Extra notes - Heroku & Git

- Dr Nick Hayward

Contents

- Basic install for Heroku
- Basic usage with Heroku
- Node and Express app setup for Heroku
- Push app to Heroku for hosting
- Rename Heroku app from cli

Basic install for Heroku Heroku is a hosted cloud service for publication of web apps &c. in a variety of languages, including Node.js.

Further details available at Heroku website,

- https://www.heroku.com

Sign up for a free account at the above site.

Then, we need to install Heroku's CLI toolbelt app,

- https://toolbelt.heroku.com

For OS X, the simplest option is to use Homebrew,

```
brew install heroku
```

and then check install version,

```
heroku --version
```

CLI help documentation can then be found using the following command,

```
heroku help
```

Reference guide for the toolbelt CLI,

- https://devcenter.heroku.com/articles/heroku-cli

Basic usage with Heroku Then, we need to login to our Heroku account on the local machine,

```
heroku login
```

This command will then request the email address registered with Heroku, and your Heroku account password.

After logging in, a local machine can now communicate with the Heroku servers.

SSH keys can also be added to Heroku CLI using the following command,

```
heroku keys:add
```

It will verify the key to upload, and then set this against the current logged in account.

n.b. Heroku will usually send a confirmation email as well - just to check you actually wanted the key added to your account…

We can now check the keys currently connected to our logged in Heroku account,

```
heroku keys
```

If we need to remove a key, we can simply use the following command,

```
heroku keys:remove
```

plus the required email address.

With the ssh key setup, we can then test the ssh connection to Heroku

```
ssh -v git@heroku.com
```

For the first run, this will check that we want to continue this remote connection, and then confirm with a debug message, `Authentication succeeded (publickey)` .

Node and Express app setup for Heroku For hosting a Node.js and Express app with Heroku, we need to modify a couple of settings in our main app file, e.g. server.js

We start by setting a variable for the server's `port` number, which is usually set to 3000, 3030 &c. for local development.

```
// store port for the app - e.g. port set by heroku for hosting OR set default for local dev...
// process.env object stores env variables as key:value pairs
const port = process.env.PORT || 3030;
```

We can then use this dynamic or default port with the app itself, either remotely with Heroku or for local dev work, e.g.

```
// specify port number for server
app.listen(port, () => {
  // output server and port - heroku will modify randomly...
  console.log(`server now listening on port ${port}`);
});
```

We can also log the server port to the console for reference.

Then, we need to modify the package.json file for the app, e.g.

```
...
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "start": "node server.js"
}
...
```

The start script is needed by Heroku to determine how to start our app. We can customise this as needed for different app.

We can then use this script to also start the app locally, e.g.

```
npm start
```

which will then start the app, and output the expected log data to the console.

Push app to Heroku for hosting After committing and pushing latest version to GitHub &c., we are then ready to push them to Heroku for the app associated on Heroku.

To add an app on Heroku for the current local app, we need to run the following local command,

```
heroku create
```

This command needs to be executed at the root of the current app, and only once per app.

The Heroku CLI toolbelt will then create a new app, assign a name, and return the url for this app on Heroku, and its associated git url on Heroku.

We can the push our app to this Heroku url for hosting.

```
git push heroku
```

This command will push the current app to the Heroku Git url just defined as part of the Heroku create process.

n.b. Heroku expects the app to reside at the root of the git repository, i.e. `heroku create` run at the root of the directory structure alongside `.git` . However, if the project you want to push to Heroku is in a sub-directory, you need to use the recent git command for `subtree` , e.g.

```
git subtree push --prefix your_app_dir_path heroku master
```

The path may be nested many levels, as long as these are specified correctly in the above command.

Also,

n.b.2 there may be sync issues with multiple collaborators to Heroku for the same project using this structure. This command can then be updated to the following,

```
git push heroku 'git subtree split --prefix your_app_dir_path branch':master --force
```

The reason this may become an issue is that the local branch may fall behind the Heroku remote.

Rename Heroku app from cli We can rename and update a Heroku app using the CLI toolbelt, again in the root directory of the enclosing git repository

```
heroku apps:rename new_name --app current_name
```

This will update the name locally, and on the Heroku servers. It will also update the git repository name for Heroku push commands.