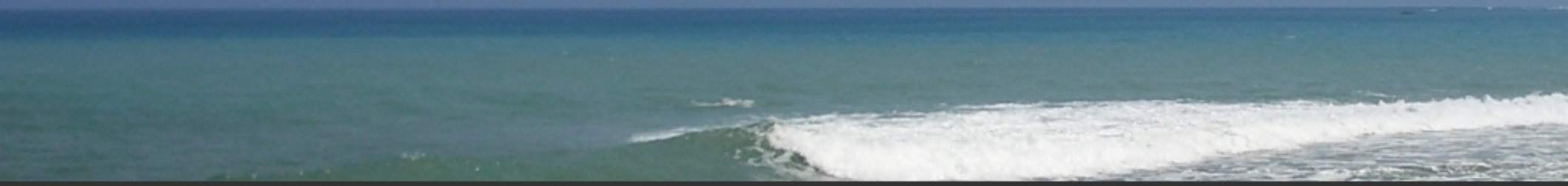
“Scaling Rails”

Pagal “Scaling Rails Screencasts”

Saulius Grigaitis
RubyOnRails LT



Serveris

NEBEPAVEŽA!!!

Perkam daugiau serverių

Server IAI

PAVEŽA!!!

Klausimai

?

Po savaitės ...

Vis dar švenčiam problemos sprendimą..

tinklapis vis populiarėja...

Server IAI

NEBEPAVEŽA!!!

Naujų serverių nebe apsimoka pirkti,
O dar krizė...pinigų nėra...

“Scaling Rails”

Kodėl lėtai kraunasi?

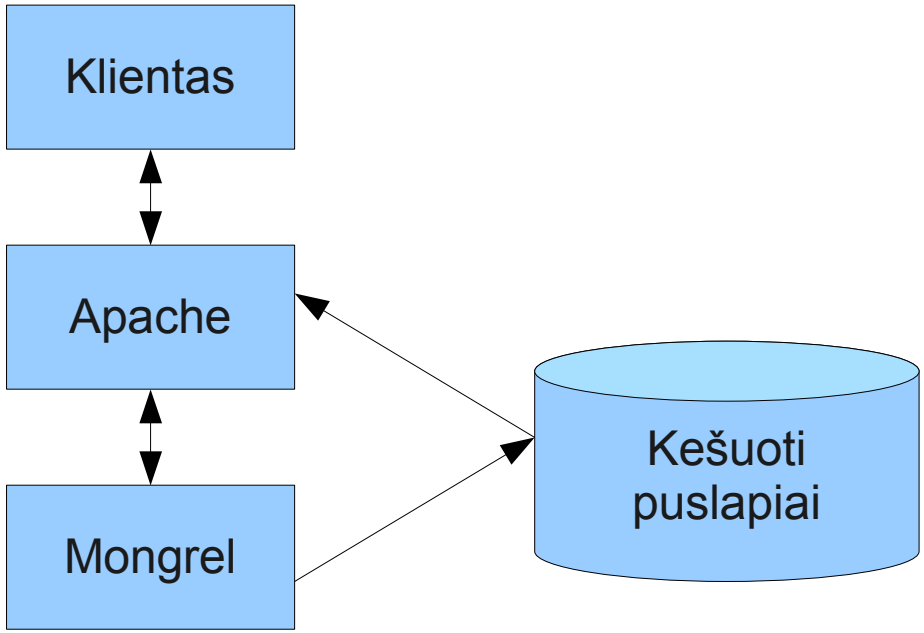
FireBug
YSlow

- <%= javascript_include_tag :all, :cache => true %>
- <%= stylesheet_link_tag :all, :cache => true %>

- Asset Packager

- ActionController::Base.asset_host = [“assets1.example.com”, “assets2.example.com”]

Puslapio kešavimas (“page caching”) - veikimo schema



Puslapio kešavimas - naudojimas

```
# config/environment.rb
config.action_controller.perform_caching = true

# app/controller/posts_controller.rb
...
caches_page :index, :show
...
def update
  expire_page :action => :index
  expire_page :action => :show, :id => @post
end
```

Puslapio kešavimas – formatai

- Failai kešuojami pagal formatą, pvz:
  - public/posts.html
  - public/posts.xml
- Galima apsibrėžti ir savo “mimetype”

Puslapio kešavimas – ignoruoja CGI parametrus

- Šios užklausos grąžins vieną ir tą patį:
  - www.example.com/posts.html
  - www.example.com/posts.html?logged_in=true
  - www.example.com/posts.html?category_id=3
- Sprendimas – pertvarkyti route'us:
  - map.post_with_category “posts/:category_id”, :controller => 'posts', :action => 'index'

Sweeper'iai

```
# sweepers/post_sweeper.rb
class PostSweeper < ActionController::Caching::Sweeper
  observe Post
  def after_save(post)
     clear_posts_cache(post)
  end
  def clear_posts_cache(post)
      expire_page :controller => :posts, :action => :index
      expire_page :controller => :posts, :action => :show, :id => post.id
   end
end
# app/controllers/posts_controller.rb
...
cache_sweeper :post_sweeper, :only => [:create, :update, :destroy]
...
```

Progresinis kešavimas

- Personalizuojame kešuotą puslapį panaudojant AJAX
- Tinka, jei puslapiai nėra per daug personalizuoti
- Atsinaujinantis vaizdas gali erzinti vartotoją

Puslapio, veiksmo (“action”) ir fragmento kešo talpyklos

- Puslapio kešas:
  - Failų sistemoje („file_store“)
- Veiksmo ir fragmento kešas:
  - Atmintis („memory_store“)
  - Sinchronizuota atmintis (“synchronized_memory_store”)
  - Failų sistemoje (“file_store”)
  - DRB (“drb_store”)
  - Memcached (“mem_cache_store”)
  - Kompresuotas Memcached (“compressed_mem_cache_store”)
  - Sava talpykla (“custom_store”)

Veiksmo kešavimas

- Panašu į puslapio kešavimą
- Vykdomi filtrai
- Dažnai naudojama autorizacijai
- caches_action :index, :show, :if => Proc.new {
  |controller| !controller.params[:no_cache] }

Fragmento kešavimas

```
<% cache(:”recent_posts-#{page_number}”) do %>
   ...
<% end %>

if !fragment_exists? :recent_posts
    @recent_posts = Post.find(:all, ...)
 end
```

Fragmento kešavimas – 4 metodai

- write_fragment(key, content, options = nil)
- read_fragment(key, options = nil)
- fragment_exists?(key, options = nil)
- expire_fragment(key, options = nil)

Memcached

- Saugo atmintyje
- Išskirstyta
- Turinys pasiekiamas panaudojant raktą
- Trumpai - { } atmintyje

Memcached - naudojimas

- config.cache_store = :mem_cache_story
- Dažniausiai naudojama Rails'uose:
  - Objektų saugojimui
  - Veiksmo ir fragmentų kešui

Memcached - naudojimas

- Pasiekiamas:
  - Rails.cache.read
  - Rails.cache.write
  - Rails.cache.fetch
  - Rails.cache.delete
  - Rails.cache.exists?
  - Rails.cache.increment
  - Rails.cache.decrement
  - Rails.cache.decrement
- Rails.cache.fetch('current_time', :expires_in => 30.second) { Time.now }

Memcached – kešuoto objekto galiojimo nutraukimas

- Panaudojant “:expires_at” (galima ir „view“e)
- Seniausi objektai išmetami automatiškai, tad:

<% cache(“#{post.id}-#{post.updated_at.to_i}-data”) do %>

.....

<% end %>

- Arba tiesiog:

<% cache(post) do %> #kvies „cache_key“ metodą

ActiveRecord - Nepamirštam indexų

```
# db/migrations/001_add_posts.rb
....
add_index :posts, :user_id
```

ActiveRecord – Nepamirštam “include”

- N + 1 problema
- Post.find(:all, :include => :user)

ActiveRecord – QueryTrace įskiepis

- Padeda surasti kur generuojami SQL sakiniai Rais apikacijoje

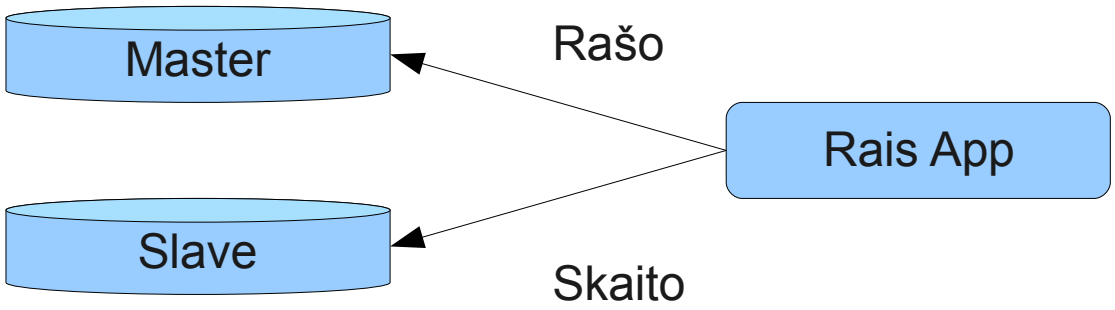
ActiveRecord – kai reikia rašyti gryną SQL

- Post.connection.execute(...)
- Post.find_by_sql(...)

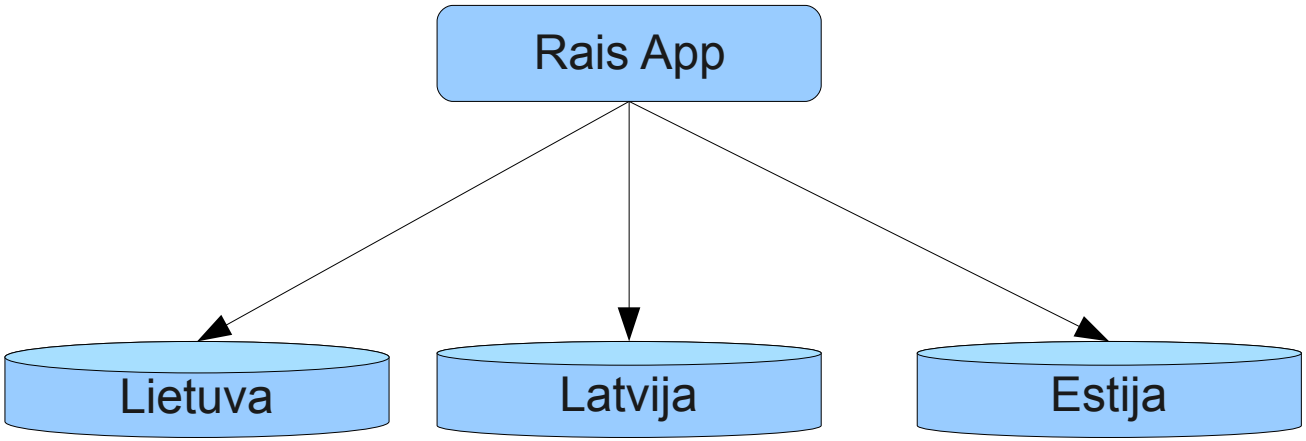
ActiveRecord – denormalizacija

- Galima paaukoti ir tai ;)

ActiveRecord – Master Slave schema (Masochism)



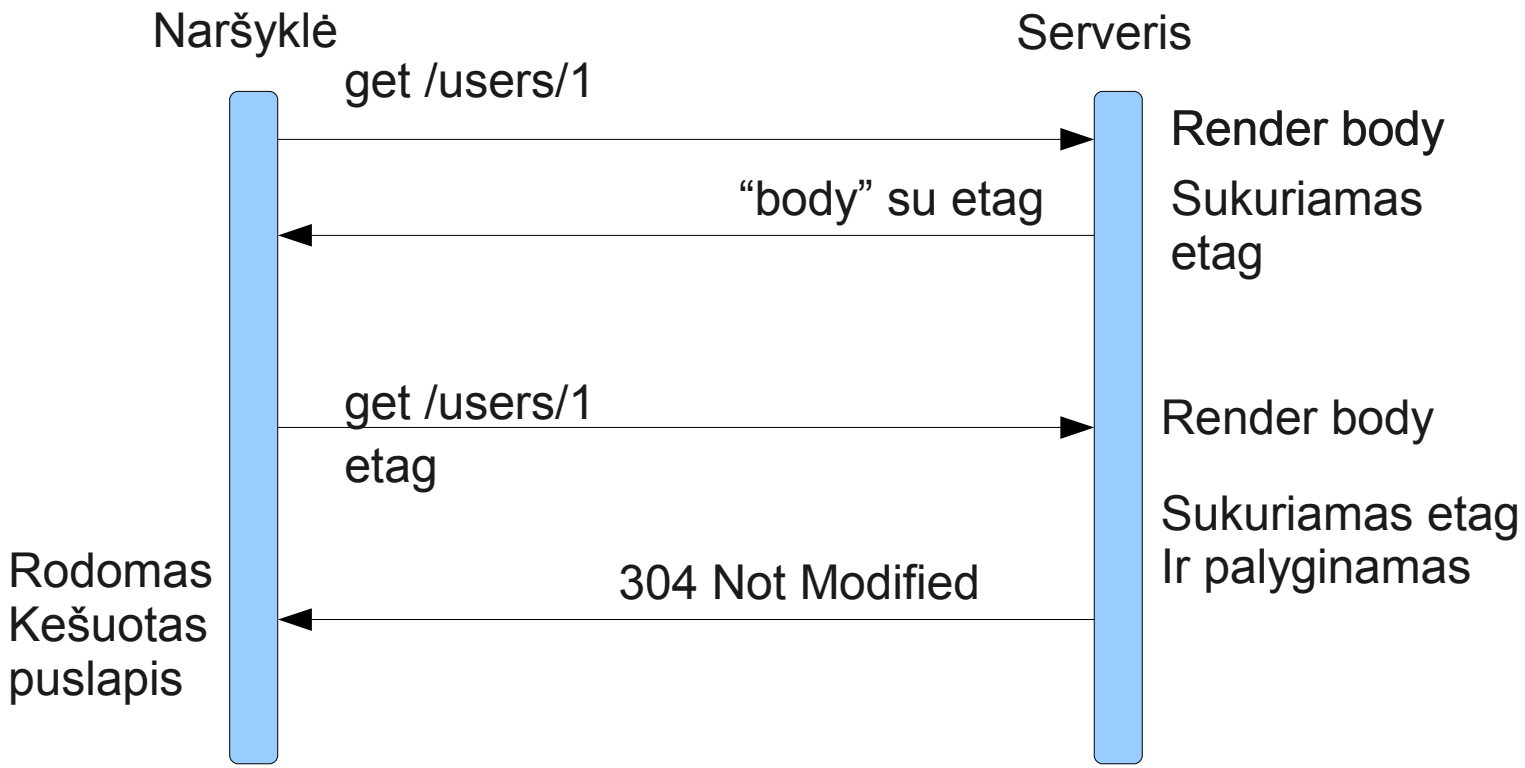
ActiveRecord – Sharding (Data Fabric)



Kešavimas kliento pusėje (Client side caching)

- 3 antraštės (“headers”)
  - max-age - “expires_in 10.minutes”
  - etag
    - “stale?”
    - “fresh_when”
    - MD5 arba savo raktas
  - last_modified - “:last_modified” atributas “stale?” ir “fresh_when” metodams

Kešavimas kliento pusėje - etag



Reverse Proxy Cache

- Rack::Cache
- Varnish
- Squid
- Akamai
- ...

Ir dar daug galimybių...

Klausimai

?