1DA1611:A
Advanced Internet Programming

Project nr. 4 Report: 2023-03-16

Illia Priadko (309062)

Plan:

- Analyse given resources and choose a way of implementing responsive pages
- Prepare environment to use Bootstrap frontend toolkit
- Make the .html page, tweak and adjust script and styles
- Test and deploy the page with previous projects

Given task:

> ## Project no.4 – 2023-03-16 – Creating Responsive Websites
>
> ### Tasks
>
> 1. Create a responsive page containing the contents of the three first exercises
> 2. The page will contain a menu with
>   a. the number of exercises (left part)
>   b. Description of the page (right side)
>   c. Execution of the page after pressing a button (right side)

1. Resource analysis

After reading about the possible ways of implementing responsive pages, following one of the 1st given resource's advice, I chose to use a frontend toolkit called Bootstrap, version 5. It allows using grid-systems, collapsible elements, headers and footers, which are used in this project. These are assigned through classes. The final .html file is referring to pre-defined Bootstrap .css and .js files, that I had imported to be accessible offline, for less dependency on a third-party resource.

2. Environment preparation

For better and more reliable testing, I had decided to use a piece of software called Prepros, that allows for local hosting of the page, that is updated in real time, as well as minification of JavaScript code for more efficient resource usage.

In order to use Bootstrap v.5 resources, as well as my own custom styles and JavaScript code, the following links were made in the index.html file:

```
<!-- CSS -->
<link rel="stylesheet" href="./dist/bootstrap.min.css">
<link rel="stylesheet" href="./dist/styles-dist.css">
<!-- JavaScript -->
```

```
<script src="./dist/bootstrap.bundle.min.js"> </script>
<script src="./dist/scripts-dist.js"> </script>
```

3. Page preparation

Inside the .html file, a header and a footer were prepared, as well as the main content area. The page has to be responsive, i.e. adjust its contents to the viewport size of the target device.

Inside of the main area, referring to Bootstrap's grid system documentation, a narrower sidebar and a wider content area were prepared, that, after the viewable width is lower than a certain threshold, would relocate to show the sidebar at the top, and the main content below it. Bootstrap's grid divides area into 12 columns as a convention, so it was decided that the sidebar would take up 3 columns, and the main content would take up 9 columns on a full desktop screen. In case the viewable width would get lower than 576 pixels, both would instead stack vertically, both taking up 12 columns in width.

Here both elements are distinctly shown in red and blue:

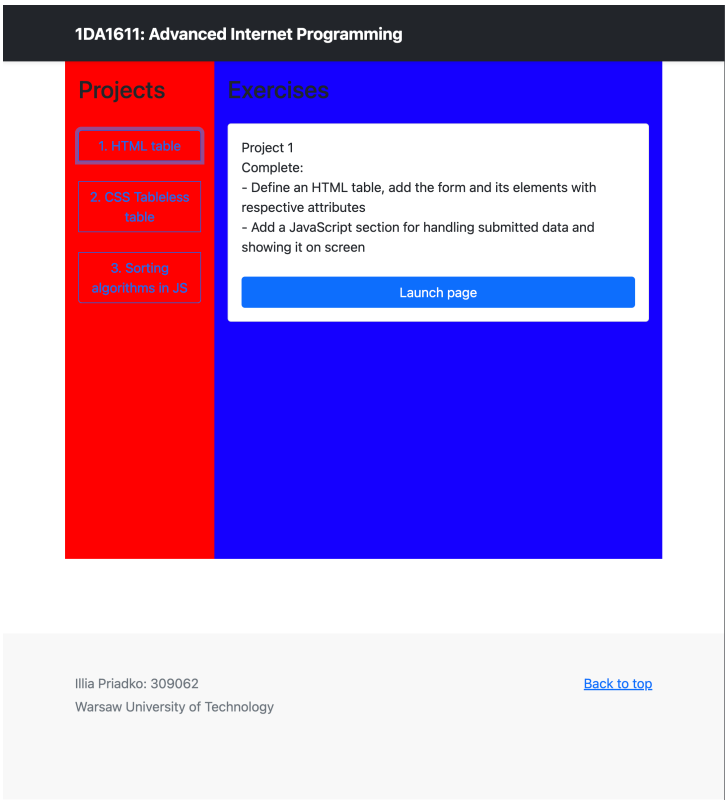

Figure 1 - Wide view

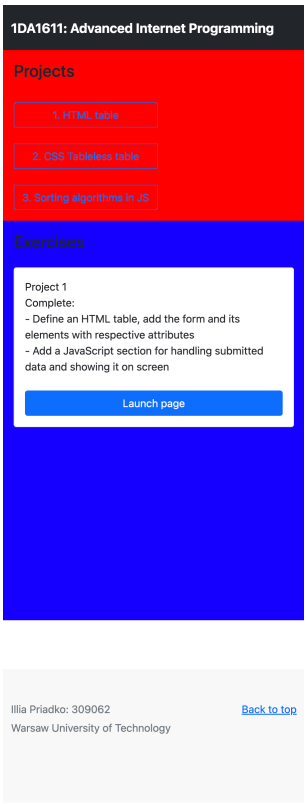

Figure 2 - Narrow view

As for the buttons, it was decided that collapsible elements would be used, connected to those buttons, to show desired content. The buttons are grouped together vertically, to fit neatly inside the sidebar.

The action of alternating content based on button press was implemented with custom JS code, where an onClick event would remove the “show” class from all elements other than the user-clicked one. During testing, it was decided to remove animations as well. The final code looks as follows:

index.html:

```
<!doctype html>
<html lang="en">

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- CSS -->
  <link rel="stylesheet" href="./dist/bootstrap.min.css">
  <link rel="stylesheet" href="./dist/styles-dist.css">
  <title>Project 4.2: Responsive page</title>
</head>

<body>
  <!-- Header -->
  <header>
    <div class="navbar navbar-dark bg-dark shadow-sm">
      <div class="container">
        <h2 class="navbar-brand d-flex align-items-center pt-3">
          <strong>1DA1611: Advanced Internet Programming</strong>
        </h2>
      </div>
```

```
    </div>
  </header>
  <!-- Main container -->
  <main>
    <div class="container my-main">
      <div class="row">
        <!-- Sidebar with project list -->
        <div class=" col-12  col-sm-12  col-md-3 sidebar-left">
          <h2>Projects</h2>
          <div class="btn-group-vertical actions">
            <button class="btn btn-outline-primary mt-4 action" id="bt1" type="button" data-bs-toggle="collapse"
              data-bs-target="#ex1" aria-expanded="false" aria-controls="ex1">
              1. HTML table
            </button>
            <button class="btn btn-outline-primary mt-4 action" id="bt2" type=" button" data-bs-toggle="collapse"
              data-bs-target="#ex2" aria-expanded="false" aria-controls="ex2">
              2. CSS Tableless table
            </button>
            <button class="btn btn-outline-primary mt-4 action" id="bt3" type=" button" data-bs-toggle="collapse"
              data-bs-target="#ex3" aria-expanded="false" aria-controls="ex3">
              3. Sorting algorithms in JS
            </button>
          </div>
        </div>
        <!-- Main content with button linking to the project -->
        <div class=" col-12  col-sm-12 col-md-9 main-content">
          <h2>Exercises</h2>

          <div class="collapse ex-content mt-4" id="ex1">
            <div class="card card-body">
              Project 1<br>Complete:<br>
              - Define an HTML table, add the form and its elements with respective attributes<br>
              - Add a JavaScript section for handling submitted data and showing it on screen
              <a class="btn btn-primary mt-4" href="./Proj1/index.html" target="_blank" role="button">Launch page</a>
            </div>

          </div>

          <div class="collapse ex-content mt-4" id="ex2">
            <div class="card card-body">
              Project 2<br>Complete:<br>
              - Link the stylesheet to the main .html file<br>
              - Change definition of the table to tableless in the .html file<br>
              - Adjust the styles
              <a class="btn btn-primary mt-4" href="./Proj2/index.html" target="_blank" role="button">Launch page</a>
            </div>

          </div>

          <div class="collapse ex-content mt-4" id="ex3">
            <div class="card card-body">
              Project 3 & 4.1<br>Complete:<br>
              -Define a set of functions in JavaScript, responsible for:<br>
              a. creating an array of 10 random numbers<br>
              b. merge-sort algorithm<br>
              c. quick-sort algorithm<br>
              - Define an HTML page as a front-end to the JS code<br>
              Additionally (4.1):<br>
              - Define time measurement inside the previous sorting algorithm script<br>
              - Adjust arrays for meaningful output
              <a class="btn btn-primary mt-4" target="_blank" href="./Proj41/index.html" role="button">Launch page</a>
            </div>
          </div>
        </div>
      </div>
    </div>

  </main>
  <!-- Footer -->
  <footer class="text-muted py-5 my-footer">
    <div class="container">
      <p class="float-end mb-1">
        <a href="#">Back to top</a>
      </p>
      <p class="mb-1"> Illia Priadko: 309062</p>
      <p class="mb-0"> Warsaw University of Technology</p>
    </div>
  </footer>
  <!-- JavaScript -->
  <script src="./dist/bootstrap.bundle.min.js"> </script>
  <script src="./dist/scripts-dist.js"> </script>
</body>

</html>
```

styles.scss (translated to styles-dist.css via Prepros):

```scss
body {
    background: white;
}

// Height of main content adjusted to viewport height

.my-main {
    height: calc(100vh - 270px);
}

.my-footer {
    height: 200px;
    background: #f8f8f8;
}

.sidebar-left {
    padding: 16px;
}

.main-content {
    padding: 16px;
    height: 600px;
}

.ex-content {
    height: 300px;
}

.actions {
    button {
        margin-top: 16px;
    }
}

// Disable animations

.collapsing {
    -webkit-transition: none;
    transition: none;
    display: none;
}
```

Scripts.js (translated to scripts-dist.js via Prepros):

```scss
const contentBoxes = document.getElementsByClassName("ex-content");

const bt1 = document.getElementById("bt1");
const bt2 = document.getElementById("bt2");
const bt3 = document.getElementById("bt3");

const toggleOtherItem = (id) => {
  for (let box of contentBoxes) {
    if (box.id !== id) {
      box.classList.remove('show');
    }
  }
}

bt1.onclick = () => {
  const id = bt1.attributes["aria-controls"].value;
  toggleOtherItem(id);
};

bt2.onclick = () => {
  const id = bt2.attributes["aria-controls"].value;
  toggleOtherItem(id);
};

bt3.onclick = () => {
  const id = bt3.attributes["aria-controls"].value;
  toggleOtherItem(id);
};
```

4. Testing and deploying

Testing was done in Microsoft Edge (Chromium-based). To test responsiveness, Devtools inside this browser allow emulating different screen sizes.

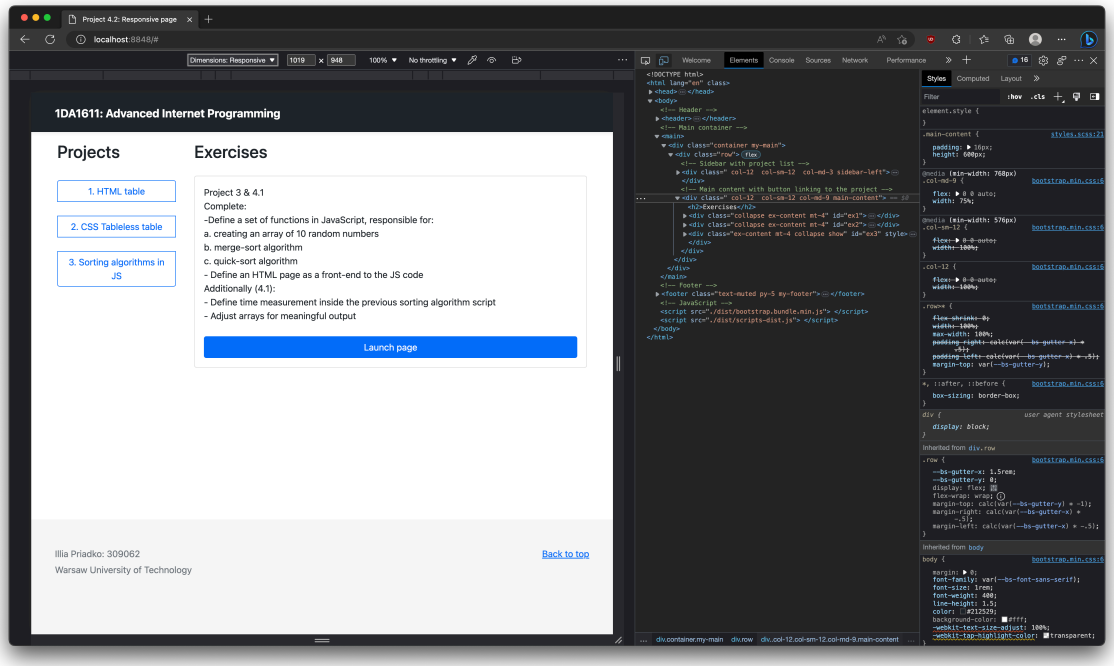

Figure 3 - Desktop view (wide)

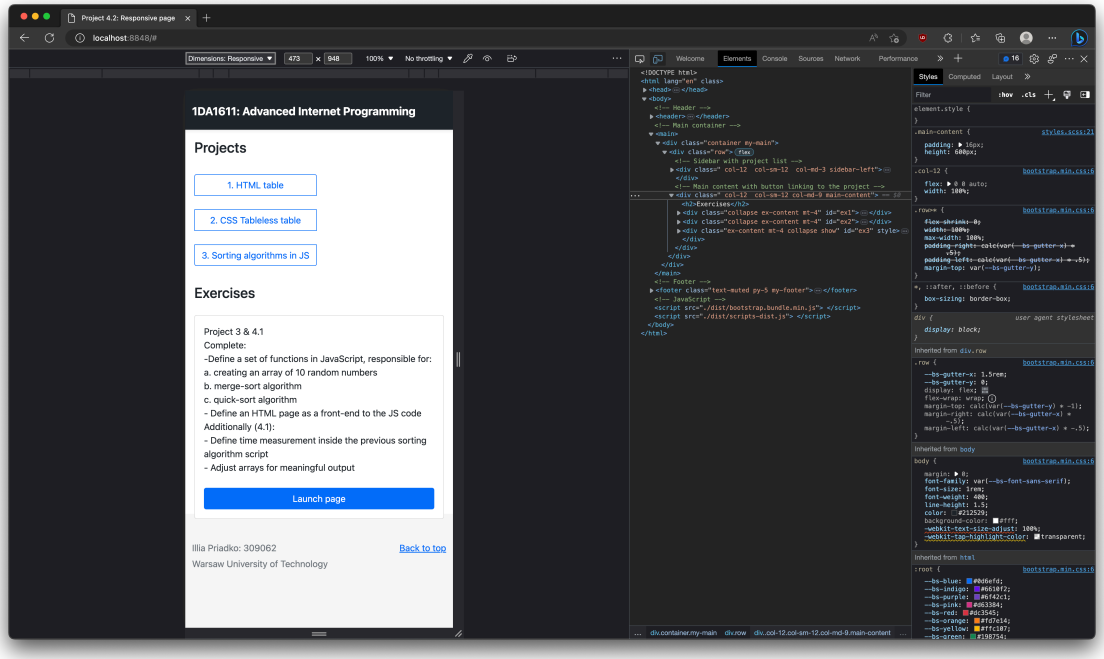

Figure 4 - Mobile view (narrow)

The page, along with the rest of projects done so far, are deployed via git, to my personal GitHub repository for easy access: https://illiapriadko.github.io/AIP_Proj_List/

Conclusion: After having completed this project, I had learned about responsive web design, the tools it is achievable with, learned classes used in Bootstrap frontend toolkit, as well as a JavaScript implementation of modifying classes and iterating, identifying elements using IDs and other attributes.