Ruby Stories 2016

When Validations Are Not Enough

About Me

Ivan Stana

Ruby on Rails programmer at PrimeHammer

lots of interests especially in tech, but also poi spinning

github.com/istana

download the slides at blog.primehammer.com

Three Basic Questions About Invalid or Disintegrated Data in Our Database

WHY BOTHER ANYWAYS?

HOW DOES IT HAPPEN?

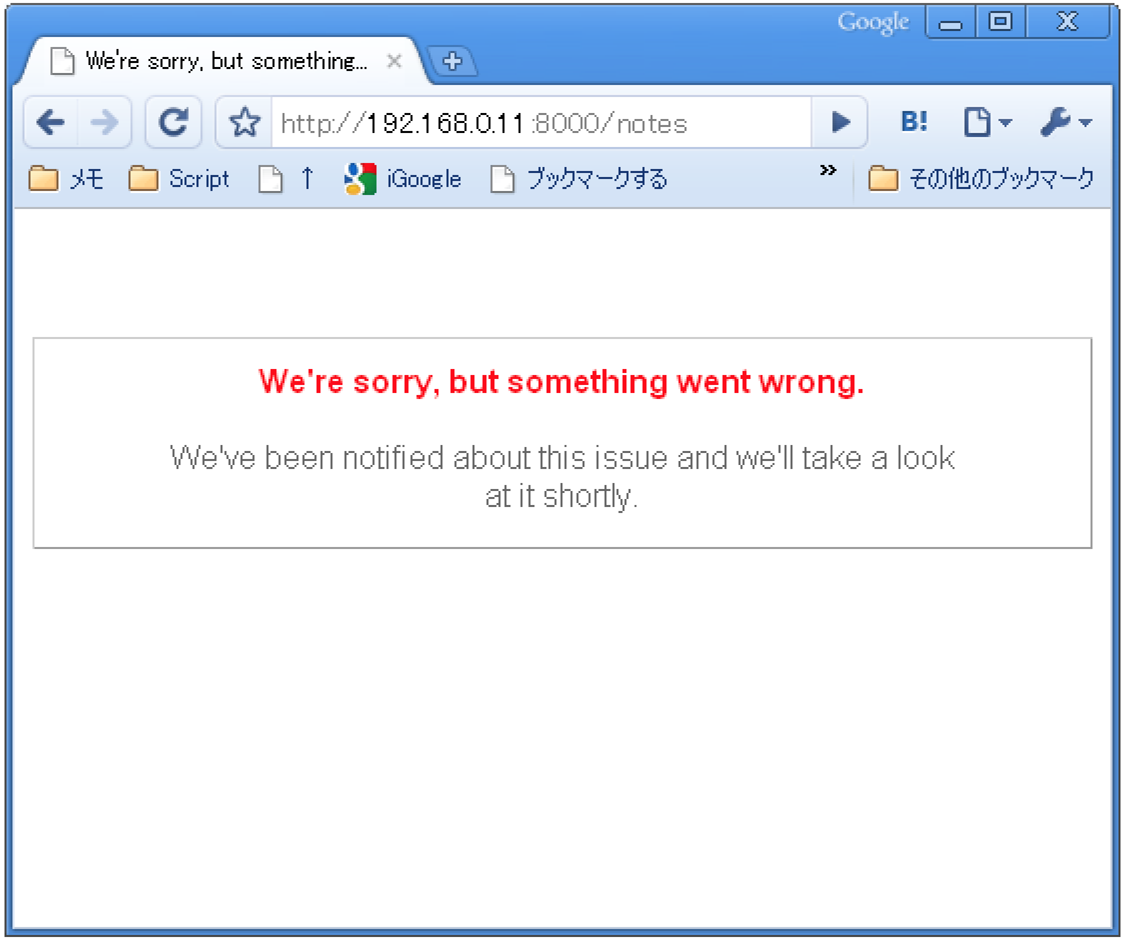
WHAT CAN WE DO ABOUT IT?

WHY BOTHER ANYWAYS?

WHY BOTHER ANYWAYS?

```
# app/views/user.html.slim
...
user.addresses.find_by(kind: 'default').zip_code
```

```
user.html.slim: undefined method 'zip_code'
 for nil:NilClass (NoMethodError)
```



We're sorry, but something went wrong.

We've been notified about this issue and we'll take a look at it shortly.

GOALS

- to have valid all instances of models (ActiveRecord/Sequel)
- security related checks, e.g. check who is a global admin against a list of known admins
- integration validations, i.e. systems outside of HTTP world
- data in DB synchronized with remote database/API
- do it daily

HOW DOES IT HAPPEN?

Sometimes we need to bypass model validation

Example: updating model that is already invalid

```
user.set_billing_info(params[:order])
user.bcrypt_password = params[:password]
user.bcrypt_password_confirmation = params[:password]
user.save(validate: false)
```

Write to a virtual attribute, but record could've been invalid before

Correct way - modify only necessary attribute(s). Example: Devise

```
user.set_billing_info(params[:order])
```

```
user.reload
user.bcrypt_password = params[:password]
user.bcrypt_password_confirmation = params[:password]
user.save(validate: false)
```

update_column

```
order.update_column(:status, :unstarted)
```

model validations and callbacks are not triggered

but it's useful for doing fast migrations or saving a state

the same for *update_all*, *delete*, *delete_all*

Database Migrations - New Columns

```
class AddAgeToUser
  def change
    add_column :age
  end
end
```

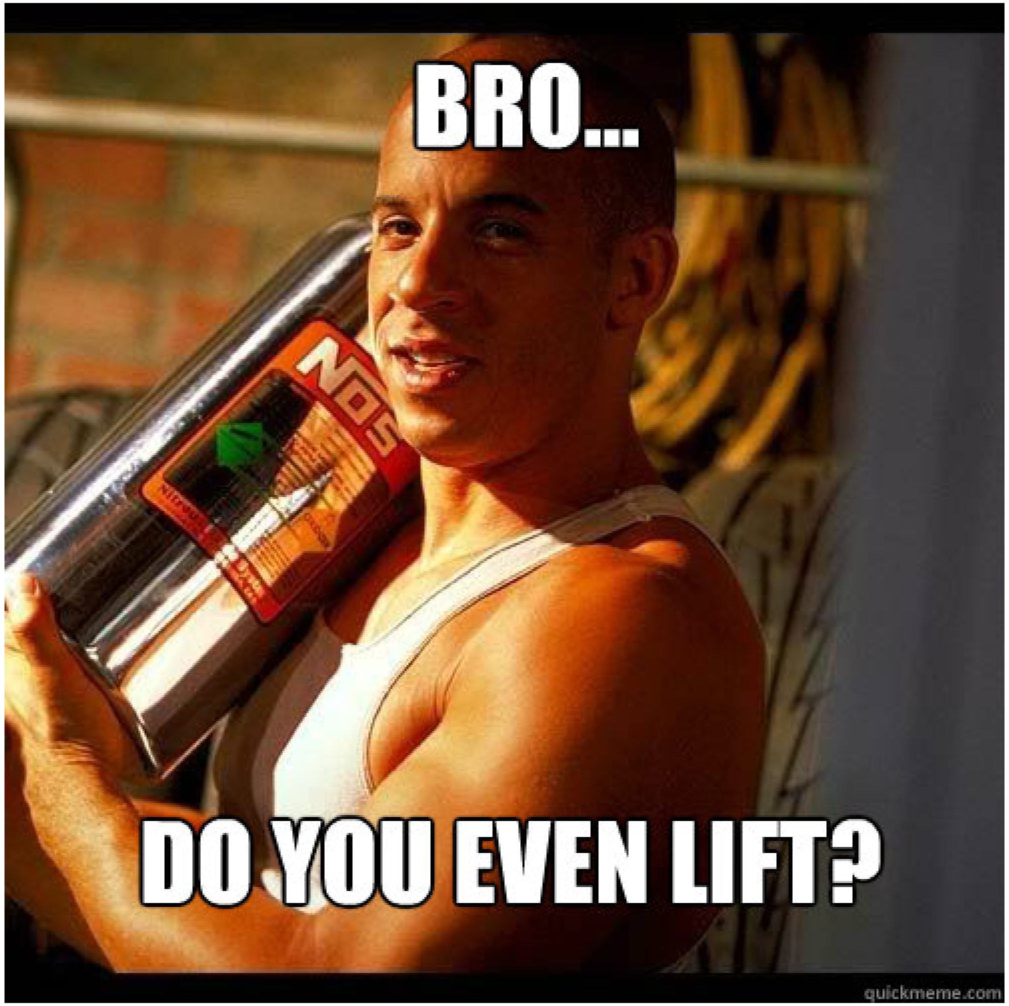
```
class User
  validates :age, presence: true
```

```
x = User.find_by first_name: 'Tremal', last_name: 'Naik'
=> #<User first_name: 'Tremal', last_name: 'Naik', age: nil>
x.valid?; x.errors
=> #["age" => ["is required"]]
```

Stricter Validations

```
class User
  validates :first_name, length: { minimum: 4 }
```

```
x = User.find_by first_name: 'Vin', last_name: 'Diesel'
=> #<User first_name: 'Vin', last_name: 'Diesel', age: 49>
x.valid?; x.errors
=> #["first_name" =>
  ["is too short (minimum is 4 characters)"]]
```


BRO...
DO YOU EVEN LIFT?
NOS
quickmeme.com

How to Limit Invalid Models

- use **null:**, **default:**, **limit:**
- https://github.com/SchemaPlus/schema_plus - a collection of gems

  https://github.com/SchemaPlus/schema_validations - auto validations

  https://github.com/SchemaPlus/schema_auto_foreign_keys - auto fk
- write a Cron job to check if all models are valid

Unsynchronized data

Unsynchronized data

they are valid in our database, but are outdated against remote database/API

- number of copyright claim of our YouTube video
- status of a document, order, package from a transport service

How it happens?

```
x = YtVideo.find_by human_id: "scorpion_king4"
x.copyright_claims
=> 0
```

Expectation

Reality

The usual problem is that YouTube sent a webhook with updated copyright claims, but we've never received it, because there was a network interruption

We can use *polling* to synchronize all data

Thank You