## 500px

## What 500px Engineering Learned from Upgrading Rails 3.2 -> 4.2.5

#500pxTechTalks

Kevin McLoughlin Jr. Web Developer

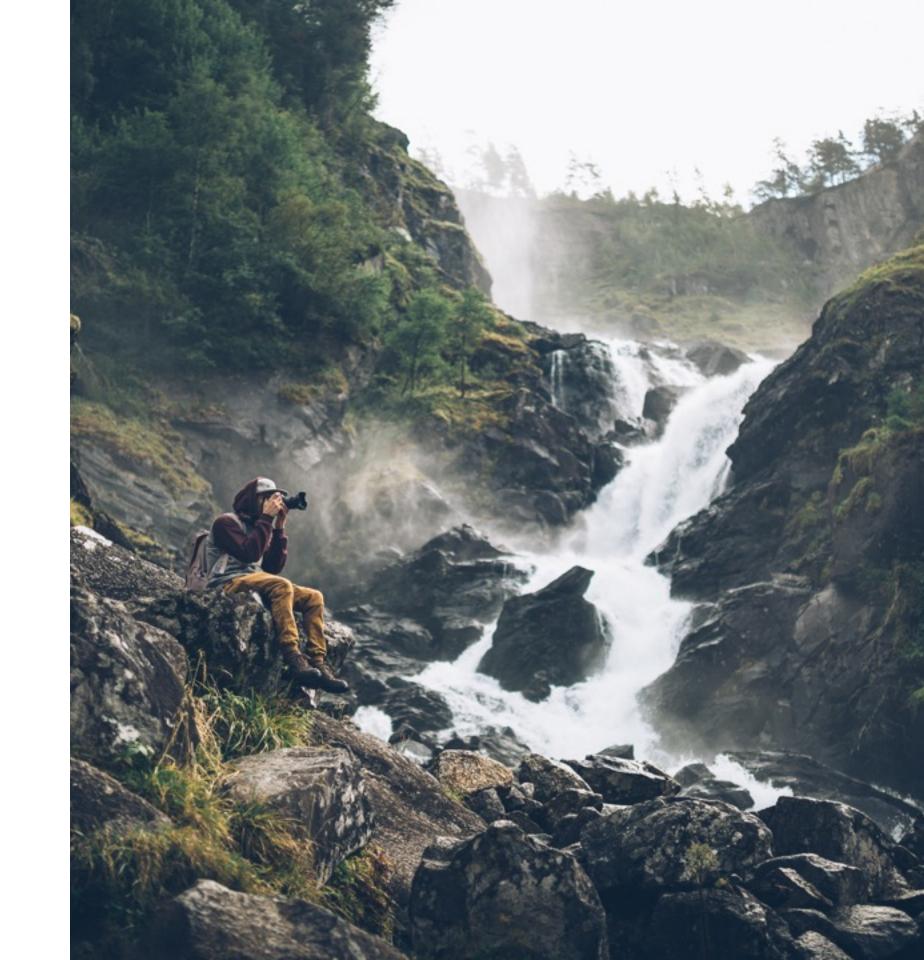


## Me

**Junior Web Developer** 

Hired by 500px Oct. 2015

Github: @trustyknave







The process we engaged in as an organization.

The technical challenges we overcame along the way.





# Process Retrospective Resources



## **Project Team**

Sr. Software Eng.



Kevin

Software Dev



Devon

Jr.Web Dev



Kevin

QA Lead



Artem

Project Lead



Chris

Web Dev



Michael

PMO



Aneta

Web Lead



Zimu

Sr.Web Dev



Ryan

#### **Our Process**

#### Phase 1: Investigation

Goal: All tests pass

#### Phase 2: Implementation

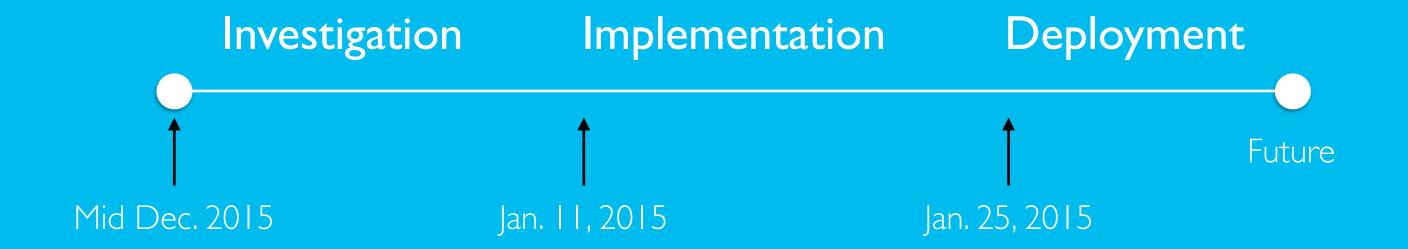
Goal: No known bugs remaining

#### Phase 3: Deployment

Goal: Equivalent or better performance



## Timeline





## Phase 1: Investigation

Goal: All tests pass





# Know the state of your system



## A comprehensive test suite is necessary and invaluable.





It aligns everyone on the value of the project, and the steps that exist between your current and future states.





## Rails 3.2 to 4.2.5 Upgrade Plan

Certain specifics have been removed to maintain confidentiality.

## Phase 1: Investigation

**Goal: All tests pass** 

Step 1: Get your environment running

- Update your Gemfile to use Rails 4.2.5 and compatible versions of gems.
- Make database creation, migration, and seeding work.
- · Correct all code violating Rails syntax/security checks so we can start a console.
- · Get sidekiq workers to start (okay if errors are reported).
- · Get test environment to run (okay if specs fail).



#### This will take time

```
208 Gemfile
                                                                                                                                                                       View
... @@ -1,111 +1,119 @@
 1 source "http://rubygems.org"
                                                                                             source "http://rubygems.org"
      #ruby '2.1.0'
                                                                                              #ruby '2.1.0'
 4 -gem 'rake', '~>10.1'
                                                                                            +gem 'rake', '~> 10.4', '>= 10.4.2'
 5 -gem 'rails', '~>3.2.22'
                                                                                         5 +gem 'rails', '~> 4.0', '>= 4.0.13'
 7 # Rails extensions
                                                                                             # Rails extensions
 8 -gem "validates_existence", "~> 0.8.0"
                                                                                             +gem 'protected_attributes', '~> 1.0', '>= 1.0.3'
 9 -gem "strong parameters", "~> 0.2.1"
                                                                                        10 +gem "validates_existence", "~> 0.8"
-gem "activerecord-mysql-index-hint", "~> 0.0.2"
                                                                                        11 +gem 'actionpack', '~> 4.0'
                                                                                        12 +# gem "strong_parameters", "~> 0.2"
                                                                                        13
# Storage Engines (DB, CDN, CACHE, ETC)
                                                                                             # Storage Engines (DB, CDN, CACHE, ETC)
13 -gem 'mysql2', '~> 0.3.20'
                                                                                        15 +gem 'mysql2', '~> 0.3'
                                                                                        16 +gem "activerecord-mysql-index-hint", "~> 0.0.2"
14 -gem 'seamless_database_pool', :git =>
     'https://github.com/500px/seamless_database_pool.git'
15 -gem 'dalli', '~> 2.6.2'
                                                                                             +gem 'seamless_database_pool', '~> 1.0'
16 -gem 'hiredis'
17 -gem 'mongoid', '~> 2.8'
                                                                                             +gem 'dalli', '~> 2.6'
18 -gem 'bson_ext', '~> 1.9'
                                                                                           +gem 'hiredis', '~> 0.6'
19 -gem 'splunk-sdk-ruby', '=1.0.2'
                                                                                            +gem 'mongoid', '~> 4.0', '>= 4.0.2'
                                                                                             +gem 'splunk-sdk-ruby', '~> 1.0'
20 -
21 -gem 'aws-sdk', '~> 2.0.0'
22 -gem 'edgecast_api'
                                                                                        24 +gem 'aws-sdk', '~> 2.0'
23 -gem 'fastly'
                                                                                        25 +gem 'edgecast_api', '~> 0.0.1'
                                                                                        26 +gem 'fastly', '~> 1.2'
```

Because we introduced version locking during the project, almost every one of our dependencies updated by at least 1 minor version.

#### The 20/80

#### The 20% of issues that took 80% of our time

- Mongoid went from 2.8 -> 4.0.2, and introduced a host of syntax changes and new dependencies for us to investigate and understand.
- New ActiveRecord query syntax.
- · Scopes now using callable objects like a Proc or Lambda e.g.

```
• scope :active, -> { where active: true }
```



## Phase 1: Investigation

**Goal: All tests pass** 

Step 2: Get all tests to pass

- · Model specs: 2982 examples, 96 failures, 910 pending
- · Controller specs: 4142 examples, 1145 failures, 15 pending
- · Worker specs: 153 examples, 15 failures
- Request specs: 731 examples, 147 failures



#### This will take time

```
Thread #3 2 more setup commands +
                                                                                                                                             Launch SSH
  export THREADNUM=3; ./script/semaphore.sh
                                                                                                                                                  10:10
    Comment
      0.0706 seconds average (0.0706 seconds / 1 example) ./spec/models/comment_spec.rb:3
    Api::V1::Admin::PhotosController
      0.06626 seconds average (0.26504 seconds / 4 examples) ./spec/controllers/api/v1/admin/photos_controller_spec.rb:3
    Api::V1::Editors::PhotosController
      0.06416 seconds average (0.25666 seconds / 4 examples) ./spec/controllers/api/v1/editors/photos_controller_spec.rb:3
    HomeController
      0.04612 seconds average (0.04612 seconds / 1 example) ./spec/controllers/home controller spec.rb:3
  Finished in 5.19 seconds
  34 examples, 34 failures
  Failed examples:
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:422 # Api::V1::FavoritesController#index sorting no sort param provided returns users :
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:434 # Api::V1::FavoritesController#index sorting sort by friends returns friends on top
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:452 # Api::V1::FavoritesController#index when requesting with "include_following" flag
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:458 # Api::V1::FavoritesController#index when requesting with "include_following" flag
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:466 # Api::V1::FavoritesController#index when requesting with "include_following" flag
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:242 # Api::V1::FavoritesController#destroy when logged in when the photo is favorited h
  rspec ./spec/controllers/api/v1/favorites controller spec.rb:232 # Api::V1::FavoritesController#destroy when logged in when the photo is favorited (
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:225 # Api::V1::FavoritesController#destroy when logged in when the photo is favorited h
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:309 # Api::V1::FavoritesController authenticated requests Favorites manipulation GET fa
  rspec ./spec/controllers/api/v1/favorites_controller_spec.rb:320 # Api::V1::FavoritesController authenticated requests Favorites mani LINE WRAP OFF
```

#### The 20/80

#### The 20% of issues that took 80% of our time

- New Mongoid query syntax changes.
- No more implicit conversion of ActiveSupport:: TimeWithZone to a Ruby Time object.
- New Rails caching structure.
- New JSON serialization structure.
- Devise compatibility issues



## Phase 1: Investigation

**Goal: All tests pass** 

Step 3: Collectively make a go/no-go decision

- · List your red flags and check your progress against understanding their potential impacts.
- Make a manual test plan to complement your suite of automated tests.
- Make a deployment plan that includes disaster recovery scenarios.



## Phase 2: Implementation

Goal: No known bugs remaining



### Phase 2: Implementation

#### Goal: No known bugs remaining

Concurrent execution of manual test plan and debugging

- · Get onto an environment that reasonably mimics Production as quickly as possible.
- Do as little as possible. We used the protected\_attributes and activerecord-deprecated\_finders gems to reduce the amount of work.
- · Most of the significant issues we encountered were very specific to our codebase.



#### The 20/80

#### The 20% of issues that took 80% of our time

- acts-as-taggable-on
- · Issues arising from the changes to Rails sessions.
- Rebasing against master.
- Deployment preparation.



## Phase 3: Deployment

Goal: Equivalent or better performance



## Phase 2: Deployment

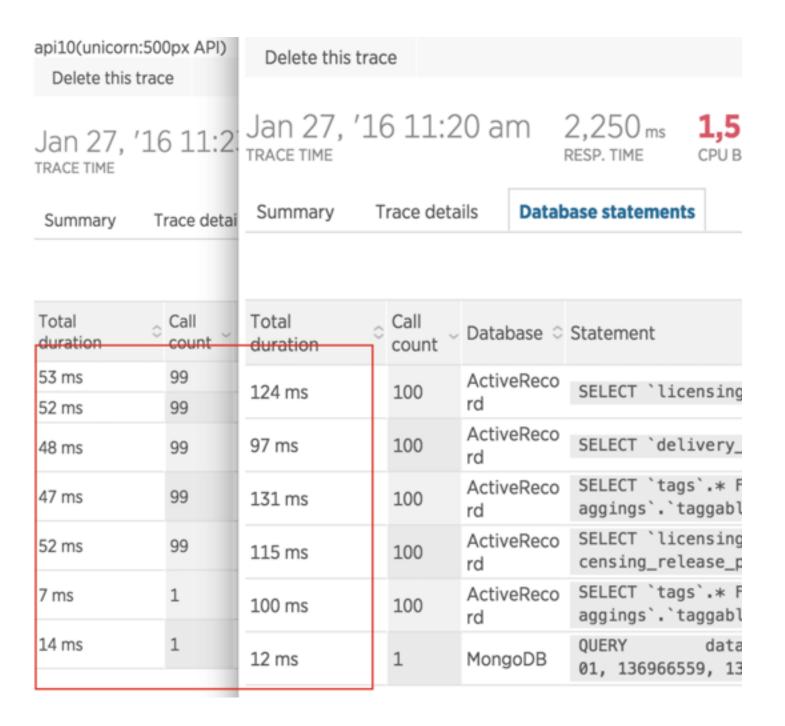
#### Goal: Equivalent or better performance

Slow rollout of Rails4 branch to the various sections of our infrastructure

- · We're 100% rolled out.
- We've seen a significant reduction in performance on our API servers.
- · Current solution is more hardware, fewer workers per machine, and incremental upgrades to different parts of our system to regain lost performance.
- · Some good news. On Rails 4.2.5, 50% of the workers are providing ~75% of the throughput of Rails 3.2.



## This may take some time





#### The 20/80

The 20% of issues that took 80% of our time

- Investigating root causes of lost performance.
- Deploying new hardware.



### Examples

#### Odd behaviours we observed when contrasting environments

Rails 4.2.5 is actually in some respects slower than Rails 3.2.



# Process Retrospective Resources

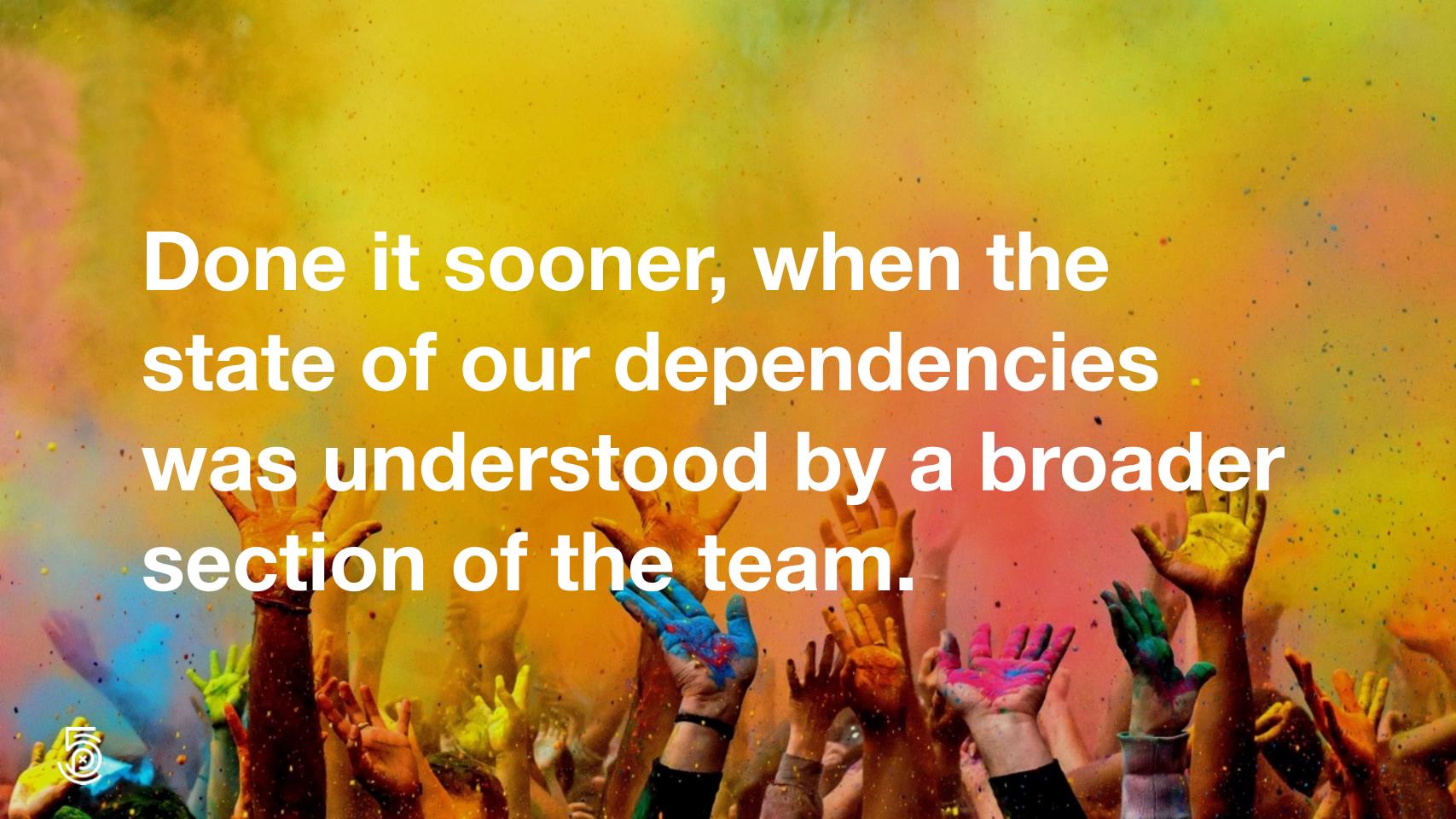


Made a way to quickly measure performance over the course of the project.



Made the data in our development environments more similar to that in our production environment.





# Process Retrospective Resources



#### Resources

#### Information we found useful

- Official Rails Upgrade Guide
- Heroku Engineering Post about Upgrading to ActiveRecord 4
- · Remarkable Labs Post about What's new in Active Record 4
- Official Mongoid 3.0 Upgrade Guide
- Artsy Post about Upgrading to Mongoid 4
- · Is Rails 4 Slower than Rails 3?







## Come talk to us about our values and open positions.

