

WHY IS THIS STUFF SO SLOW?

Ruby and Rails “Performance”



@futuremint

Dave Woodward

Making your code go faster is really easy!
Just do this... 1. find the part that is slow
2. stop doing that!



@futuremint Dammit, now what am I supposed to talk about tomorrow?

RUBY IS **SLOW**

- At Garbage Collection
- When Creating Lots of Objects
- When Doing Stupid Stuff
- At a **lot** of other things...

HOW TO MAKE IT **FASTER**

- Don't do things that are slow
- Use a faster Ruby implementation
- Do things later
- Don't do stupid things

DON'T DO **SLOW** THINGS

- N+1 (or 2, or 3, or 20)
- In-memory data manipulations
- Things that create lots of or large objects
- Things that make external requests

FIXING **N+1** (OR MORE)

- **Bullet** Gem (can be slow)
- **DataMapper**
- Load with **:include**
- **Flatten** Data
- **Cache** Things

BULLET

- **<https://github.com/flyerhzm/bullet>**

```
2009-08-25 20:40:17[INFO] N+1 Query: PATH_INFO: /posts;      model: Post  
=> associations: [comments].
```

```
Add to your finder: :include => [:comments]
```

```
2009-08-25 20:40:17[INFO] N+1 Query: method call stack:.
```

```
/Users/richard/Downloads/test/app/views/posts/index.html.erb:11:in
```

```
`_run_erb_app47views47posts47index46html46erb'
```

```
/Users/richard/Downloads/test/app/views/posts/index.html.erb:8:in `each'
```

```
/Users/richard/Downloads/test/app/views/posts/index.html.erb:8:in
```

```
`_run_erb_app47views47posts47index46html46erb'
```

```
/Users/richard/Downloads/test/app/controllers/posts_controller.rb:7:in
```

```
`index'
```


DATAMAPPER

<http://datamapper.org/why>

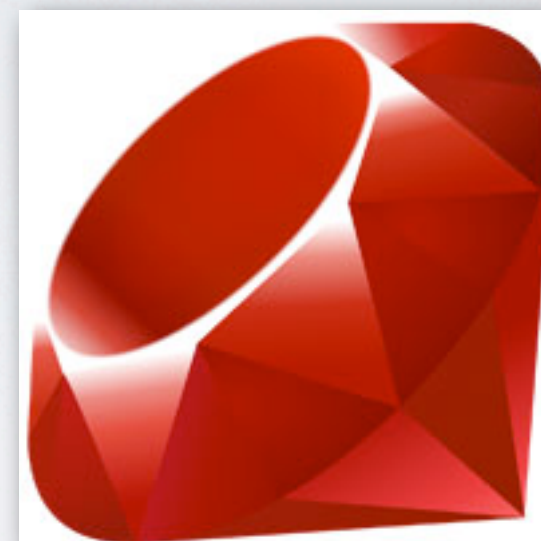
```
1 zoos = Zoo.all

2 zoos.each do |zoo|
3   # on first iteration, DM loads up all of the exhibits for all of
the items in zoos
4   # in 1 query to the data-store.
5
6   zoo.exhibits.each do |exhibit|
7     # n+1 queries in other ORMs, not in DataMapper
8     puts "Zoo: #{zoo.name}, Exhibit: #{exhibit.name}"
9   end
10 end
```

IN-MEMORY DATA **MANIPULATION**

- Sort, Match, Group_by, etc
- **Fine** in *small* doses
- Create more objects for **GC** to clean up
- Use **Bang** methods or avoid when feasible
- Like when your **Database** can do the work

USE A **FASTER** RUBY



Ruby **1.9**

JRUBY



DO STUFF **LATER**

- **AJAX** In complex (slow) data
- Do N+1 **on request**
- Use `delayed_job`, `resque`, etc to **defer** work

BENCHMARK THINGS

demo

WARNING!

- Don't sacrifice your **sanity** for performance
- Stay away from **really crazy** performance tricks
- Keep **Separation** of Concerns