Seneca
SCHOOL OF INFORMATION AND
COMMUNICATIONS TECHNOLOGY

WEB322

Web Programming Tools and Frameworks

Schedule

Notes

Graded Work

Resources

Heroku Guide

MyApps Instructions

Code examples

Getting Started with Heroku

heroku

The main server environment that we will be using in WEB322 is **Heroku**. From their website:

> *"Heroku is a cloud platform that lets companies build, deliver, monitor and scale apps — we're the fastest way to go from idea to URL, bypassing all those infrastructure headaches."*

Essentially, Heroku manages the hardware infrastructure and deployment tasks for our node.js applications in a remote server environment. Developers using Heroku can extend their applications by incorporating components from a large marketplace featuring components like database servers, scheduling components, toolkits for monitoring applications, etc. all managed remotely on their servers. Developers just need to provide the code and Heroku does the rest:

The best thing - **getting started is free!** - This is where we come in:

Creating Your Account

1. The first thing we need to do is sign up with Heroku. Navigate to their homepage and click on the large **"SIGN UP FOR FREE"** button

2. Next, fill in all of your details including your first name, last name, **Seneca email address**, country (Canada) and select **Node.js** for your "Primary Development Language". Complete the "I'm not a robot" CAPTCHA by clicking the checkbox

and passing the human test, before finally clicking the **“CREATE FREE ACCOUNT”** button.

3. This will take you to an **“Almost there…“** page prompting you to verify your email address. By now, Heroku will have sent you an email with the subject “Confirm your account on Heroku” - click on the link in the email to activate your account.

4. This will redirect you to a page to enter a password for your account. Enter **a password that you will remember** twice before clicking the **“SET PASSWORD AND LOG IN”** button.

5. This will take you to a **Welcome to Heroku** page. Click the link to proceed.

6. **Success!** - welcome to the main Heroku dashboard. This will show links to a number of “getting started” guides for various environments. Don’t worry about these for now - we will be creating our app from the command line.

Required Software

- By now, you should have **Node.js** (available here) and **Visual Studio Code** (available here). However we will also need two other pieces of software: git & the Heroku CLI

- To download git, proceed to this download page and download git for your operating system.

- Proceed to install git with the default settings. Once this is complete, you can verify that it is installed correctly by opening a command prompt / terminal and issuing the command **git --version**. This should output something like: git version 2.10.1 (…). If it does not output the installed version of git, then something is wrong and it is not installed correctly.

- Next, we need the **“Heroku CLI”** - a command line interface to our Heroku account. This can be downloaded here. Proceed to the page and download the Heroku CLI for your operating system. **NOTE**: If you’re using Visual Studio Code with MyApps, Heroku can be installed using the command: “npm install -g heroku” from the Integrated Terminal.

- Next, install the Heroku CLI with the default settings. Once this is complete, you can verify that it is installed correctly by once again opening a command prompt / terminal and issuing the command **heroku --version**. This should output something like: heroku-cli/5.6.18-9f6df83 (…) go1.7.5. If it does not output the installed version of the Heroku CLI, then something is wrong and it is not installed correctly.

Creating Your First App on Heroku!

1. Now that everything is downloaded / installed, we can create our first Node.js app on Heroku!. To begin, **create a directory** called **helloworld** on your local machine.

2. Open **Visual Studio Code** and Open your “helloworld” folder by clicking on the “Explorer” icon in the top left (it looks like two files) to expand the left “Explorer” pane. You should see an “Open Folder” button - click this and select to your newly created “helloworld” folder.

3. You should now see an empty “HELLOWORLD” folder in the sidebar. To begin building our app within this folder we have to issue a few commands in Visual Studio Code’s “integrated terminal”. To open this use the keyboard shortcut (ctrl + `) or select “View” -> “integrated terminal” from the top menu.

4. In the (now visible) integrated terminal, type the command **npm init** and enter the following information:

   - Press enter for name (use the default)
   - Press enter for version (use the default)
   - Press enter for description (use the default)
   - enter **server.js** for entry point:
   - Press enter for test command (use the default)
   - Press enter for git repository (use the default)
   - Press enter for keywords (use the default)
   - Type **your own name** for author:
   - Press enter for license: (use the default)
   - and finally, press enter to accept your entries as ok

5. This will create a package.json file under your helloworld folder. This will help us manage our dependencies later on and is always the first step creating node.js applications.

6. Next, we must retrieve the **express** module using npm. Execute the following command from within the terminal: **npm install express** - this will create a **node_modules** folder with our new express module as well as update our **package.json** file with our dependency!

7. The next step is to create your **server.js** file in your helloworld folder either by right-clicking underneath HELLOWORLD in the Explorer pane and choosing "new file", or by clicking the new file icon next to HELLOWORLD. Name this file **server.js**

8. You should now be editing your **server.js** file. Enter the following code:

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

// setup a 'route' to listen on the default url path
app.get("/", (req, res) => {
    res.send("Hello World!");
});

// setup http server to listen on HTTP_PORT
app.listen(HTTP_PORT);
```

   Be sure to save your file before moving on.

9. Next, issue the following command from the integrated terminal: **git init** - this will initialize a local git repository in your helloworld folder. You will notice that an icon in the left bar now has a blue icon showing a number. This represents all of the files that must be added to our local git repo. Click the button and type "first commit" for the message in the "Message" box. Once this is done, click the checkmark above the message box to commit your changes.

   NOTE: If, at this point, you receive the error: "Git: Failed to execute git", try executing the following commands in the integrated terminal:

```
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
```

   Once this is complete, attempt step 9 again.

10. Now that our local git repo is ready to go, we have to create an application within Heroku to send our code to. This is done by issuing the following commands in the integrated terminal:

    - **heroku login** - this command will prompt you to enter your Email and Password. Once you have done this successfully (by providing the correct email and password for your Heroku account) you will see a message "Logged in as …" where … is your email address.

    - **heroku create** - this command will create our new app within Heroku! The name that it gives our app is random (you can change it later). You will know the name it has given your app by looking at the text next to creating app… done, [*app name here*]. You will also see a url in the form **https://[*app name here*].herokuapp.com**

11. We're getting close, but not quite done yet - we need to issue one more command: **git push heroku master** - this command pushes the content of our local git repo to our new app on Heroku!

12. To verify that this is working correctly, visit that new url that you received when creating the new app (it should still be in the integrated terminal - you just have to scroll up). You should see the text **Hello World!** from our server.js file.

Congratulations! you have published your first Node.js application online. We will continue to do this throughout the semester so this process will become very familiar. If you would like to see your application in the Heroku dashboard, you can log into Heroku here: https://id.heroku.com/login. You should see your newly created app waiting for you - click on it to see information about the application including recent activity, metrics, app settings, resources, etc.

Lastly, If you would like to make modifications to your existing application, simply:

1. Make your changes
2. Click on the Source Control (Git) icon
3. Write an appropriate message describing your change
4. Click the checkmark to commit your changes

5. Push your changes to heroku using: **git push heroku master**

You should now see the changes reflected in your app!

Happy Coding!

Sources

- https://devcenter.heroku.com
- https://www.heroku.com/