

Trucker

Let's talk about migrating

Suck => Less Suck

PHP => Rails

Perl => Rails

Java => Rails

Rails => Rails

Migrating = pain

- **Develop new site**
- **Migrate data**
- **Deploy new site**

Migrate data

- **Dump database**
- **Drop some tables**
- **Rename attributes**
- **Tweak data by hand**
- **Import data into new app**
- **Pray things are okay**

Sucks, right?

- **Labor intensive**
- **Error prone**
- **One way trip**



Trucker

**Migrate legacy data
(with less suck)**

- **Migrate data from day one**
- **Improve data over time**
- **Launch with confidence**

**rake +
legacy classes +
migration helper**

Let's get started

~~sude~~ gem install trucker

config.gem “trucker”

script/generate truck

\$ script/generate truck

create app/models/legacy

create app/models/legacy/legacy_base.rb

exists lib/tasks

create lib/tasks/legacy.rake

insert added legacy adapter to end of database.yml

insert added new load path to environment.rb

config/environment.rb

```
Rails::Initializer.run do |config|  
  config.load_paths += %W( #{RAILS_ROOT}/app/  
models/legacy )  
end
```

config/database.yml

legacy:

adapter: mysql

encoding: utf8

database: trucker_legacy

username: root

password:

app/models/legacy/legacy_base.rb

```
class LegacyBase < ActiveRecord::Base
  self.abstract_class = true
  establish_connection "legacy"

  def migrate
    new_record = self.class.to_s.gsub(/
Legacy/, '::').constantize.new(map)
    new_record[:id] = self.id
    new_record.save
  end
end
```

app/models/legacy/legacy_post.rb

```
class LegacyPost < LegacyBase
  set_table_name "blog_posts"

  def map
    {
      :name => self.headline.squish,
      :body => self.body.squish
    }
  end
end
```

app/models/legacy/legacy_post.rb

```
class LegacyPost < LegacyBase
  set_table_name "blog_posts"

  def map
    {
      :name => tweak(self.headline.squish),
      :body => self.body.squish
    }
  end

  def tweak(name)
    name.capitalize.gsub(/teh/, "the")
  end
end
```

lib/tasks/legacy.rake

```
namespace :db do
  namespace :migrate do

    desc 'Migrates posts'
    task :posts => :environment do
      Trucker.migrate :posts
    end

  end
end
```

Let's do some migrating.

**Don't forget to import
your legacy database!**

```
$ rake db:migrate:posts
```

```
Migrating all posts (1/10)
```

```
Migrating all posts (2/10)
```

```
Migrating all posts (3/10)
```

```
Migrating all posts (4/10)
```

```
Migrating all posts (5/10)
```

```
Migrating all posts (6/10)
```

```
Migrating all posts (7/10)
```

```
Migrating all posts (8/10)
```

```
Migrating all posts (9/10)
```

```
Migrating all posts (10/10)
```

```
$ rake db:migrate:posts limit=5
```

```
Migrating 5 posts (1/10)
```

```
Migrating 5 posts (2/10)
```

```
Migrating 5 posts (3/10)
```

```
Migrating 5 posts (4/10)
```

```
Migrating 5 posts (5/10)
```

```
$ rake db:migrate:posts limit=5 offset=5
```

```
Migrating 5 posts after 5 (6/10)
```

```
Migrating 5 posts after 5 (7/10)
```

```
Migrating 5 posts after 5 (8/10)
```

```
Migrating 5 posts after 5 (9/10)
```

```
Migrating 5 posts after 5 (10/10)
```

```
$ rake db:migrate:posts limit=5 offset=5 delete_all=true
```

```
Migrating 5 posts after 5 (6/10)
```

```
Migrating 5 posts after 5 (7/10)
```

```
Migrating 5 posts after 5 (8/10)
```

```
Migrating 5 posts after 5 (9/10)
```

```
Migrating 5 posts after 5 (10/10)
```

DEMO

**Use helper method for
custom migrations**

```
namespace :db do
  namespace :migrate do

    desc 'Migrate pain_in_the_ass model'
    task :pain_in_the_ass => :environment do
      Trucker.migrate :pain_in_the_ass, :helper => pain_in_the_ass_migration
    end

  end
end

def pain_in_the_ass_migration
  # Custom code goes here
end
```

What about?!?

Trucker helps you move.



**But you still need to
pack your stuff up.**

Resources

- [**http://github.com/mokolabs/trucker**](http://github.com/mokolabs/trucker)
- [**http://github.com/mokolabs/
trucker sample app**](http://github.com/mokolabs/trucker_sample_app)

**Special thanks
to Dave Thomas**

The End