Refinery CMS

Heroku

Heroku is a popular hosting choice for many developers. This guide will show you how to:

- Install and deploy a Refinery application on the Heroku hosting platform

Chapters

Edit this guide on Github

1 Creating a new Refinery application on Heroku

First you need to install Refinery. To do that you need the refinerycms gem.

```
gem install refinerycms
```

RootFile

Then, if you haven't done so already, follow the first three steps of the Heroku quick start guide. They cover signing up for Heroku, installing the Heroku client, and logging in through the client.

Now it's time to create your Refinery application using the built in `--heroku` option:

```
refinerycms myapp --heroku
```

RootFile

Heroku relies on Git being installed on your system. You should install it beforehand.

Watch the output for these lines:

```
Creating Heroku app..
Running: cd /path/to/app/myapp && heroku create
Creating random-site-name..... done
Created http://random-site-name.herokuapp.com/
```

This will output the URL for your Heroku-hosted Refinery application. Your application should now be live at http://random-site-name.heroku.com.

Note that you may have issues precompiling your assets, which may result in system images not loading. Skip to the following Step 3 for a fix.

2 Deploying an existing local Refinery application

If you have already built a Refinery application locally, you'll need to make some changes to be able to deploy to Heroku.

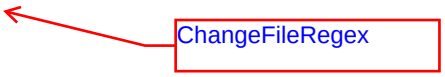
2.1 Step 1: Update the Gemfile

2.1.1 If your local database is not PostgreSQL

You don't have to change your local database settings to use PostgreSQL, but Heroku depends on the presence of the
`pg`
gem. So, in your Gemfile, change:

```
gem 'sqlite3' # or whatever the database driver for your local database is
```

to:




```
group :development, :test do
  gem 'sqlite3'
end
group :production do
  gem 'pg'
end
```

Using differing databases for development and production is not

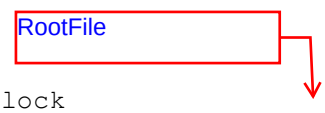
recommended. Occasionally, specific Rails idioms may have different effects on different databases. We encourage you to set up and develop on PostgreSQL if you intend to deploy your application to Heroku.

2.1.2 Getting a place to store files

If you want to use Refinery's image and resource support, you need to add the 'fog' gem too. Edit the Gemfile as shown in "Adding Amazon S3 Support" below (you can do the other steps in that section after your site is first deployed).

2.1.3 Applying your changes

Now we just need to run bundle and add the changes to git:




```
bundle install
git add Gemfile
git add Gemfile.lock
git commit -m "setup for Heroku"
```

2.2 Step 2: Set up your app on Heroku:

```
app_name="your-app-name"
heroku create $app_name --stack cedar
git push heroku master
```

2.3 Step 3: Set up asset precompilation

Inside `config/application.rb`, at the end of the config block, make sure you add the following:

```
config.assets.initialize_on_precompile = true
```

This is necessary to make post-deploy asset precompilation happen. Without this, Refinery will fail to compile its assets, owing to its use of Rails' URL helpers inside of its Javascript files (necessary for its WYSIWYG editor).

You may also need to enable the experimental `user-env-compile` option on Heroku. You can read more **here**, but in short, run the following command:

```
$ heroku labs:enable user-env-compile
```

In theory, this should only affect applications where `initialize_on_precompile` is false or default; however, you may need to set the `user_env_compile` option if you receive complaints about being unable to connect to `127.0.0.1`.

(If someone else created the Heroku app for you, make sure it is on the **Cedar stack**. Cedar is the newest stack and Heroku recommends it for new apps. You can run
`heroku stack`
to check which stack your app is on.)

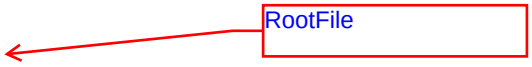
2.4 Step 4 (Option 1): Start from clean slate

If you haven't set up anything locally, or don't want to copy your local database to heroku, you'll need to run a few commands to get Refinery's database set up.

```
heroku run rake db:migrate
heroku run rake db:seed
```

This will set up the required database tables, and set up a homepage. Log in to your site to set up your first user.

```
heroku open
```



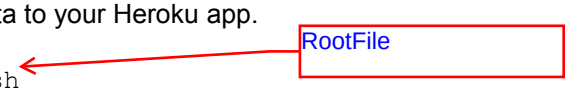
2.5 Step 4 (Option 2): Copy your data from your local database to the Heroku app

If you've developed your website locally, you likely have information in a local database that you would like to use. Rather than trying to recreate all that on Heroku, Heroku provides you with a task that requires you to install the
`taps`
gem. Be warned, though: Taps has been known to raise errors with Ruby 1.9.3. If you receive any errors or the transfer fails the first time, switch to 1.9.2 to be safe.

You'll want to actually install taps to your system – not just add it to your Gemfile.

```
gem install taps
```

Now copy the data to your Heroku app.




```
heroku pg:push
```

If that command gives you the error “no such file to load — taps/operation”, you have run into **this Heroku and taps bug**. See its comments for fixes to try.

3 Adding Amazon S3 Support

If you want to use Refinery’s image and resource support you’ll need to setup storage, too. Heroku does not persist your app’s local filesystem, so to store uploaded files, you need to store them “in the cloud”. This section explains how to store the files in Amazon S3.

On Amazon S3, create a bucket called “my_app_production”. Then add this to the end of your Gemfile (this might already be done for you):


```
group :production do
  gem 'fog'
end
```

Next, tell Heroku about your new S3 bucket.


```
heroku config:add S3_KEY=123key S3_SECRET=456secret S3_BUCKET=my_app_production
```

Make sure the config vars you add to Heroku match Refinery’s environment variables: `S3_KEY`, `S3_SECRET`, and `S3_BUCKET`

If you have created your bucket in a region other than ‘us-east-1’ you need to add S3_REGION=s3region also.

That’s it! Heroku will restart your site and it should be live with S3 support.

4 Troubleshooting

4.1 Missing a required gem

Simply add that gem to the Gemfile.

4.2 Images or Resources don’t work

Double check your S3_ information and make sure that the right buckets actually exist. You can confirm against the values Heroku has recorded by running `heroku config`.

See the **How to use Amazon S3 for storage** guide for more specific information on file storage.

4.3 Other problems?

Otherwise, run `heroku logs` or `heroku logs —tail` and see if you can spot the error yourself. Or you could explore the **help options available**.

Follow us on Twitter

- **Showcase**
- **Hosting**
- **Guides**
- **Feedback**
- **Extensions**

- **Download**
- **Community**
- **Blog**
- **About**
- **Home**