Get a Rails 4 App with Ruby 2 Deployed to Heroku in 5 Minutes

A few days ago I started a small project, and decided that since Rails 4 was just released, I may as well get the latest and greatest, and run it on Ruby 2.
I had to compose the right incantation from a few StackOverflow questions, some old blog posts and the likes, and decided to share the right series of button presses that worked for me.

This is by no means the Best Way™, just what worked for me. If you have any notes about the steps here, feel free to drop me catch me on twitter (@avivby).

Step 1: Make sure you're using Ruby 2.0.0

For this tutorial I'm using RVM for managing the rubies on my computer. You can read up on installing and using RVM here. In case you're using a different ruby manager, I'm sure you can find equivalent commands for your tool.

First, let's check that we have Ruby 2 installed:

```
$ rvm list

rvm rubies

   ruby-1.8.7-p302 [ i686 ]
   ruby-1.8.7-p334 [ i686 ]
   ruby-1.9.2-p318 [ x86_64 ]
   ruby-1.9.3-p125 [ x86_64 ]
=> ruby-1.9.3-p392 [ x86_64 ]
   ruby-2.0.0-p0 [ x86_64 ]
```

As you can see, I happen to already have Ruby 2 installed. In case you don't see it on the list on your computer, first install it using:

```
$ rvm install 2.0.0
```

Now let's switch to using it:

```
$ rvm use 2.0.0
```

Step 2: Create an isolated gemset

I'm pedantic and don't like polluting my global Ruby installation, and so use gemsets to isolate my projects. Lets create a new gemset, r2r4 (Ruby 2 Rails 4):

```
$ rvm gemset create r2r4
$ rvm gemset use r2r4
```

Step 3: Create a new Rails app

First we'll install the Rails gem:

```
$ gem install rails
```

Now let's create a new app and initialize a new Git repository to keep track of it:

```
$ rails new r2r4
$ cd r2r4
$ git init
$ git add .
$ git commit -m 'initial commit'
```

Step 4: Create our index page

Edit *config/routes.rb* and add a route for the index page:

```
root to: 'welcome#index'
```

Create *app/controllers/welcome_controller.rb* and put in it:

```
class WelcomeController < ApplicationController
      def index
      end
end
```

And now we'll create a dead-simple view:

```
$ mkdir app/views/welcome
$ echo "Hello, World" > app/views/welcome/index.html.erb
$ git add .
$ git commit -m 'adding index page'
```

Step 5: Deploying to Heroku

First, Heroku needs some adjustments in our *Gemfile*.
At the top of it add the following line to signal that we want to use Ruby 2:

```
ruby '2.0.0'
```

Then add a production group that contains some gems that will be used to run our server properly on Heroku:

```
group :production do
      gem 'pg' # Use Postgres on Heroku
      gem 'rails_12factor' # Heroku gem that enables static assets and logging
for heroku
      gem 'unicorn' # For running the Heroku instance with unicorn
end
```

Then move the *sqlite* gem to the development group, since it's not support on Heroku:

```
group :development do
      gem 'sqlite3'
end
```

Now we will configure our app to run with unicorn on Heroku. For the specifics of these steps and the origin of the scripts, see here.

Create a *Procfile*:

```
web: bundle exec unicorn -p $PORT -c ./config/unicorn.rb
```

Create *config/unicorn.rb*:

```
worker_processes 1
timeout 30
preload_app true

before_fork do |server, worker|

  Signal.trap 'TERM' do
    puts 'Unicorn master intercepting TERM and sending myself QUIT instead'
    Process.kill 'QUIT', Process.pid
  end

  defined?(ActiveRecord::Base) and
    ActiveRecord::Base.connection.disconnect!
end

after_fork do |server, worker|

  Signal.trap 'TERM' do
    puts 'Unicorn worker intercepting TERM and doing nothing. Wait for master to
sent QUIT'
  end

  defined?(ActiveRecord::Base) and
    ActiveRecord::Base.establish_connection
end
```

Since we changed our *Gemfile*, let's do a quick install, and commit our changes:

```
$ bundle install --without production
$ git add .
$ git commit -m 'customizations for heroku'
```

Now, make sure that you have the Heroku toolbelt installed by running the following command and making sure you get no errors:

```
$ heroku
```

If it doesn't seem to be installed, follow the instructions here before continuing.
Create a new Heroku app and deploy to it:

```
$ heroku create
$ git push heroku master
$ heroku open
```

Enjoy!