Run the app locally

Running apps locally in your own dev environment does require some effort. Rails typically requires a database. This sample application uses Postgres. You will need to follow the instructions on Dev Center for installing Postgres locally (https://devcenter.heroku.com/articles/heroku-postgresql#local-setup). When installing Postgres, make note of the password you set for the default user.

Open the `config\database.yml` file and set the username and password for your local (development) database. Look for these line:

```
development:
  <<: *default
  database: ruby-getting-started_development

  # The specified database role being used to connect to postgres.
  # To create additional roles in postgres see `$ createuser --help`.
  # When left blank, postgres will use the default role. This is
  # the same name as the operating system user that initialized the database.
  #username: ruby-getting-started

  # The password associated with the postgres role (username).
  #password:
```

Modify the username and password like this (but replace the password with the one you used upon installing Postgres):

```
  username: postgres

  # The password associated with the postgres role (username).
  password: postgres
```

Repeat this for the `ruby-getting-started_test` database, which will be in the `test:` section below the development database entry.

Now you can create the appropriate database and tables for the app using this rake task:

```
> jruby -S bin\rake db:create db:migrate
== 20140707111715 CreateWidgets: migrating ===================================
-- create_table(:widgets)
   -> 0.0076s
== 20140707111715 CreateWidgets: migrated (0.0077s) ===========================
```

The example project also contains a `Procfile.windows`, which contains the line, `web: jruby -S bin\puma -t 5:5 -p %PORT% -e development`.

This file is necessary because the command used to run the application on Windows is different from the command used to run the application on Heroku, which is Linux-based. We'll use this file later in the tutorial.

Now start your application locally using the `heroku local` command, which was installed as part of the Heroku CLI:

```
> heroku local web -f Procfile.windows
13:15:47 web.1  | started with pid 67489
13:15:47 web.1  | I, [2014-07-07T13:15:47.655153 #67489]  INFO -- : Refreshing Gem list
13:15:48 web.1  | I, [2014-07-07T13:15:48.495226 #67489]  INFO -- : listening on addr=0.0.0.0:5000 fd=10
13:15:48 web.1  | I, [2014-07-07T13:15:48.621967 #67489]  INFO -- : master process ready
13:15:48 web.1  | I, [2014-07-07T13:15:48.624523 #67491]  INFO -- : worker=0 ready
13:15:48 web.1  | I, [2014-07-07T13:15:48.626285 #67492]  INFO -- : worker=1 ready
13:15:48 web.1  | I, [2014-07-07T13:15:48.627737 #67493]  INFO -- : worker=2 ready
```

The `-f Procfile.windows` flag ensures your Windows-specific Procfile is picked up. Just like Heroku, `heroku local` examines it to determine what to run.

Open http://localhost:5000 (http://localhost:5000) with your web browser. You should see your app running locally.

To stop the app from running locally, go back to your terminal window and press `Ctrl` + `C` to exit.

Log in to report a problem (/login?back_to=%2Farticles%2Fgetting-started-with-jruby%23run-the-app-locally)

I can run my app locally (run-the-app-locally)

(Log in (/login?back_to=%2Farticles%2Fgetting-started-with-jruby%23run-the-app-locally) to save and track your progress)