Power Search

Front-End Program

Mobina

Advisor:
Aien
Hossein

Sep 2, 2021

Contents

1 Introduction

To access web pages of any web application, you need a web server. The web server will handle all the http requests for the web application. Node.js provides capabilities to create your own web server which will handle HTTP requests asynchronously.
Therefor, with the use of an old stage project (Remake of Google homepage) and Node.js runtime environment, the http server was created for this project. Also, some awesome JavaScript features was added to the stage project to make it more functional.

2 Product Documentation

2.1 Roles and Responsibilities

This project was meant to be a product of teamwork with two collaborators, but due to some unexpected events, it was decided to be a solo project assigned to 'Mobina'.
Thus, the whole process of Defining tasks and responsibilities, project management, developing the project, creating web server and documenting the progress was a one-person case.
In order to complete the project in the best possible way, 'Quire' was used as a project management tool and also Github to keep track of branches and commit histories and to prevent any conflicts throughout the project.

2.2 Team Goals and Objectives

Main goal of the project was to learn how to build a server using Node.js and add Functional features using JavaScript.

- build a http server with Node.js
- Create Google apps drop-down menu
- Create Setting menu

2.3 Assumptions

- Learning Node.js is a requirement for this project, beside HTML, CSS and JavaScript
- A Github repository to record commit history and access given to advisors
- The content is provided by the existing website
- This changes are deployed on a previously staged project

2.4 User Interaction and Design

Due to this project being a remake of an Existing website, the exact designs were implemented from Google website's homepage.

2.5 Questions

2.5.1 How to allow web server to recognize different file extensions?

a HTML content-type was given as the default extension of received files.
and then with the use of a switch statement,different existing content types were defined. [1]

2.5.2 How to style a scroll bar?

Currently, styling scrollbars for Chrome, Edge, and Safari is available with the vendor prefix pseudo-element -webkit-scrollbar.

- ::-webkit-scrollbar addresses the background of the bar itself. It is usually covered by the other elements
- ::-webkit-scrollbar-track addresses the empty space "below" the progress bar
- ::-webkit-scrollbar-thumb addresses the draggable scrolling element that resizes depending on the size of the scrollable element

(this style doesn't get rendered on Firefox browser)

2.5.3 How to use one SVG file for different icons using positioning?

Sometimes, combining multiple images into a single image file for use on a website will help with performance.
The theory is that you get the image once, and shift it around and only display parts of it using background-position to determine the exact part of the image to locate . This reduces the overhead of having to fetch multiple images and also reduces the number of server requests and saves bandwidths.[2]

2.5.4 How to make menu scroll to top each time it renders?

To make the Google apps menu opens up from the top, scrollTop property was used.
The Element.scrollTop property gets or sets the number of pixels that an element's content is scrolled vertically.It's value is a measurement of the distance from the element's top to its topmost visible content.

2.5.5 How to make a long caption fit into a small container?

One of the icon's captions in the Google apps menu, to be precise, the 'Art and culture' caption was too long to fit into its container div.
In order to fix this problem, the text-overflow property was set to 'ellipsis' and then in the hover state in was set to 'initial', also white-space property was set to normal and the value of 'break-word' was given to the word-wrap property.

2.6 Source Code

2.6.1 Creating an http web server using Node JS

An HTTP server is software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages).
Whenever a browser needs a file that is hosted on a web server, the browser requests the file via HTTP. When the request reaches the web server, the HTTP server accepts the request, finds the requested document, and sends it back to the browser, also through HTTP.
We start by loading the http, fs and path modules

```
    const http = require("http");
    const fs = require("fs");
    const path = require("path");
```

The http module contains the function to create the server.
the the "create Server" method gets called from the "HTTP" module. This method passes two arguments, the first one is the HTTP request object and the second one is the HTTP response object.

```
const server = http.createServer((req, res) => {
  let filePath = path.join(
    __dirname,
    "files",
    req.url === "/" ? "index.html" : req.url
  );

  let extname = path.extname(filePath);

  let contentType = "text/html";

  switch (extname) {
    case ".js":
      contentType = "text/javascript";
      break;
    case ".css":
      contentType = "text/css";
      break;
    case ".json":
      contentType = "application/json";
      break;
    case ".png":
      contentType = "image/png";
      break;
    case ".jpg":
      contentType = "image/jpg";
      break;
  }

  fs.readFile(filePath, (err, data) => {
    if (err) {
      if (err.code == "ENOENT") {
        fs.readFile(path.join(__dirname, "public", "404.html"), (err, data) => {
          res.writeHead(404, { "Content-Type": "text/html" });
          res.end(data, "utf8");
        });
      } else {
        res.writeHead(500);
        res.end(‘Server Error : ${err.code}‘);
      }
    } else {
      res.writeHead(200, { "Content-Type": contentType });
      res.end(data, "utf8");
    }
  });
});
```

And then it's time to add listen() method on the server on a port(3000 in this case) and this add a listener on the server.

```
server.listen(3000, () => console.log("Server running on port 3000"));
```

The Switch statement is in order to read different content-types. The call back function that catches Errors has to return :

- A 404 Error when the requested file or directory don't exist
- A 500 Error when the problem is from the server
- A 200 code when the files gets send successfully

[3]

2.6.2 Styling The scrollbar

the scrollbar with the class of 'drop-menu-g-apps' was customized using these thumbnails as pseudo classes.

```
.drop-menu-g-apps::-webkit-scrollbar {
  width: 16px;
}

.drop-menu-g-apps::-webkit-scrollbar-track {
  background: none;
  border: none;
}
.drop-menu-g-apps::-webkit-scrollbar-thumb {
  background: #dadce0;
  background-clip: padding-box;
  border: 4px solid transparent;
  border-radius: 8px;
  box-shadow: none;
  min-height: 50px;
}
```

The background-clip CSS property sets whether an element's background extends underneath its border box, padding box, or content box.
when it's set to 'padding-box' The background extends to the outside edge of the padding. No background is drawn beneath the border. [4]

2.6.3 Default size portions given to Google apps icons

During creating the Google apps menu, i realised the icons weren't quite the size the should be, even though they had the exact same width and height that was necessary.
After inspecting it was founded that it's due to this line of code which transform the images with a scale

```
transform: scale(calc(5/6));
```

2.6.4 closing the modal window by clicking outside of it

The target event property returns the element that triggered the event. in this case, we want to set the display of 'popUpMenu' to none when the event that gets clicked is other than google apps icon and the menu itself.

```
window.addEventListener("click", function (event) {
  //   console.log(event);
  if (event.target != popUpMenu && event.target != googleApps) {
    popUpMenu.style.display = "none";
});
```

2.6.5 Auto-Scroll on the menu

To make the Google apps menu opens up from the top each time it renders, this line of code was used.

```
    popUpMenu.scrollTop = "0px";
```

The scrollTop property sets or returns the number of pixels an element's content is scrolled vertically.[5]

2.6.6 fitting 'Art and culture' long caption fit into its small container

The property of 'text-overflow' was set to 'ellipsis' in the beginning.

```
  text-overflow: ellipsis;
```

Then on the hover state, it was set to its initial value.

```
.icon-arts .icon-text:hover {
  text-overflow: initial;
}
```

Also these properties were given to the element.

```
.icon-arts .icon-text:hover {
  white-space: normal;
  word-wrap: break-word;
}
```

The white-space property specifies how white-space inside an element is handled.Sequences of white-space will collapse into a single white-space. Text will wrap when necessary.
And the word-wrap property, allows long words to be able to break and wrap onto the next line.

3 Process Documentation

3.1 Plans, Estimates ,and Schedules

The first week of working on the project was assigned to two different kind of tasks: One, learning Node.js and how to build a web server. Second, working on Drop-down menus with JavaScript. And Second week was assigned to Documentation and Refactor tasks.
Even though the questions and references were being recorded during the process of project, a final touch-up was necessary.

3.2 Reports and Metrics

As i mentioned before, my teammate couldn't participate in the project due to unexpected events, so i had to manage and develop the project on my own. it was a challenge but fortunately it was handled.
The project was started off by defining tasks on 'Quire' and setting due date in a Gantt board.
the second step, was creating a Github Repository and inviting advisors as collaborators to supervise the process. and then implementing the tasks were started.

References

[1] “express.Response.contentType JavaScript and Node.js code examples — Tabnine,” 03 2019.

[2] C. Coyier, “CSS Sprites: What They Are, Why They’re Cool, and How To Use Them,” 06 2017.

[3] S. Abuse, “How To Create a Web Server in Node.js with the HTTP Module,” 07 2021.

[4] W. Le, “How To Style Scrollbars with CSS,” 08 2021.

[5] “Scroll to the top of the page using JavaScript?,” 07 2009.