DataMapper on Infinispan

Clustered NoSQL

Lance Ball

strangeloop
·2011·

Lance Ball

- Red Hat senior engineer
- TorqueBox core developer
- Perl -> C++ -> Java -> Ruby
- @lanceball

project:odd

What are we talking about?

- DataMapper – Ruby ORM
- Infinispan – Java+Scala distributed cache
- Hibernate Search – Java ORM
- Lucene – Indexing and search
- JBoss AS7 – JEE application server
- TorqueBox – JRuby application server

TorqueBox

JRuby Application Server

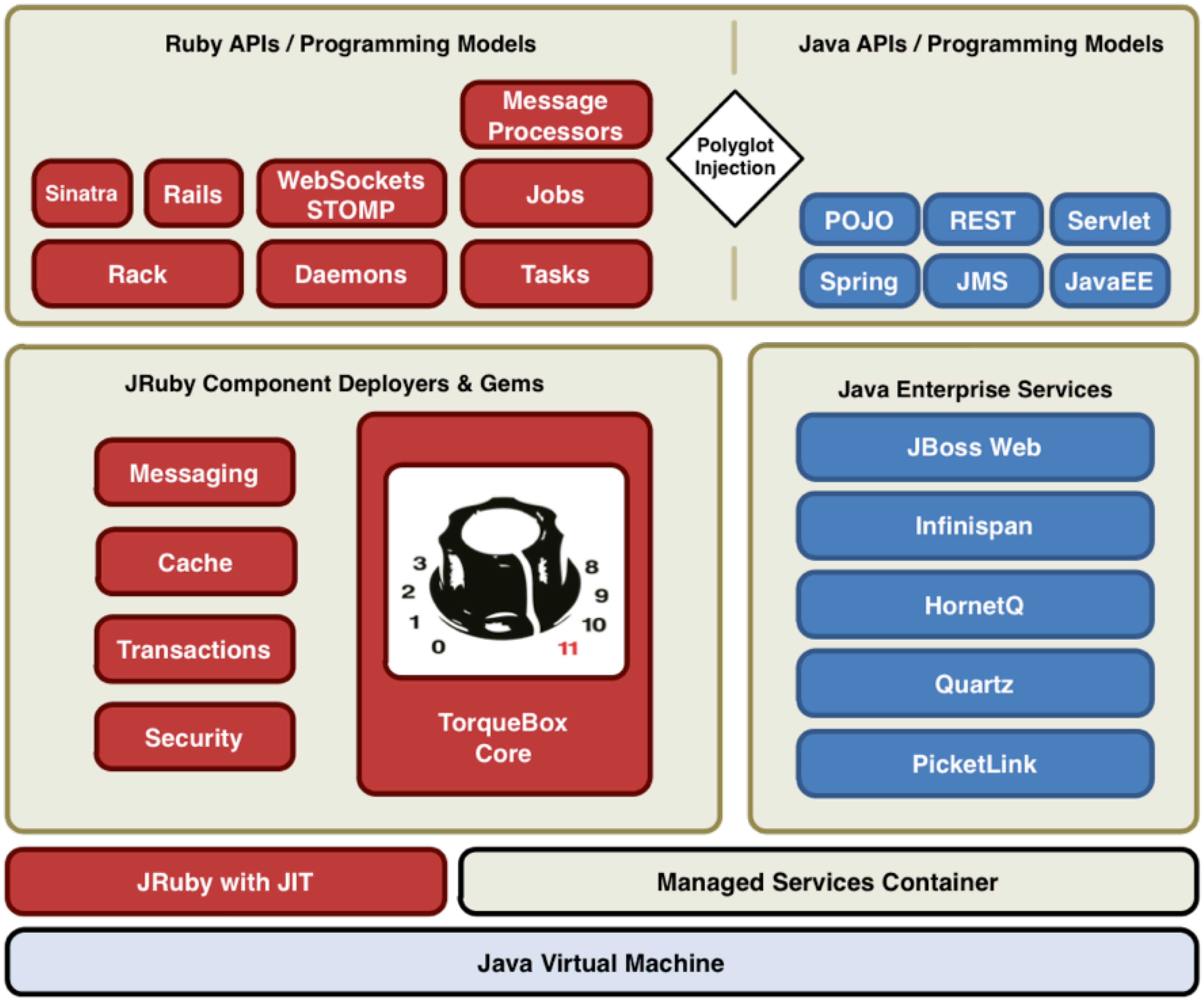
http://torquebox.org

TorqueBox

The **Power of JBoss**

with the

Expressiveness of Ruby


Ruby APIs / Programming Models
Java APIs / Programming Models
Message Processors
Polyglot Injection
Sinatra
Rails
WebSockets STOMP
Jobs
POJO
REST
Servlet
Rack
Daemons
Tasks
Spring
JMS
JavaEE
JRuby Component Deployers & Gems
Java Enterprise Services
Messaging
Cache
Transactions
Security
3
2
1
0
8
9
10
11
TorqueBox Core
JBoss Web
Infinispan
HornetQ
Quartz
PicketLink
JRuby with JIT
Managed Services Container
Java Virtual Machine

TorqueBox

- Ruby, baby!
- Rack apps
- Scheduled Jobs
- Background Tasks
- Message Queues & Topics
- Message Processors
- Long-running Services
- Distributed / Replicated Cache

JRuby

- Healthy community
- Real threads
- Java libraries
- Java tools
- Fast runtime
- Better memory management**

** for long running things like servers

Java

CacheFactory.java

```
public class CacheFactory {

  public CacheFactory() {
    config = new Configuration();
    store  = new FileCacheStoreConfig();
    store.purgeOnStartup( false );
    config.fluent().loaders().addCacheLoader( store );
    manager = new DefaultCacheManager( config.build() );
  }

  public Cache getCache() { return manager.getCache(); }
}
```

JRuby

cache_factory.rb

```
class CacheFactory
  def initialize
    @config = Configuration.new
    @store  = FileCacheStore.new
    @store.purge_on_startup false
    @config.fluent.loaders.add_cache_loader( @store )
    @manager = DefaultCacheManager.new( @config.build )
  end

  def get_cache ; @manager.get_cache end
end
```

Infinispan

An extremely scalable, highly available data grid

http://www.jboss.org/infinispan

Infinispan

- Key / Value store
- Highly concurrent core
- Data Grid
    - Replicated
    - Distributed
    - Local

Hotrod

- Binary TCP protocol
- Clients
  - Java
  - Python
  - Ruby
  - ...

NoSQL?

- Non-relational
- Scales out, not up
- Big data
- Low ceremony

Infinispan API

Cache.java

```
cache.put(key, value, lifespan, timeUnit);

cache.putIfAbsent(key, value, lifespan, timeUnit);

cache.replace(key, oldVal, value, lifespan, timeUnit);

cache.putAsync(key, value, lifespan, timeUnit);

cache.keySet();

cache.values();

cache.entrySet();
```

Hibernate Search

Example.java

```
   import org.hibernate.search.annotations.*;

@Indexed @ProvidedId
public class Book {
   @Field String title;
   @Field String description;
   @Field Date publicationYear;
}
```

Indexing: Lucene

Search.java

```
   org.apache.lucene.search.Query luceneQuery =
    queryBuilder.phrase()
                    .onField( "description" )
                    .andField( "title" )
                    .sentence( "pat the bunny" )
                    .createQuery();

CacheQuery query = searchManager.getQuery( luceneQuery,
                                            Book.class );
```

DataMapper

- Object Relational Mapper
- Alternative to ActiveRecord
- Written in Ruby
- http://datamapper.org

Resources

beer.rb

```
class Beer
  include DataMapper::Resource
  property    :id,      Serial
  property    :name,    String
  property    :rating,  Integer
  property    :notes,   Text
  belongs_to :user
end
```

DataMapper Queries

sample.rb

```ruby
Beer.all

Beer.get(1)

Beer.first( :name => 'Pisgah Pale' )

Beer.last( :name.like => 'IPA' )

Beer.all( :notes.like => 'hoppy' )
```

DataMapper Adapter SPI

sample_adapter.rb

```ruby
module DataMapper::Adapters

  class SampleAdapter < AbstractAdapter
    def initialize( name, options ) ; end
    def create( resources ) ; end
    def read( query ) ; end
    def update( attributes, collection ) ; end
    def delete( collection ) ; end
  end

end
```

DataMapper Filtering

some_adapter.rb

```
def read( query )
  records = @search_manager.search( query )
  query.filter_records( records )
end
```

Testing

adapter_spec.rb

```
require 'dm-core/spec/shared/adapter_spec'
describe DataMapper::Adapters::InfinispanAdapter do

  before :all do
    @adapter = DataMapper.setup(:default,
                                :adapter => 'infinispan')
  end

  it_should_behave_like 'An Adapter'

  describe "other important things to test" do
    # Your tests here
  end
end
```

torquebox-cache

- TorqueBox 2.0 gem
- dm-infinispan-adapter
- TorqueBox::Infinispan::Cache
- ActiveSupport::Cache::TorqueBoxStore

dm-infinispan-adapter

Use Infinispan as your object data store

dm-infinispan-adapter

```
require 'dm-core'
require 'dm-infinispan-adapter'

class Beer
  include DataMapper::Resource
  property :id,      Serial
  property :name,    String
  property :rating,  Integer
  property :notes,   Text
  belongs_to :user
end

DataMapper.setup(:default,
                 :adapter=>'infinispan', :persist=>true)
```

But How?

Hibernate Search

Annotated Java classes

Runtime Class Creation

How do we make Ruby's `Beer.class` look like an annotated Java class at runtime?

Metaprogramming!

dm-infinispan-adapter.rb

```
require 'datamapper/model'

module DataMapper::Adapters

  class InfinispanAdapter < AbstractAdapter

    DataMapper::Model.append_inclusions( Infinispan::Model )

  end
end
```

Metaprogramming!

datamapper/model.rb

```
module Infinispan
  module Model

    def self.included(model)
      model.extend(ClassMethods)
      model.before_class_method(:finalize, :configure_index)
    end

  end
end
```

Annotations

datamapper/model.rb

```
require ‘jruby/core_ext’

annotation = {org.hibernate.search.annotations.Field => {}}
add_method_annotation( “getName”, annotation )
```

Annotations

datamapper/model.rb

```
require ‘jruby/core_ext’

annotation = {
  org.hibernate.search.annotations.Indexed => {},
  org.hibernate.search.annotations.ProvidedId => {},
  org.infinispan.marshall.SerializeWith => {"value" =>
org.torquebox.cache.marshalling.JsonExternalizer.java_class }}

add_class_annotation( annotation )
```

Become Java!

```
java_class = become_java!
```

JSON Externalizer

Forget

`java.io.Serializable`

JSON Externalizer

JsonExternalizer.java

```java
public class JsonExternalizer
    implements Externalizer<IRubyObject> {

   @Override
   public void writeObject(ObjectOutput output,
                           IRubyObject object)
        throws IOException {
      String theType = object.getType().getName();
      output.writeObject( theType );
      output.writeObject( toJSON(object) );
   }
 }
```

JSON Externalizer

JsonExternalizer.java

```
public class JsonExternalizer implements
Externalizer<IRubyObject> {

   @Override
   public IRubyObject readObject(ObjectInput input)
           throws IOException, ClassNotFoundException {
      String theType = (String) input.readObject();
      String theJson = (String) input.readObject();
      return fromJSON(theJson, theType);
   }
 }
```

JSON Externalizer

JsonExternalizer.java

```
public class JsonExternalizer
    implements Externalizer<IRubyObject> {

    protected IRubyObject fromJSON(String json,
                                   String type)
              throws ClassNotFoundException {

        RubyModule objectClass = runtime.getClassFromPath( type );
        return (IRubyObject) JavaEmbedUtils.invokeMethod( runtime,
                                                          objectClass, "new",
                                                          new Object[] { jsonHash },
                                                          IRubyObject.class);
    }
 }
```

JSON Externalizer

JsonExternalizer.java

```java
public class JsonExternalizer implements Externalizer<IRubyObject> {

    protected String toJSON(IRubyObject object) {
        return (String) JavaEmbedUtils.invokeMethod(
                            getCurrentRuntime(), object, "to_json",
                            EMPTY_OBJECT_ARRAY, String.class );
    }
 }
```

Sequences

dm-core/property/serial.rb

```
DataMapper::Property::Serial
```

Sequences

dm-core/adapters/abstract_adapter.rb

```
initialize_serial( resource, next_id )
```

Sequences

cache.rb

```
module TorqueBox
  module Infinispan

    class Cache
      def increment( sequence_name, amount = 1 )
        # increment an integer
      end
      def decrement(name, amount = 1)
        # decrement an integer
      end
    end

  end
end
```

Sequences

dm-infinispan-adapter.rb

```
@metadata = Cache.new( options )

initialize_serial( resource,
  @metadata.increment( "metadata/beers/index" ) )
```

Transactions

cache_spec.rb

```ruby
describe "with JTA transactions" do

  it "should behave like a transaction" do
    @cache.transaction do |cache|
      cache.put('Tommy', 'Dorsey')
      raise "yikes!"
      cache.put('Elvis', 'Presley')
    end
    @cache.get('Tommy').should be_nil
    @cache.get('Elvis').should be_nil
  end

end
```

Transactions

dm-infinispan-adapter.rb

```
def delete( collection )
  cache.transaction do
    collection.each do |resource|
      cache.remove( key(resource) )
    end
  end
end
```

TorqueBox::Infinispan::Cache

Use Infinispan for...

Cache

some_file.rb

```ruby
include TorqueBox::Infinispan::Cache

cache = Cache.new(:name => 'MyCache',
                  :mode => :replicated)

cache.put(key, value)
cache.get(key)
```

ActiveSupport::Cache::TorqueBoxStore

Caching in Rails.

Replaces in-memory or **memcached** caches.

TorqueBoxStore

config/application.rb

```ruby
module YourApp

  class Application < Rails::Application
    config.cache_store = :torque_box_store
  end

end
```

TorqueBoxStore

my_app.rb

```
require 'sinatra'
require 'torquebox'

class MyApp < Sinatra::Base

  use TorqueBox::Session::ServletStore

  get '/' do
    session[:message] = 'Hello World!'
    haml :index
  end

end
```

Beer Catalogue!

http://www.flickr.com/photos/burnblue/308441464/

Model

beer.rb

```ruby
require 'dm-core'
require 'dm-infinispan-adapter'

class Beer
  include DataMapper::Resource
  property :id, Serial
  property :name, String
  property :rating, Integer
  property :notes, Text
  belongs_to :user
end

DataMapper.setup(:default, :adapter=>'infinispan',
                           :persist=>true)
```

Sinatra

application.rb

```ruby
module BeerCatalogue
  class Application < Sinatra::Base
    get '/' do
      @beers = Beer.all( :user_id => current_user.id )
      haml :index
    end

    post '/beer' do
      Beer.create(
        :name=>params[:name], :notes=>params[:notes],
        :rating=>params[:rating], :user=>current_user)
      redirect '/'
    end
  end
end
```

View

views/index.haml

```haml
#welcome
  Hello
  =current_user.name

#body
  #beer-list
    %h2 Your Beers
    - if @beers.empty?
      %strong You haven't rated any beers yet. Do that now!
    %ul
      - @beers.each do |beer|
        %li
          =beer.name
          =beer.rating
          =beer.notes
```

Source

http://github.com/torquebox/torquebox

http://github.com/torquebox/presentations

Installation

```
$ gem install torquebox-server --pre \
    --source http://torquebox.org/2x/builds/LATEST/gem-repo/

$ gem install bundler

$ bundle install

$ torquebox deploy .
Deployed: beer.yml
    into: /path/to/jboss/standalone/deployments

$ torquebox run
```

http://www.flickr.com/photos/crystalflickr/2317183342/

TorqueBox 2.x

- Under development
- Continuous integration
- TorqueBox::Infinispan::Cache
- ActiveSupport::Cache::TorqueBoxStore
- dm-infinispan-adapter

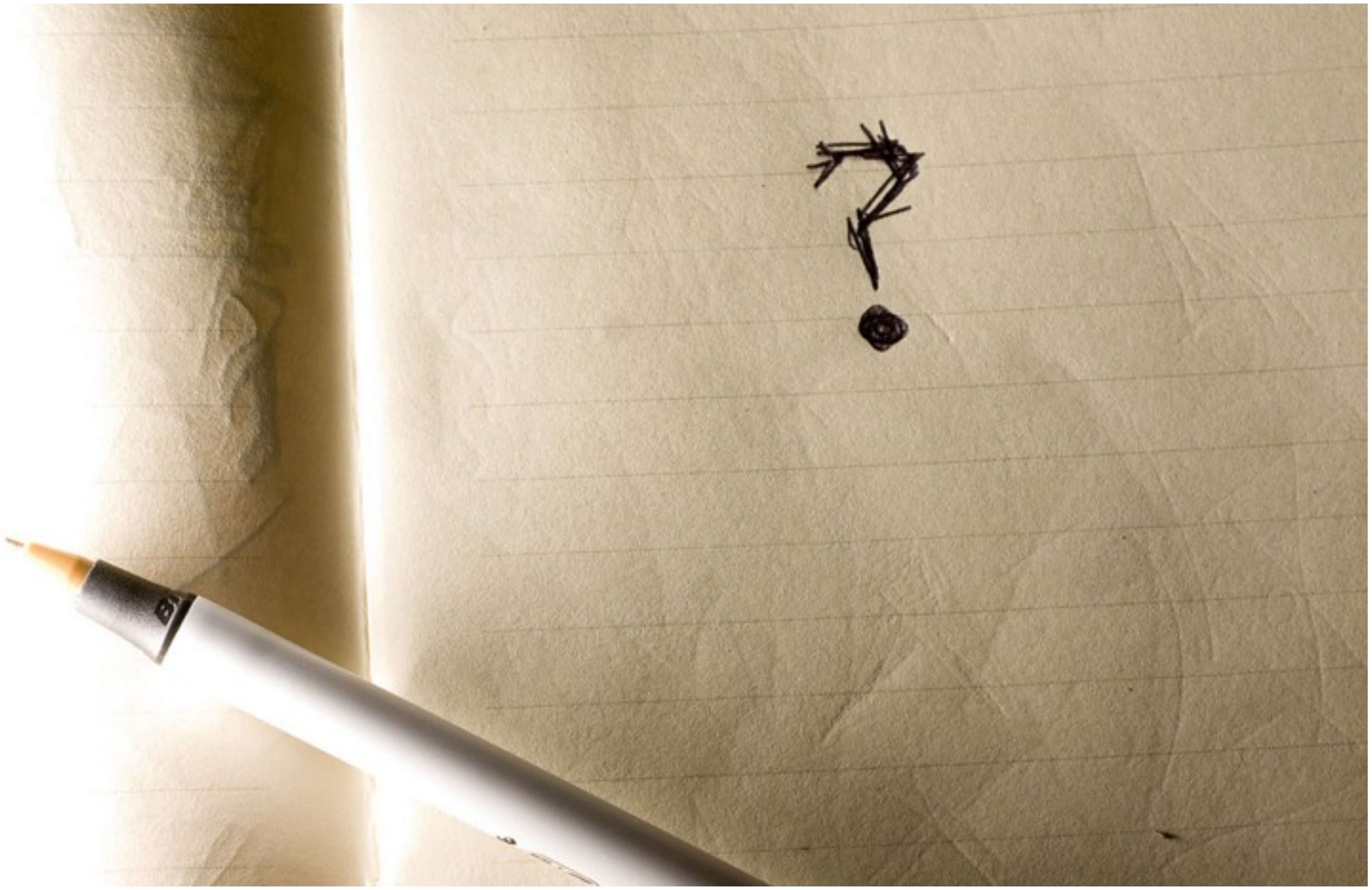
TorqueBox 1.x

- Version 1.1.1 released in August
- Continuous integration
- ActiveSupport::Cache::TorqueBoxStore

Resources

- http://torquebox.org/
- http://github.com/torquebox
- #torquebox on FreeNode
- @torquebox

Questions



http://www.flickr.com/photos/eleaf/2536358399/