the HTTP call");
        Promise<Response> response = WS.url(url).get();
        Promise<Result> result = response.map(toResult);
        Logger.info("After the HTTP call");

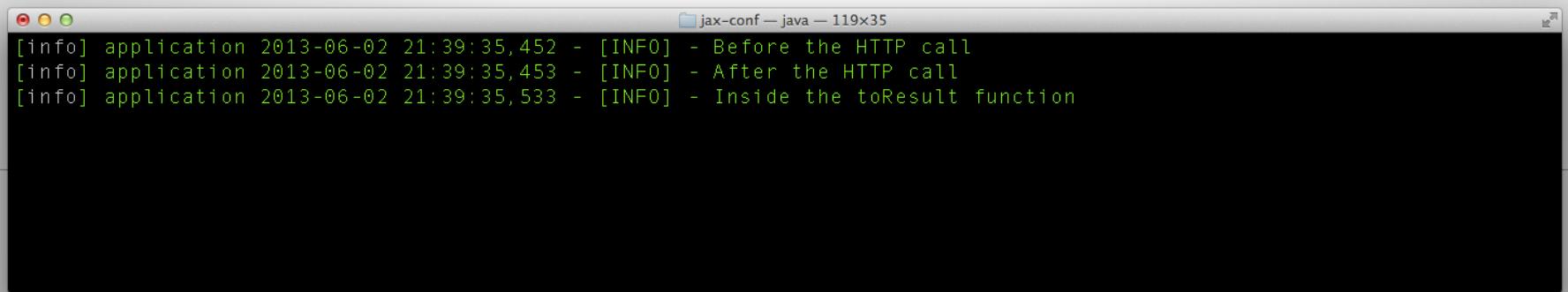
        return async(result);
    }

    private static Function<Response, Result> toResult =
        new Function<Response, Result>() {
            public Result apply(Response response) {
                Logger.info("Inside the toResult function");
                return ok(response.getBody()).as("text/html");
            }
        };
}

```

app/controllers/Proxy.java

To see that it's non-blocking, let's add logging



A screenshot of a terminal window titled "jax-conf — java — 119x35". The window contains the following log output:

```
[info] application 2013-06-02 21:39:35,452 - [INFO] - Before the HTTP call
[info] application 2013-06-02 21:39:35,453 - [INFO] - After the HTTP call
[info] application 2013-06-02 21:39:35,533 - [INFO] - Inside the toResult function
```

Refresh the page and the logs show the
HTTP call really is non-blocking

NIO makes parallel requests easy.
Let's try it.

```

class Timing {
    public String url;
    public long latency;

    public Timing(String url, long latency) {
        this.url = url;
        this.latency = latency;
    }

    // Fire an HTTP request and record how long it took
    public static Promise<Timing> timedRequest(final String url) {
        final long start = System.currentTimeMillis();
        Promise<Response> res = WS.url(url).get();

        return res.map(new Function<Response, Timing>() {
            public Timing apply(Response response) {
                long latency = System.currentTimeMillis() - start;
                return new Timing(url, latency);
            }
        });
    }
}

```

app/models/Timing.java

First, define a class that times an HTTP request

```

public class Parallel extends Controller {
  public static Result index() {
    // A Promise that will redeem when the 3 parallel
    // HTTP requests are done
    Promise<List<Timing>> all = Promise.waitFor(
      Timing.timedRequest("http://www.yahoo.com"),
      Timing.timedRequest("http://www.google.com"),
      Timing.timedRequest("http://www.bing.com")
    );
    // Asynchronously transform timings into a JSON response
    return async(all.map(new Function<List<Timing>, Result>() {
      public Result apply(List<Timing> timings) {
        return ok(Json.toJson(timings));
      }
    }));
  }
}

```

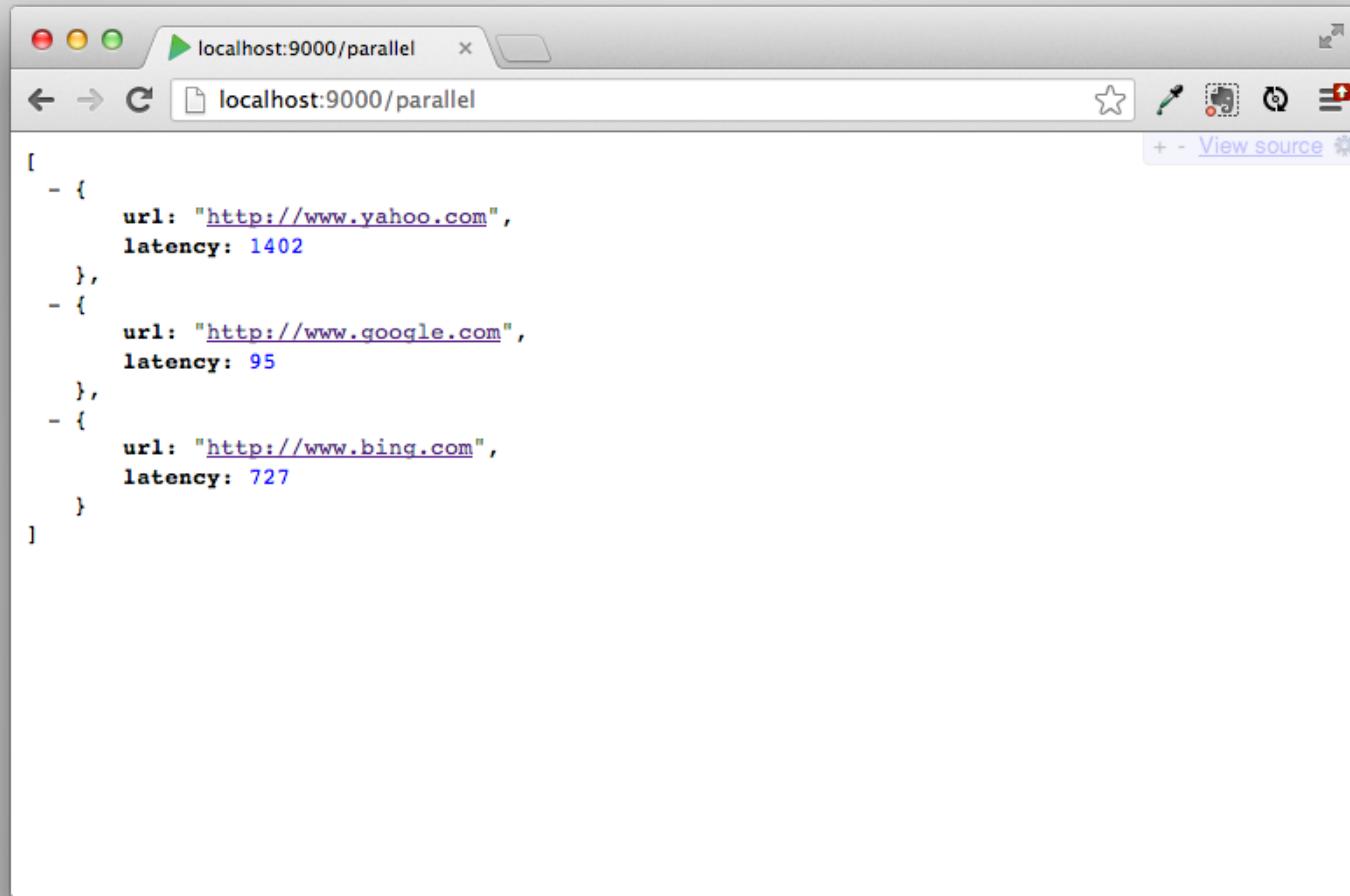
app/controllers/Parallel.java

Next, add a controller that fetches 3 URLs in parallel and returns the timings as JSON

```
GET      /parallel      controllers.Parallel.index()
```

conf/routes

Add it to the routes file



A screenshot of a web browser window titled "localhost:9000/parallel". The address bar also shows "localhost:9000/parallel". The content area displays a JSON array with three elements, each representing a URL and its latency:

```
[  
  - {  
      url: "http://www.yahoo.com",  
      latency: 1402  
    },  
  - {  
      url: "http://www.google.com",  
      latency: 95  
    },  
  - {  
      url: "http://www.bing.com",  
      latency: 727  
    }  
]
```

<http://localhost:9000/parallel>

Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. **Scala**
7. Performance
8. Community



Play has native support for Scala

```
object HelloWorldScala extends Controller {  
  
    def index = Action {  
        Ok("Hello World Scala")  
    }  
}
```

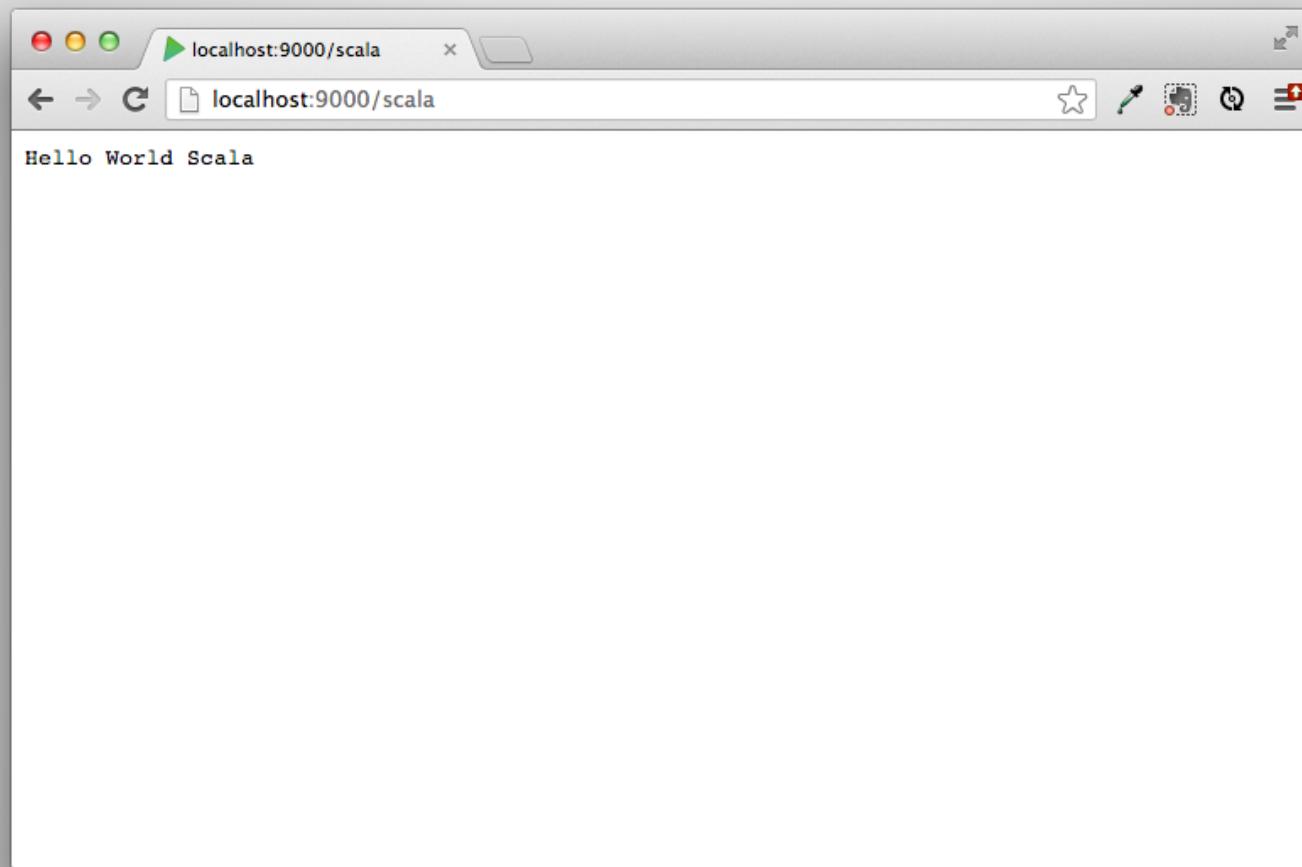
app/controllers/HelloWorldScala.scala

Just add a .scala file under /app and
Play will compile it

```
GET      /scala      controllers.HelloWorldScala.index()
```

conf/routes

Add it to the routes file as usual



<http://localhost:9000/scala>

Play/Scala: the good parts

```
// 2 parallel async calls
val fooFuture = WS.url(url1).get()
val barFuture = WS.url(url2).get()

for {
    foo <- fooFuture
    bar <- barFuture
} yield {
    Ok(....)
}
```

Sequential async calls

```
// 2 sequential async calls
for {
    foo <- WS.url(url1).get()
    bar <- WS.url(buildUrl(foo)).get()
} yield {
    Ok(....)
}
```

Sequential async calls

API is more concise, expressive, & composable, especially for async code.

```
[jax-conf-frontend] $ console
[info] Starting scala interpreter...
[info]
Welcome to Scala version 2.9.2 (Java HotSpot(TM) 64-Bit Server VM, Java 1.6.0_45).
Type in expressions to have them evaluated.
Type :help for more information.

scala> import controllers.HelloWorld
HelloWorld      HelloWorldScala

scala> import controllers.HelloWorld
import controllers.HelloWorld

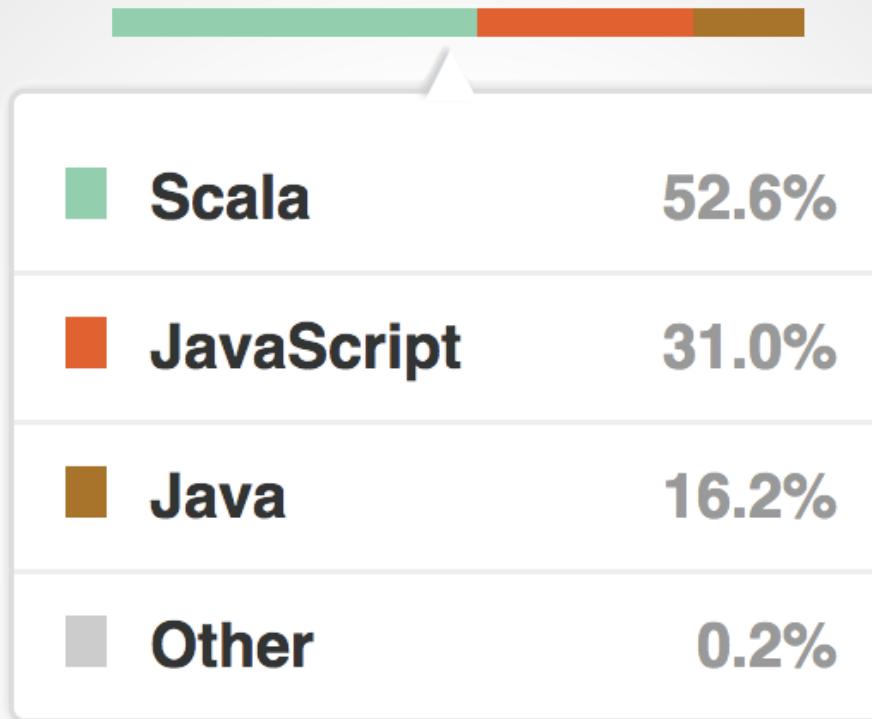
scala> val result = HelloWorld.index("Jim", 29)
result: play.mvc.Result = SimpleResult(200, Map(Content-Type -> text/html; charset=utf-8))

scala> new String(play.core.j.JavaResultExtractor.getBody(result))
res0: java.lang.String =
"

<html>
  <head></head>
  <body>
    
    <p>
      Hello <b>Jim</b>, you are <b>29</b> years old
    </p>
  </body>
</html>
"

scala>
```

Interactive console with full classpath of your app



Play is Scala at its core: main libraries,
routes, templates, SBT, etc [7]

Play/Scala: the less good parts

The logo for sbt, consisting of the lowercase letters "sbt" in a white sans-serif font, centered on a solid dark teal rectangular background.

sbt

Simple Build Tool: Play's build system. Very powerful, but *very* steep learning curve.

```
val settings = Seq(  
    bootPath <=> target(_ / "boot") ,  
    buildDir <=> baseDirectory(_ / ".." / "build") ,  
    srcDir <=> baseDirectory(_ / ".." / "src") ,  
    enableConfigCompile := true,  
    fabrics <+=> Seq("EI", "EI2")  
)
```

project/Build.scala

Abandon all hope, ye who enter here
without knowing Scala.

```
fabrics <++= Seq("EI", "EI2")
```

project/Build.scala

WTF is <++=? Some kind of fish bones?
How do I google that?

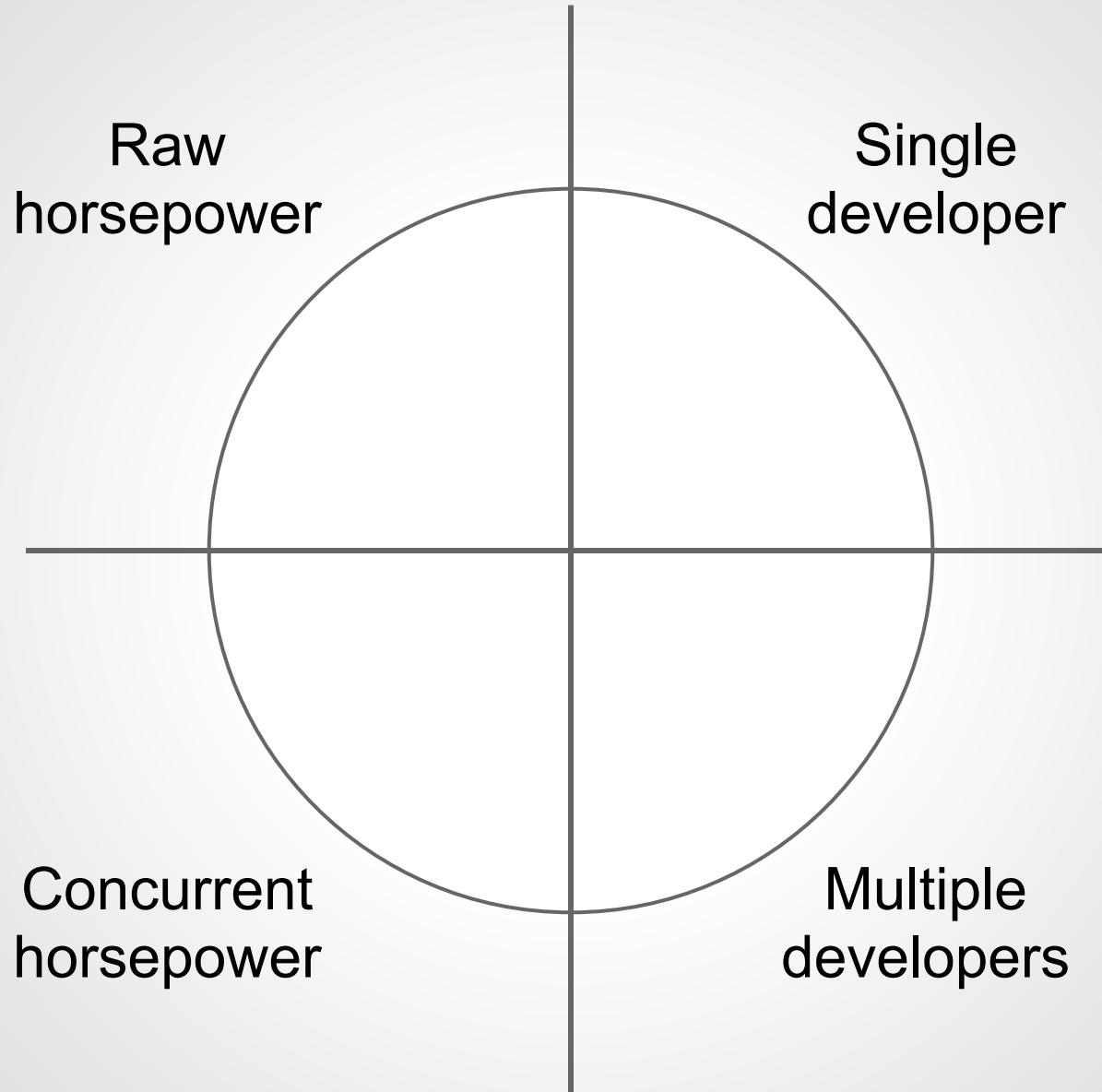
Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. **Performance**
8. Community



Is Play web scale? [8]

Scalability can be measured along
many dimensions



Raw horsepower: theoretical maximum performance for the server in ideal conditions. A measure of language and framework overhead.

Concurrent horsepower: performance with many users, I/O, and more real-world scenarios. A measure of the framework's approach to concurrency.

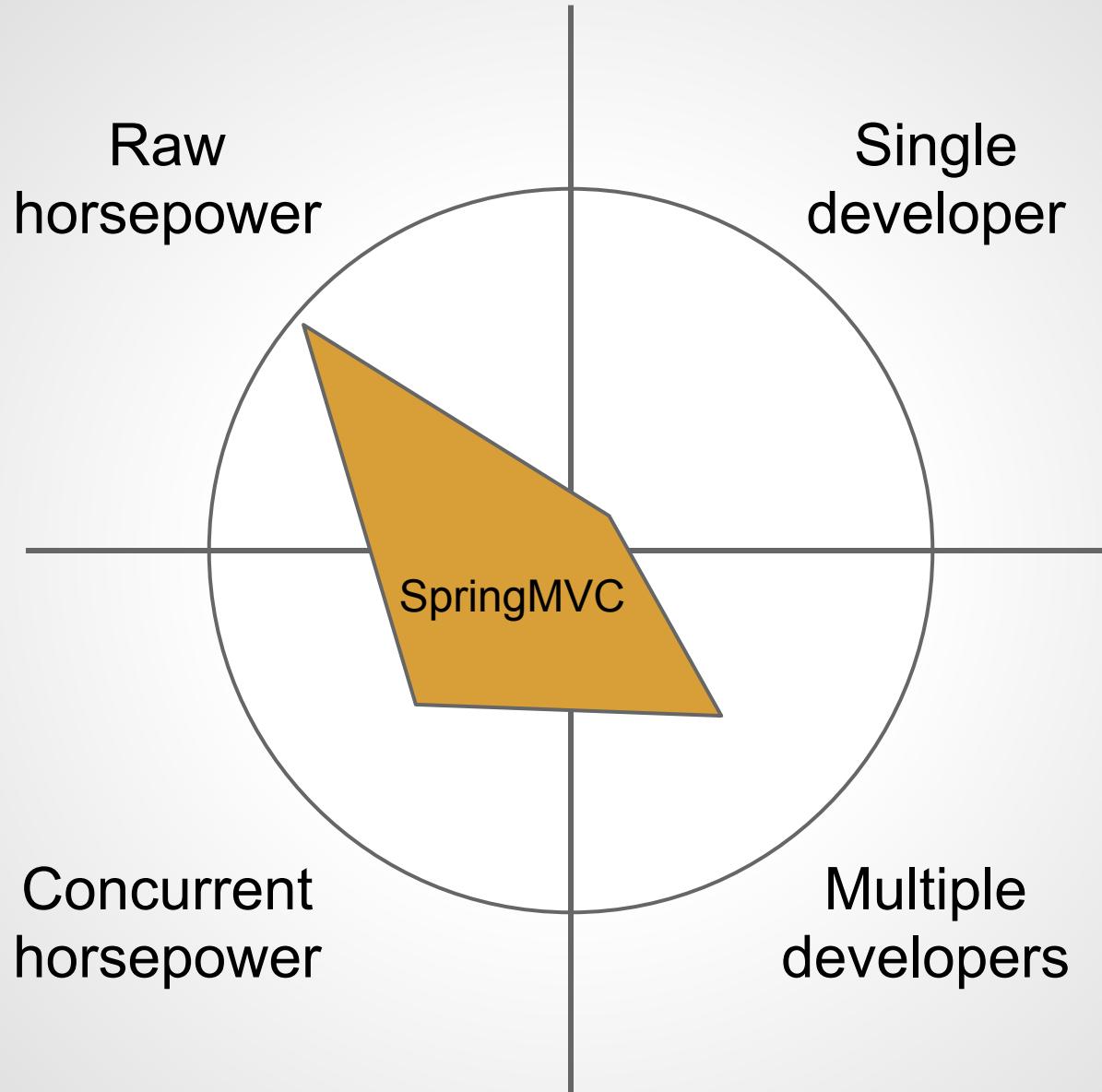
Single developer: how easy it is to get started, how quickly a single developer can build things. A measure of the framework's raw productivity and tooling.

Multiple developers: how the framework tolerates many developers working concurrently over many years. A measure of code rot and maintainability.

Here is how I'd rate some of the frameworks I've used. YMMV.



- **Pros**
 - Good raw throughput (qps, latency)
 - Type safety reduces code rot
- **Cons**
 - Not developer friendly. Getting things done takes forever.
 - Threaded, synchronous approach difficult to scale for lots of I/O in a SOA environment.



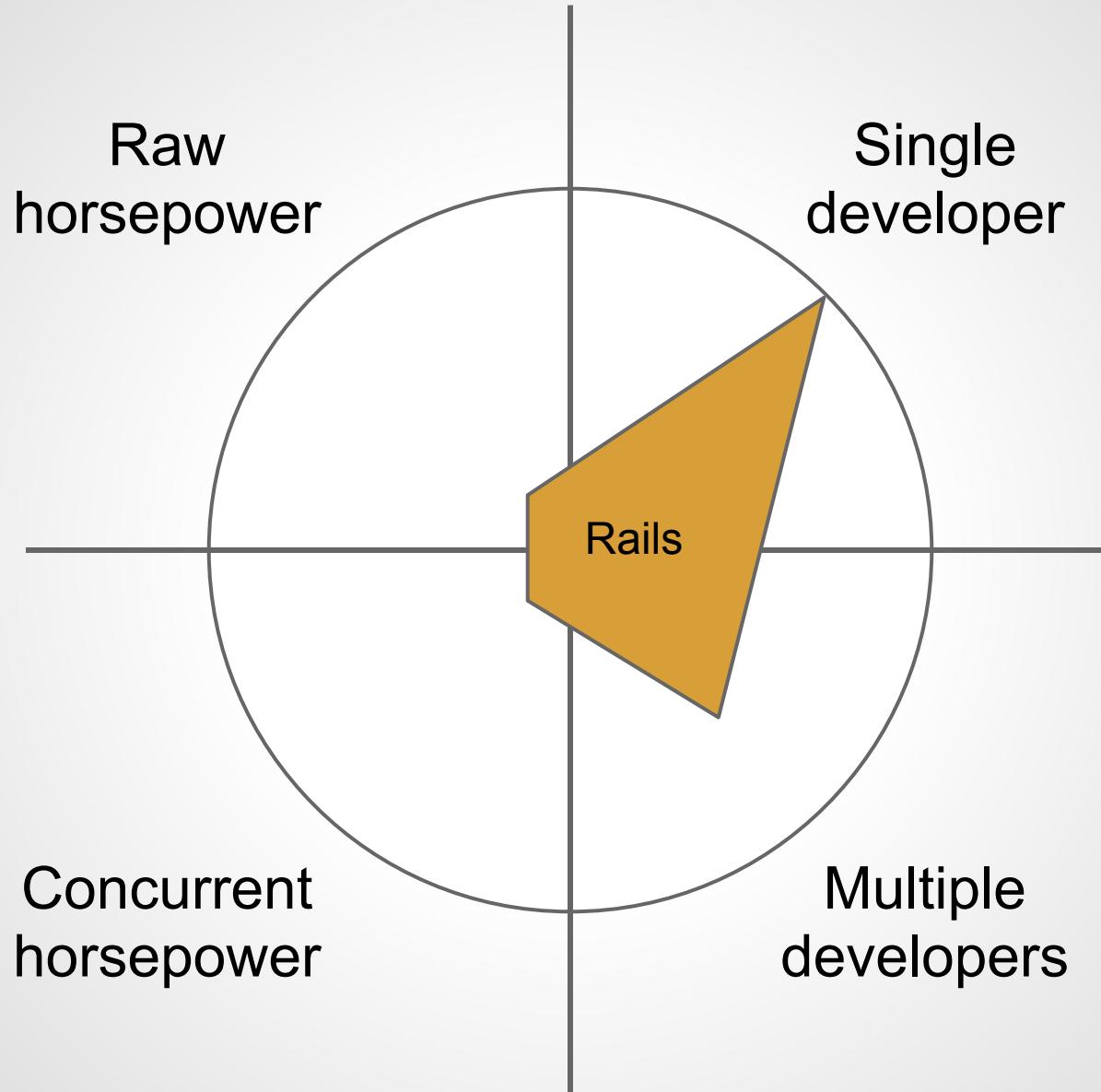


- **Pros**

- Set the bar for developer productivity; all other frameworks are still trying to catch up.

- **Cons**

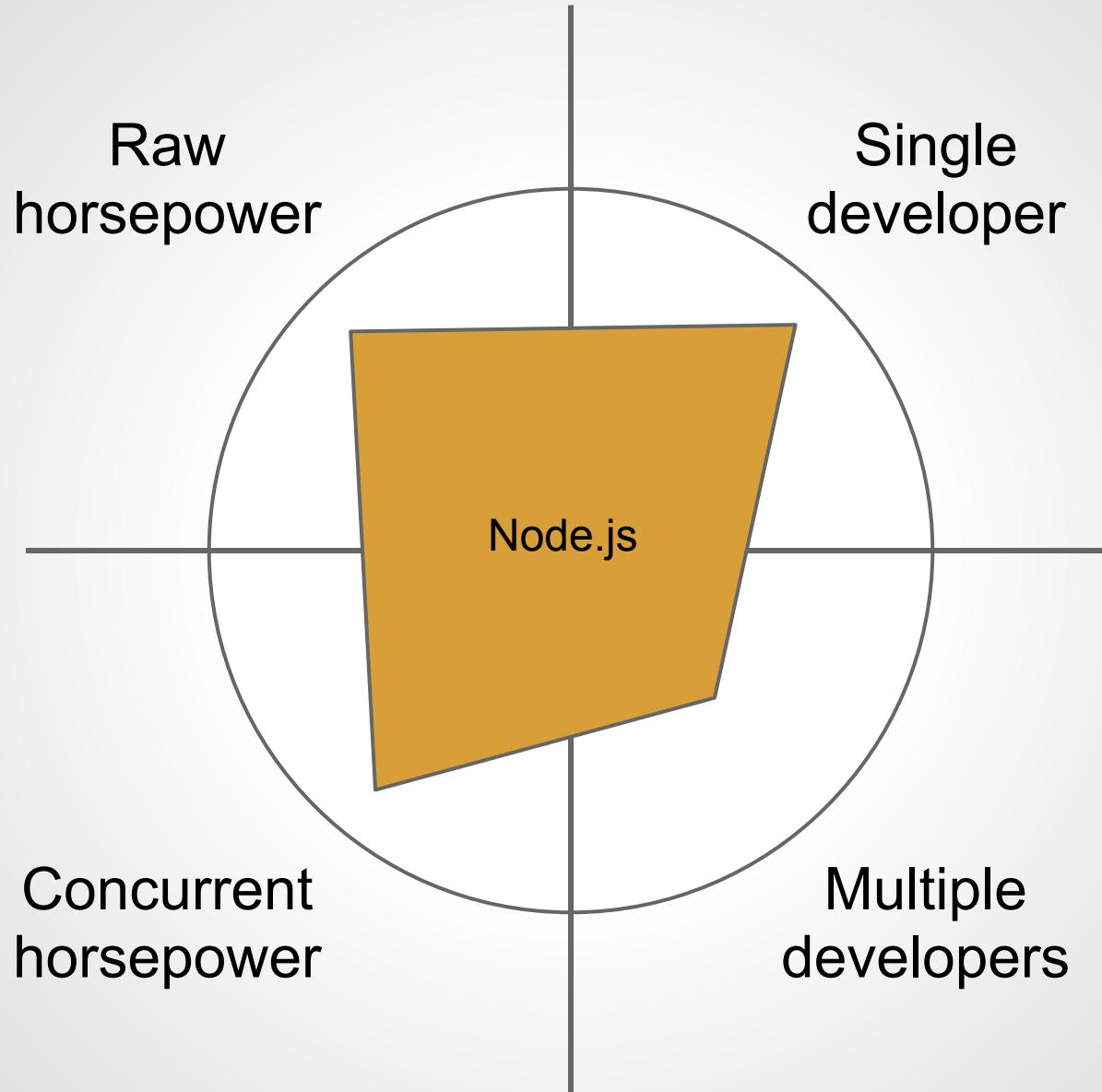
- Ruby is slow
- Ruby doesn't have real multithreading nor a great evented framework
- Dynamic language makes it tougher to maintain a large codebase





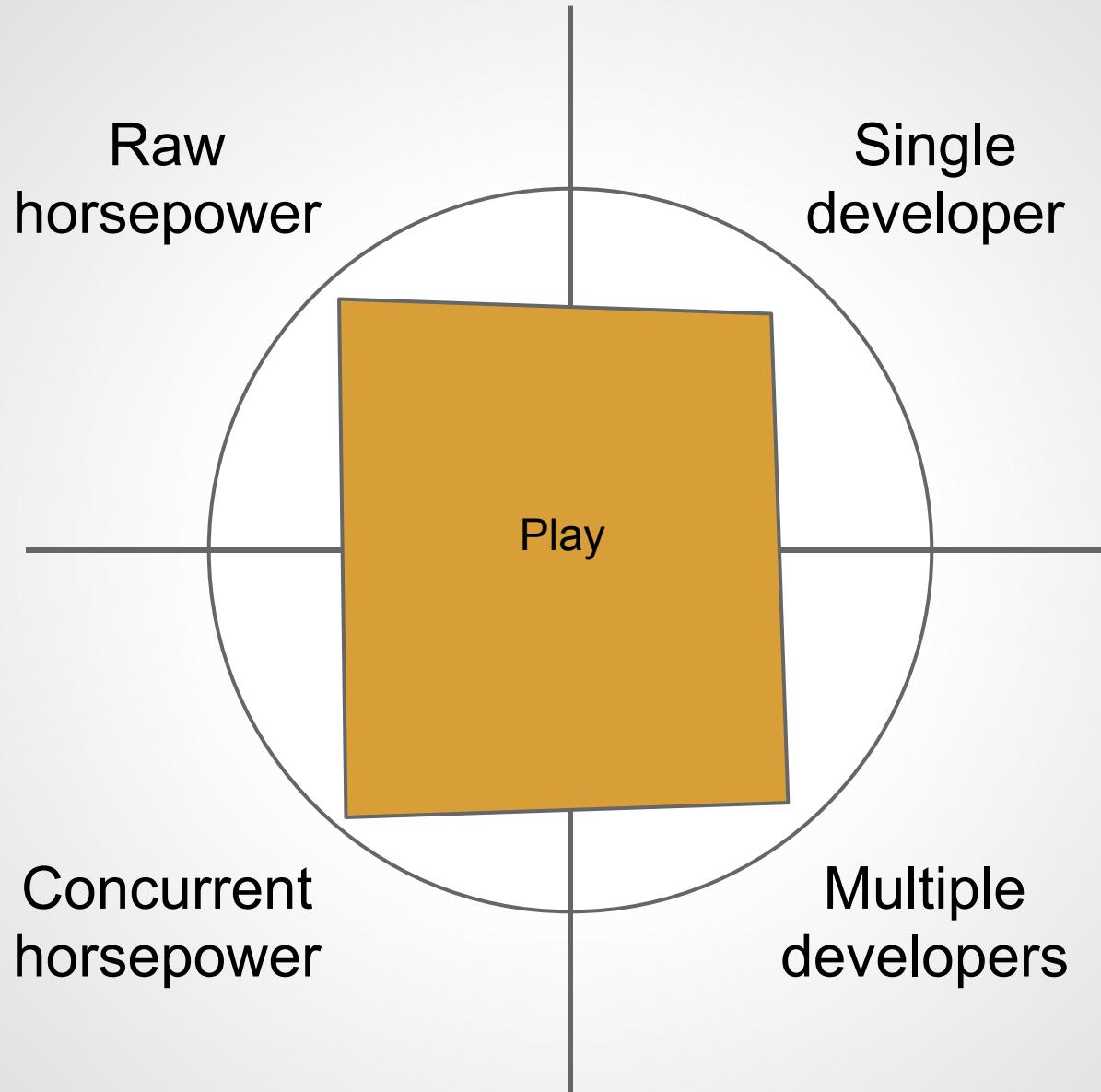
- **Pros**
 - v8 engine is pretty fast for raw throughput
 - Non-blocking I/O at the core makes concurrency easy
 - Strong open source community and lightning fast startup time makes it easy to get things done quickly

- **Cons**
 - Dynamic language makes it tougher to maintain a large codebase
 - Lots of immature libraries that constantly make backwards incompatible changes





- **Pros**
 - Fast for raw throughput
 - Non-blocking I/O at the core makes concurrency easy
 - Hot reload makes it possible to get things done quickly
 - Strong type safety throughout reduces code rot
- **Cons**
 - Even with hot reload, a compiled statically typed language isn't *quite* as fast as an interpreted dynamically typed language



Outline

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. **Community**

The screenshot shows a GitHub repository page for 'playframework / Play20'. The repository is public and has 3,203 stars, 982 forks, and 188 issues. It includes tabs for Code, Network, Pull Requests (38), Issues (188), and Graphs. Below the tabs, there's a note about the Play framework 2.x documentation and a link to <http://www.playframework.org/>. There are download options for Clone in Mac, ZIP, HTTP, SSH, and Git Read-Only, along with a copy link and a 'Read-Only access' button. The 'Code' tab is selected, showing a list of recent commits. The commits are as follows:

File	Date	Message
documentation	a day ago	fix test code to cope with new architecture [XuefengWu]
framework	6 days ago	Fixed race condition in JPA async commit [jroper]
samples	a month ago	Allow user code supplied to iteratees and enumerators to run in its o... [richdougherty]
.gitignore	a year ago	Findbugs is not needed as a dependency at runtime. Upgrading Javassis... [guillaumebort]
CONTRIBUTING.md	3 months ago	Updated contributor guidelines [jroper]
README.md	3 months ago	Updated issue tracker information [jroper]
play	2 months ago	Merge pull request #777 from mikemckibben/fix/play_launch_script_rels... [huntc]
play.bat	a year ago	[#324 #323] add a check against 'project' folder in play scripts [pk11]

Play is open source [7]

Expose a withTimeout method

GitHub, Inc. [US] https://github.com/playframework/Play20/pull/689

Discussion Commits 1 Files Changed 1

brikis98 opened this pull request 4 months ago

Expose a withTimeout method in anorm

No one is assigned No milestone

Merged

+ 10 additions
- 1 deletion

This PR adds a `withTimeout` method to anorm so it's possible to set a query timeout on DB calls. This uses `Statement.setQueryTimeout` under the hood.

A question for reviewers:

1. Is there any way to unit test this? I couldn't find any unit tests for anorm and I have no clue how to simulate a slow DB query.
2. Is there any way for anorm to get a config parameter so it would be possible to set a timeout for *all* DB queries? It looks like anorm is intentionally kept separate from Play code, so I can't exactly do `Play.current.configuration.getInt(...)`.

3 participants

Jim Brikman added a commit 4 months ago

Jim Brikman #950. Expose a withTimeout method to be able to set the queryTimeout... [...](#) [d74d576](#)

jroper commented 4 months ago

1. My first thought when I looked at it was that there are no tests, but then I thought maybe it's not practical. I seem to remember that hsqldb ran all transactions in serial, so maybe you could force it to block by having two transactions, and setting the timeout on a query in the second, you might be able to force a timeout. But maybe that's too fragile.

LinkedIn is contributing

The screenshot shows a Google search results page for the query "play-framework". The top navigation bar includes links for Get involved, Issues - playframework/Play, and (99+) play-framework - G. Below the bar, there are links for +Yevgeniy, Search, Images, Maps, Play, YouTube, News, Gmail, Drive, Calendar, and More. The main content area is titled "Groups" and features a "POST A QUESTION" button. On the left, a sidebar lists "My groups" (play-framework), "Home", "Starred", and "Favorites" (with a note to click a star icon). Under "Recently viewed", it lists play-framework, scala-user, simple-build-tool, scala-internals, nodejs, and Recent searches (https, compass, Communications). At the bottom of the sidebar are links for ©2013 Google, Privacy - Terms of Service -, and Google Home.

play-framework

32 of 14888 topics (99+ unread)

Members · About

The first post is moderated for new members - sorry for any delay this causes.

Please start subject lines with a Play version number or a module name-version, e.g. [2.0], [2.0-java], [2.0-scala], [1.2.4] or [gae-1.4].

Play 2.1.1 released (24)
By James Roper - 24 posts - 3494 views Apr 8

New rule for subject line: (12)
By Guillaume Bort - 24 posts - 6241 views 12/30/12

[2.1.1] Template escaping of curly braces. (1)
By James Lowry - 1 post - 0 views 1:36 AM

How do I exclude play-java related libs in a play-scala only project? (4)
By Shark.Z - 4 posts - 19 views 1:23 AM

[2.1-scala] classloader bug, code to reproduce included (1)
By fred - 1 post - 1 view Jun 2

Trouble configuring Play 2 to run with JPA (persistence) (3)
By Johann Gomes - 3 posts - 19 views Jun 2

[Play 2.1.0] Comprehensive Specs2, Mockito examples? (5)
By Leon Derkx - 5 posts - 69 views Jun 2

Setup postgres with JPA model (6)
By Bill O'Brien - 6 posts - 61 views

Very active google group (mailing list) [9]

Commit Activity · playfram × Highest Voted 'playframework' ×

← → C stackoverflow.com/tags/playframework

StackExchange v Yevgeniy Brikman 666 • 6 • 15 review chat meta about faq [playframework]

Ask Question

Tagged Questions

info newest frequent votes active unanswered

The Play! Framework is a modern Java and Scala web application open-source framework that provides a clean alternative to bloated Enterprise Java stacks.

learn more... | improve tag wiki | top users | synonyms (1)

177
votes
14
answers
31k views

Should I use Play or Lift for doing web development in Scala?

I'm stuck on whether I should focus on Play or Lift for doing web development in Scala. Play looks very polished. The Scala-specific tutorial looks amazing. Furthermore, since I've been coding in ...

scala lift playframework

asked Sep 8 '10 at 17:14 by Shannon jj Behrens 2,138 • 2 • 5 • 13

100
votes
6
answers
26k views

Play Framework: Real-world production experiences? [closed]

Has anyone used the Play framework for a reasonably complex or large, deployed production app yet? If so, I would like to hear what the pros and cons of that experience were and what you might do ...

java playframework

asked Apr 15 '10 at 15:31 by Rob 1,575 • 5 • 20 • 31

68
votes
7
answers
24k views

What is pro and contra of using Play Framework?

Can you provide a brief comparison of Play framework vs: Spring Roo, Grails, Django. In terms of learning curve performance maturity speed development/code reuse convention over configuration

playframework web-frameworks

asked Mar 21 '11 at 11:04 by yura 4,791 • 2 • 22 • 52

Community Bulletin

event Microsoft can help you port your app to Windows 8 - win prizes with Apptivate! – now through June 7

blog Welcome Tim Post, our latest Community Manager

Windows Azure

Move apps to the cloud without rewriting them.

Simple. Flexible. Connected.

FREE TRIAL

AdChoices

StackOverflow tag [10]

The screenshot shows a web browser window with the title bar "Modules" and the URL "www.playframework.com/documentation/2.1.1/Modules". The main content area displays the "Play 2.0 Modules" documentation. At the top right, there are buttons for "Browse versions" (set to 2.1.1), "Browse APIs" (with tabs for "Scala" and "Java"), and a search bar. Below these are sections for "Getting started", "Working with Play", "Detailed topics", "Additional documentation", and "Books".

Play 2.0 Modules

Airbrake.io notifier

- ▶ Website: <http://teamon.github.com/play-airbrake/> ↗
- ▶ Documentation: <https://github.com/teamon/play-airbrake/blob/master/README.md> ↗
- ▶ Short description: Send exception notifications to airbrake.io

Amazon STS module (Scala)

- ▶ Website: <https://github.com/Rhinofly/play-libraries/tree/master/apis/sts> ↗
- ▶ Documentation: <https://github.com/Rhinofly/play-libraries/tree/master/apis/sts/README.md> ↗
- ▶ Short description: STS (Security Token Service) API wrapper for Play 2.0

Amazon SES module (Scala)

- ▶ Website: <https://github.com/Rhinofly/play-libraries/tree/master/apis/ses> ↗
- ▶ Documentation: <https://github.com/Rhinofly/play-libraries/tree/master/apis/ses/README.md> ↗
- ▶ Short description: SES (Simple Email Service) API wrapper for Play 2.0

Amazon S3 module (Scala)

- ▶ Website: <https://github.com/Rhinofly/play-libraries/tree/master/apis/s3> ↗

Getting started

- ▶ Installing Play
- ▶ Creating a new application
- ▶ Anatomy of a Play application
- ▶ Using the Play console
- ▶ Setting-up your preferred IDE
- ▶ Sample applications
- ▶ Security policy

Working with Play

- ▶ Play for Scala developers
- ▶ Play for Java developers

Detailed topics

- ▶ The Build system
- ▶ Working with public assets
- ▶ Managing database evolutions
- ▶ Configuration
- ▶ Deploying your application

Additional documentation

- ▶ Scala ↗
- ▶ Akka ↗
- ▶ sbt ↗
- ▶ Configuration ↗
- ▶ Logback ↗

Books

Open source modules and plugins [11]

The screenshot shows a GitHub repository page for the project `linkedin / play-testng-plugin`. The repository is public and has 27 commits. The commit history shows several changes, including a fix for unconditional loading of `play.api.Logger` and refactoring. The repository also includes a `README.md` file.

Code

No description or homepage.

Clone in Mac, ZIP, HTTP, SSH, Git Read-Only, git@github.com:linkedin/play-testng-plugin.git, Read+Write access

branch: master, Files, Commits, Branches (1), Tags

play-testng-plugin / [+ 27 commits](#)

Fixed unconditional loading of class play.api.Logger that died horrib... [Dean Thompson] 4 months ago

latest commit 8d98ccf9d3

helpers 9 months ago Refactoring [jto]

plugin 4 months ago Fixed unconditional loading of class play.api.Logger that died horrib... [Dean Thompson]

project 4 months ago Fixed unconditional loading of class play.api.Logger that died horrib... [Dean Thompson]

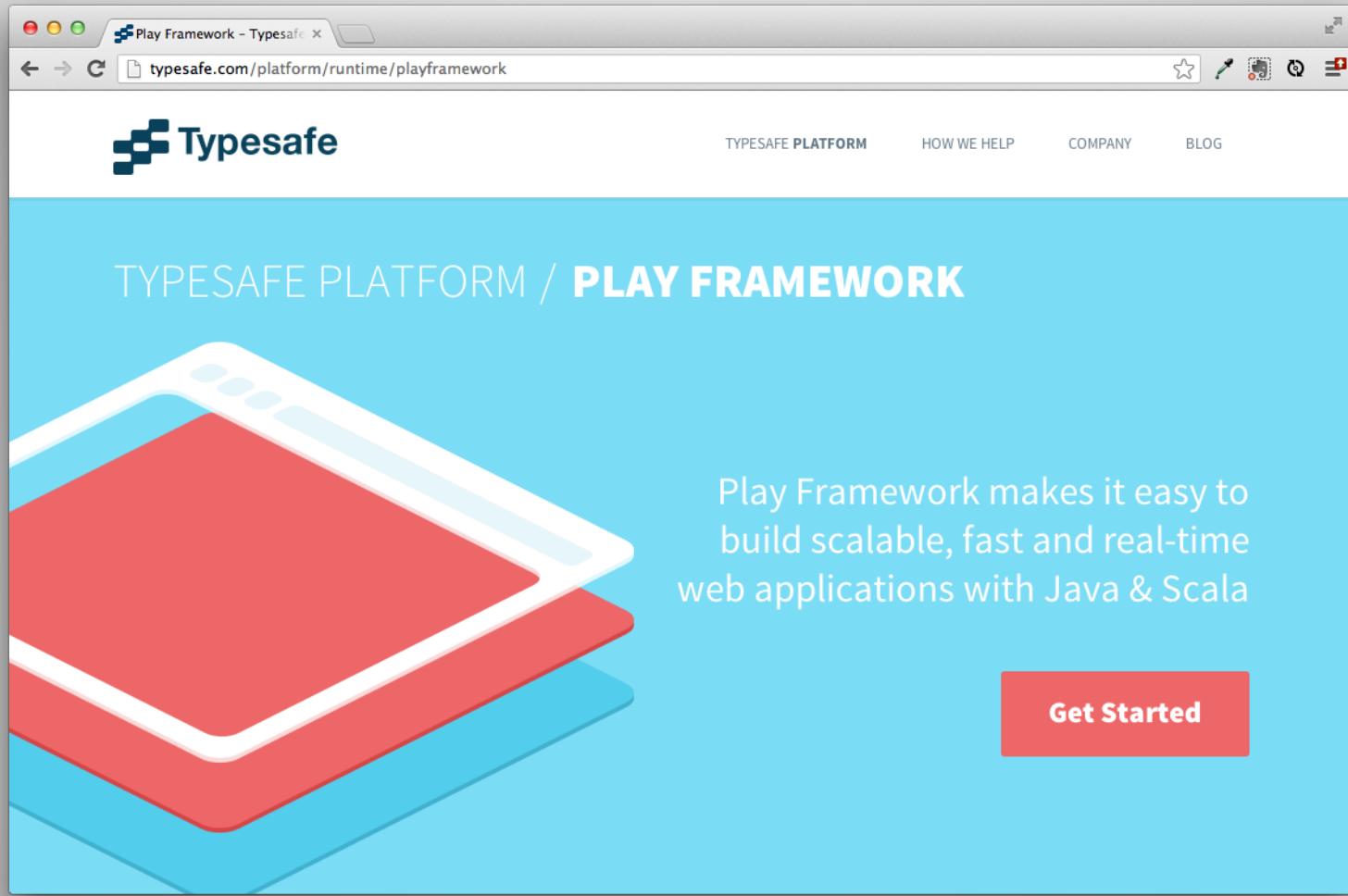
sample 9 months ago @WithTestServer [jto]

.gitignore a year ago first commit [jto]

README.md 10 months ago rename LITests to NGTests [jto]

README.md

We've open sourced a few of our plugins,
with many more on the way [12]



Commercial support from Typesafe [13]

Recap

1. Getting started with Play
2. Make a change and reload
3. Error handling
4. Threaded vs. evented
5. Non-blocking I/O
6. Scala
7. Performance
8. Community



That's the story of Play at LinkedIn so far...

But we're just getting started.

The screenshot shows the LinkedIn Engineering blog at engineering.linkedin.com. The page features a header with navigation links for Home, Projects, Technology, Team, Blog, LinkedIn Labs, Tech Talks, and Jobs. A Twitter follow button (@LinkedInEng) is also present. Below the header, there's a banner for the 'InMaps' product, showing a colorful network visualization and a team photo. To the right of the banner are social sharing counts: 362 LinkedIn shares, 218 Twitter tweets, and 241 Facebook likes. The main content area displays three blog posts:

- Play Framework: async I/O without the thread pool and callback hell** by Yevgeniy Brikman (posted 03/27/2013). The post discusses the Play Framework's asynchronous nature and how it avoids thread pool hell. It includes a bio for Yevgeniy Brikman and a link to read more.
- Engineering the New LinkedIn Profile** by Josh Clemm (posted 05/28/2013). This post covers the redesign of the LinkedIn profile and the use of various technologies like JSON mappers, Fizzy, and dust.js to achieve it.
- White Elephant: The Hadoop Tool You Never Knew You Needed** by Matthew Hayes (posted 03/20/2013). The post explores the use of Hadoop in organizations, focusing on scheduling, capacity planning, and billing.

On the right side of the page, there are two sidebar sections: 'LinkedIn Labs: Experimental & Hackday Projects' featuring Swarm, Infinity, and InMaps, and 'Work With Us at LinkedIn' listing job openings for Software Engineers across different application domains.

LinkedIn Engineering Blog [14]

LinkedIn Engineering (Link X)

Twitter, Inc. [US] https://twitter.com/LinkedInEng

Home Connect Discover Me

Tweets Following Followers Favorites Lists

Tweet to LinkedIn Engineering @LinkedInEng

582 TWEETS 271 FOLLOWING 9,764 FOLLOWERS Following

LinkedIn Engineering @LinkedInEng FOLLOW YOU
The engineering behind the world's largest professional network.
Mountain View, CA • engineering.linkedin.com

Tweets

LinkedIn Engineering @LinkedInEng 30 May
Beer, Pizza, and JavaScript Unit Testing tonight at LinkedIn.
meetup.com/Front-End-Deve...
[View summary](#)

LinkedIn Engineering @LinkedInEng 29 May
How we used client-side templating, JSON endpoints, and UI aggregation to build the new LinkedIn Profile.
engineering.linkedin.com/profile/engine...
[Expand](#)

LinkedIn Engineering @LinkedInEng 28 May
Engineering the New LinkedIn Profile.

@LinkedInEng on twitter [15]



Thank you!

References

1. <http://zeroturnaround.com/java-ee-productivity-report-2011/>
2. <http://static.springsource.org/spring/docs/2.5.x/api/org/springframework/...>
3. <http://engineering.linkedin.com/play/play-framework-linkedin>
4. http://www.eecs.berkeley.edu/~rcs/research/interactive_latency.html
5. <http://engineering.linkedin.com/play/play-framework-async-io-without...>
6. <http://netty.io/>
7. <https://github.com/playframework/Play20>
8. <http://mongodb-is-web-scale.com/>
9. <https://groups.google.com/forum/?fromgroups#!forum/play-framework>
10. <http://stackoverflow.com/tags/playframework>
11. <http://www.playframework.com/documentation/2.1.1/Modules>
12. <https://github.com/linkedin>
13. <http://typesafe.com/platform/runtime/playframework>
14. <http://engineering.linkedin.com/>
15. <https://twitter.com/LinkedInEng>