Seneca
SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY

BTI425

Web Programming for Apps and Services

Deploy Angular app to Heroku

During development, your Angular app is “served” from an instance-based on-demand web server, to one of your browsers.

Now, we will learn how to package the Angular app so that it can be deployed to a new Heroku app, and therefore available on the public web (so that you can get to it from your computer or smartphone).

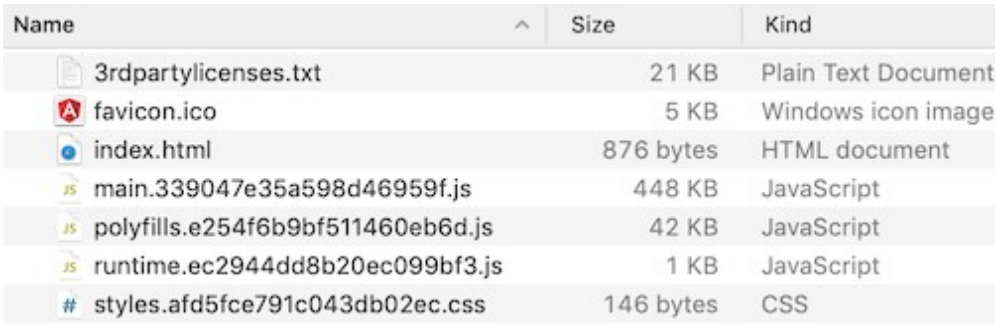
> *Good news - the process is similar to the one you learned for a React app.*

Build the app

The Angular CLI enables us to build an app. Navigate into the app/project folder. Then:

```
ng build --prod
```

It will create a folder named `dist`, and inside, another folder that matches the project name (for example, `PROJECTNAME`). In that folder, there are several static code assets in the folder. Soon, these files will be copied to a Node.js + Express.js server.



Name	Size	Kind
3rdpartylicenses.txt	21 KB	Plain Text Document
favicon.ico	5 KB	Windows icon image
index.html	876 bytes	HTML document
main.339047e35a598d46959f.js	448 KB	JavaScript
polyfills.e254f6b9bf511460eb6d.js	42 KB	JavaScript
runtime.ec2944dd8b20ec099bf3.js	1 KB	JavaScript
styles.afd5fce791c043db02ec.css	146 bytes	CSS

Node.js + Express.js server app

Create a Node.js + Express.js server app, either from new (i.e. npm init, and then npm install express), or from a template that you have. It needs a server.js file too.

It will need a git configuration. Run the `git init` command to do that.

It will need a new folder named `public`. Copy the several static code assets from the `dist/PROJECTNAME` folder into this new `public` folder.

The `server.js` code can be something like the following:

```
var express = require("express");
var app = express();
var HTTP_PORT = process.env.PORT || 8080

// Setup static content folder
app.use(express.static("public"));

// This handles a situation where the requestor sends
// a URL for a specific resource within the app
// The resource will not exist here at the server,
// because it exists only in the client device AFTER
// the Angular app is loaded
// So... handle all requests for anything other than the root
app.get('*', function (req, res) {
  console.log('Redirect was triggered');
  res.sendFile(__dirname + '/public/index.html');
});

app.listen(HTTP_PORT, () => {
    console.log("Ready to handle requests on port " + HTTP_PORT);
});
```

Test it locally. When you're happy, commit the work by using the Source Control feature in Visual

Studio Code (or do it via the command line).

Heroku hosting

For reference, here is the full guidance again. Here's a brief summary:

Before continuing:

- Confirm that `git init` was done
- Confirm that the git commit task was done

Next, login to Heroku:

```
heroku login
```

After login, we suggest that you create a new Heroku app (obviously, replace "name-of-heroku-app" with whatever name you wish):

```
heroku create name-of-heroku-app
```

Finally, deploy the app to Heroku:

```
git push heroku master
```

Update the Angular app, and redeploy

When you update the Angular app, run the build task again:

```
ng build --prod
```

Then, copy the several static code assets from the `dist/PROJECTNAME` folder into the server app's `public` folder (and overwrite/replace, obviously).

Do a git commit task again, for the server app, using Visual Studio Code, or the command line.

Finally, deploy the app to Heroku:

```
git push heroku master
```

Troubleshooting

Occasionally, it is possible that the “git push…” task will fail, and it’s likely because the project folder has lost its configuration information. Or, it’s because the new Heroku app was created with the Heroku web console, and not with the command line interface.

To fix this, run this command to update the project with the name of the new Heroku app:

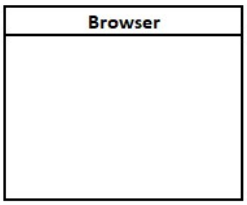
```
heroku git:remote -a name-of-heroku-app
```

Then, you can deploy to Heroku.

Diagram

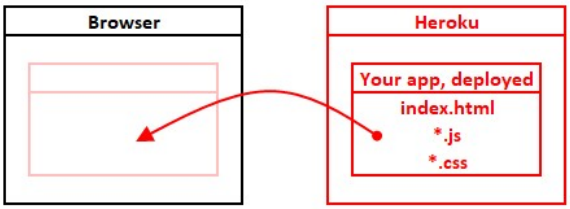
Here is a diagram of the app hosting configuration. Right-click to open it in a new tab/window and view it full size.

A browser is a content-presentation app.
It has a content-rendering engine,
and a JavaScript runtime engine/platform.



On your device (laptop, phone)

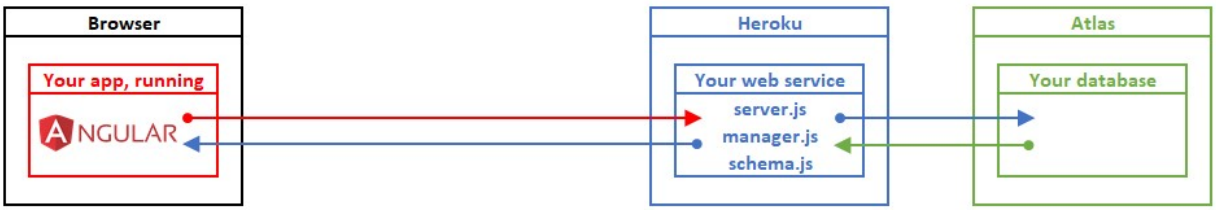
How does your browser load an app?
It fetches it from a public endpoint on the web.



On your device

For example, hosted at:
peter-a2-angularapp.herokuapp.com

Now the app is loaded and is
running in the browser.
The app's data is provided by
a remote hosted web service.



On your device

For example, hosted at:
peter-a2-webservice.herokuapp.com

Hosted at
MongoDB Atlas