Assignment 1 – Environment Setup

Setup

Each week you will be asked to create a new folder under web-330 following a naming convention of **<week>-number**. If we are on week two, the folder name should be **week-2**. All files associated with the weekly assignment will be added to the appropriate folder. All programs must be linked in the appropriate landing page. Projects will be linked under the "Projects" section of the index.html landing page.

The document title of all HTML files in this course must say "WEB 330 – Enterprise JavaScript II." And, all HTML and CSS files must be valid HTML/CSS, tested through the WC3 validator. The links were provided during WEB 200 and 231. As part of your submission, be sure to include screenshots of the results from the validation tests (HTML and CSS validators).

User interface styling and formatting requirements are located in the Web 330 HTML, CSS, and JavaScript Requirements document.

HTML: **index.html**
CSS: **site.css and theme.css**

Grading Reminders

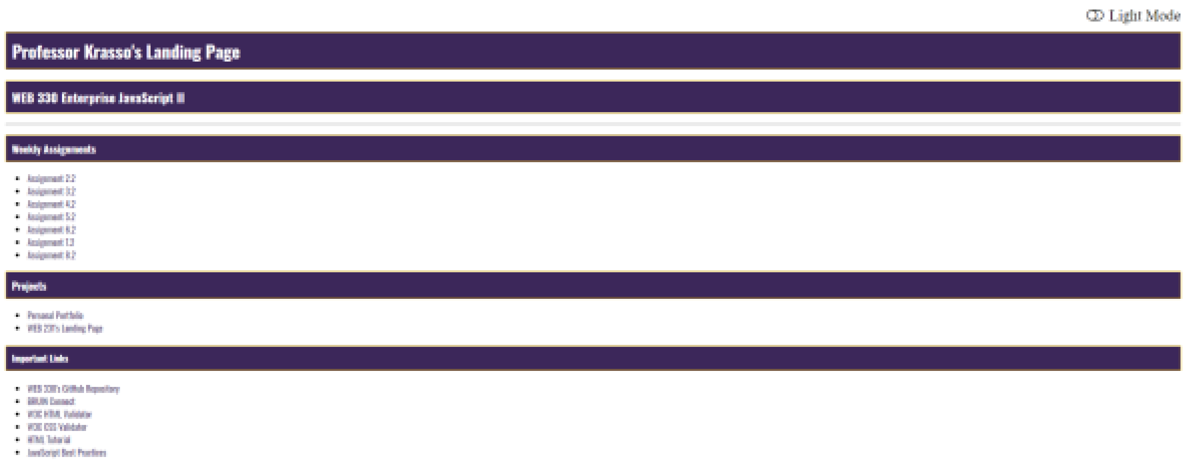
A. (rubric) All code sources (.html, .css, .js) are measured against
   1. Code functionality: Does it work? Does it meet requirements?
   2. Adherence to standards and conventions. Are you using the appropriate data types, including proper indention, are variables named appropriate (variable x is an example of poor naming conventions), is there an appropriate use of whitespace, is the code organized, and are semicolons being used to terminate code sentences?
   3. Efficiency: Use of language features. Are you practicing DRY (Don't-Repeat-Yourself?), are you leveraging built-in language features where appropriate, and are you using classes/functions to reduce code clutter?

4. Documentation: Code is maintainable by others
   i. Code comments are present in all blocks of code, written as full sentences, free of grammatical errors, and function/class parameters and data types have been identified.
   ii. Code attribution is present in all files and authorship is clearly annotated.
5. Error trapping/handling. Are there errors in the program? Is there evidence of coding best practices to reduce user errors?
6. Assignment Specific Compliance. Does the delivered solution follow the instructions, as they are written? Does the output match what was provided in the screenshots (including spaces, styling, etc.)?

Required Modifications

- Cite any sources in your opening programmer's comment

Exhibit A. User Interface (final solution)



1. Add a new file under web-330 and name it index.html

HTML Requirements

a. Give the HTML document a title of "WEB 330 – Enterprise JavaScript II"
b. Add a link to theme.css and site.css files in the <header> section.

c. Add a link to the CDN for font-awesome. If you run into issues, visit the courses GitHub repository under week-0 and look for a file named theme.html. This file includes a reference to the font-awesome CDN.
d. Add the courses Google font kit

Google Font Kit Requirements

All assignments in this course will be using the Google font kit: Oswald, Verdana, Arial, sans-serif. The process for creating Google fonts was covered in WEB 200. This is the only approved font type for WEB 330 – Enterprise JavaScript II.

e. Give the HTML body the CSS class "light-theme."
f. Add a main div to the body of the HTML page and give it a CSS id of **container**.

Additional HTML Requirements

1) Add an HTML icon tag with an onclick even that calls the function toggleMode and pass-in the reference this. Give the element and id of icon-mode, CSS classes "fa fa-toggle-off pull-right," and inline CSS styling with a font size of 28 pixels. Inside the icon element added an HTML span tag with an id of icon-text. If you run into issues, visit the courses GitHub repository under week-0 and look for a file named theme.html. This file includes the code you will need to create this icon.
2) Add an h1 tag and give it a value of "<yourFirstName> <yourLastName>'s Landing Page"
3) Add an h2 tag and give it a value of "WEB 330 Enterprise JavaScript II"
4) Add an HTML horizontal line.
5) Add an h3 tag and give it a text value of "Weekly Assignments."
6) Underneath the h3 tag add an unordered list with list items for the assignments shown in Exhibit A.
7) Underneath the assignments section add another h3 tag and give it a text value of "Projects."
8) Underneath the projects section add another unordered list with list items for the projects shown in Exhibit A.
9) Underneath the projects section add an h4 tag with a text value of "Important Links."

10) Underneath the important links section add links for
    i. WEB 330's GitHub Repository: https://github.com/buwebdev/web-330
    ii. BRUIN Connect: https://bruinconnect.bellevue.edu/
    iii. W3C HTML Validator: https://validator.w3.org/
    iv. W3C CSS Validator: http://jigsaw.w3.org/css-validator/
    v. HTML Tutorial: https://www.w3schools.com/html/
    vi. JavaScript Best Practices: https://www.w3schools.com/js/js_best_practices.asp

2. Add a new file under web-330 and name it site.css

Additional Styling Requirements

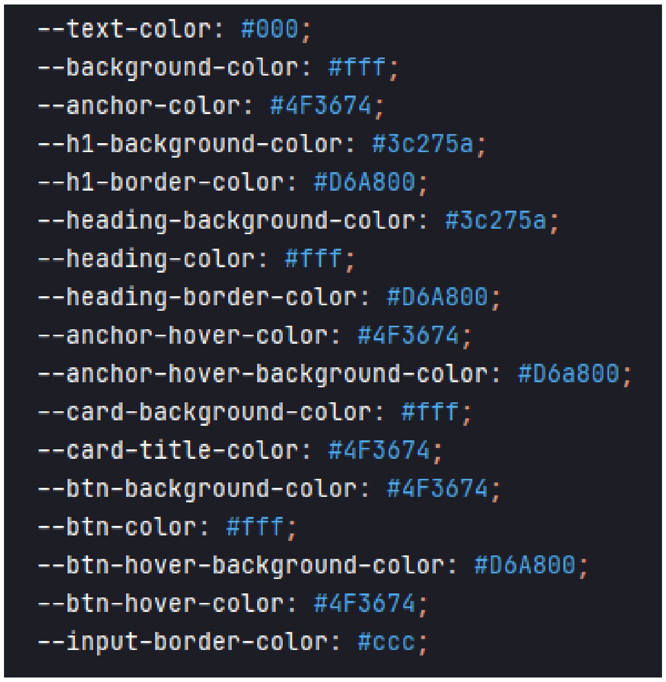
a. Add a CSS entry for #container and set the margin top to 50 pixels, the left margin to 100 pixels, and the right margin to 100 pixels.
b. Add a CSS entry for #icon-mode:hover and set the cursor to a pointer.
c. Add a CSS entry for #icon-text and set the left padding to 10 pixels.
d. Add a CSS class named pull-right and give it a style of float right.
e. Verify the CSS classes: assign-container, assign-content, assign-results-text, app-header, return-home, return-home:hover, return-home:visited, form, form-field, and full-width were added to this file (this was covered in the document titled Web 330 HTML, CSS, and JavaScript Requirements).

3. Add a new file under web-330 and name it theme.css

Additional Styling Requirements

a. Add a CSS class for light-theme and assign it to the HTML body element
    1) Code snippet: **body.light-theme**
    2) See Exhibit B for a list of the required variables

Exhibit B. Light-theme variables

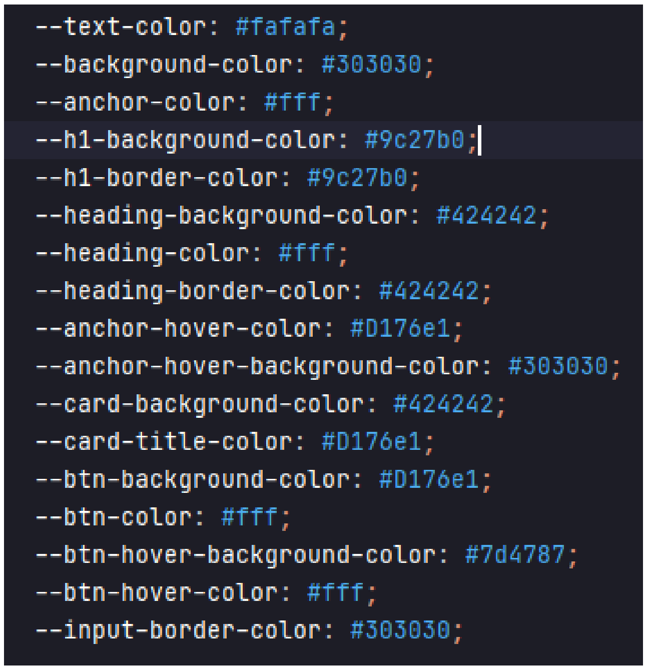


```
--text-color: #000;
--background-color: #fff;
--anchor-color: #4F3674;
--h1-background-color: #3c275a;
--h1-border-color: #D6A800;
--heading-background-color: #3c275a;
--heading-color: #fff;
--heading-border-color: #D6A800;
--anchor-hover-color: #4F3674;
--anchor-hover-background-color: #D6a800;
--card-background-color: #fff;
--card-title-color: #4F3674;
--btn-background-color: #4F3674;
--btn-color: #fff;
--btn-hover-background-color: #D6A800;
--btn-hover-color: #4F3674;
--input-border-color: #ccc;
```

b. Add a CSS class for dark-theme and assign it to the HTML body element

1) See Exhibit C for a list of the required variables

Exhibit C. Dark-theme variables



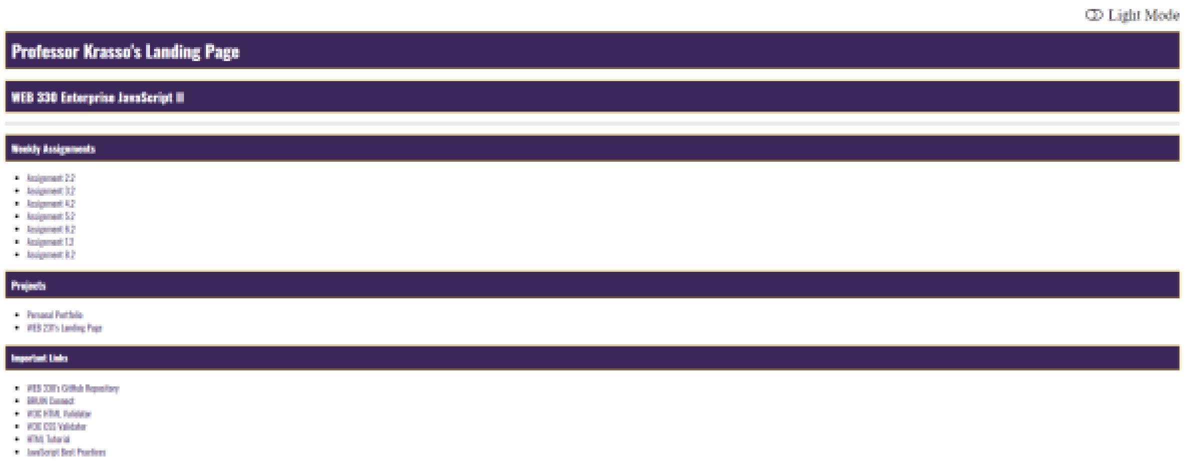
```
--text-color: #fafafa;
--background-color: #303030;
--anchor-color: #fff;
--h1-background-color: #9c27b0;
--h1-border-color: #9c27b0;
--heading-background-color: #424242;
--heading-color: #fff;
--heading-border-color: #424242;
--anchor-hover-color: #D176e1;
--anchor-hover-background-color: #303030;
--card-background-color: #424242;
--card-title-color: #D176e1;
--btn-background-color: #D176e1;
--btn-color: #fff;
--btn-hover-background-color: #7d4787;
--btn-hover-color: #fff;
--input-border-color: #303030;
```

c. Set the body to the course's font family (Oswald, Verdana, Arial, sans-serif).

d. Set the HTML body color and background to the --text-color and --background-color variables. If you are not sure what this means, visit the courses GitHub repository under week-0 and look for the files theme.css and theme.html.
e. Set the HTML anchor color to the variable --anchor-color and remove the underline.
f. Set the HTML anchor hover background color to the variable --anchor-hover-background-color and the text color to the variable --anchor-hover-color.
g. Set the HTML h1 heading color to the variable --heading-color, the background color to the variable --h1-background-color, give it a padding of 10 pixels, and a boarder of a solid one pixel and assign the variable --h1-border-color to the border color.
h. Set the HTML h2, h3, and h4 color to the variable --heading-color, the background color to the variable --heading-background-color, give it padding of 10 pixels, and a border of a solid one pixel and assign the variable --heading-border-color to the border color.
i. Replace the CSS card classes background color with the variable --card-background-color.
j. Replace the CSS card-title classes color with the variable --card-title-color.
k. Replace the CSS btn-primary classes background color with the variable --btn-background-color and the text color with the variable --btn-color.
l. Replace the CSS btn-primary:hover background color with the variable --btn-hover-background-color and the text color with the variable --btn-hover-color.
m. Replace the CSS input classes border color with the variable --input-border-color.
n. Replace the CSS drop-down-menu classes border color with the variable --input-border-color.

Exhibit D. Light mode

When a user selects “Light Mode” the icon will change, the text will be replaced with “Dark Mode,” and the HTML elements will be updated with the appropriate classes.



Professor Krasso's Landing Page

WEB 330 Enterprise JavaScript II

Weekly Assignments

- Assignment 2.2
- Assignment 3.2
- Assignment 4.2
- Assignment 5.2
- Assignment 6.2
- Assignment 7.2
- Assignment 8.2

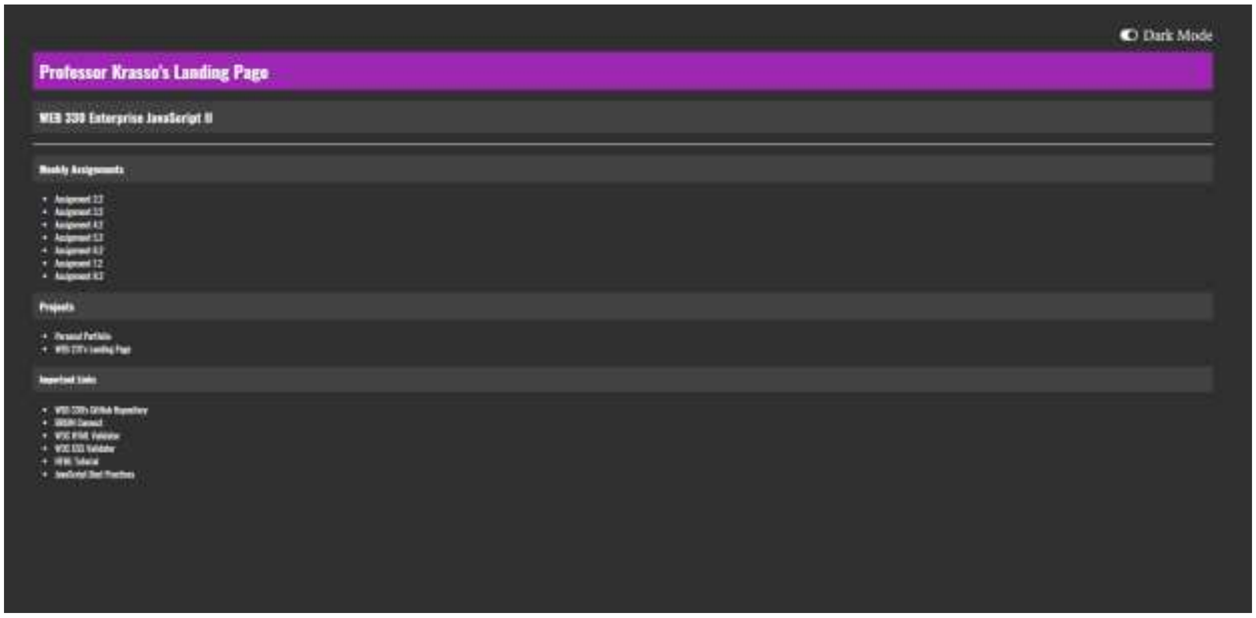
Projects

- Personal Portfolio
- WEB 231's Landing Page

Important Links

- WEB 330's GitHub Repository
- BRUIN Connect
- W3C HTML Validator
- W3C CSS Validator
- HTML Tutorial
- JavaScript Best Practices

Exhibit E. Dark mode



Professor Krasso's Landing Page

WEB 330 Enterprise JavaScript II

Weekly Assignments

- Assignment 2.2
- Assignment 3.2
- Assignment 4.2
- Assignment 5.2
- Assignment 6.2
- Assignment 7.2
- Assignment 8.2

Projects

- Personal Portfolio
- WEB 231's Landing Page

Important Links

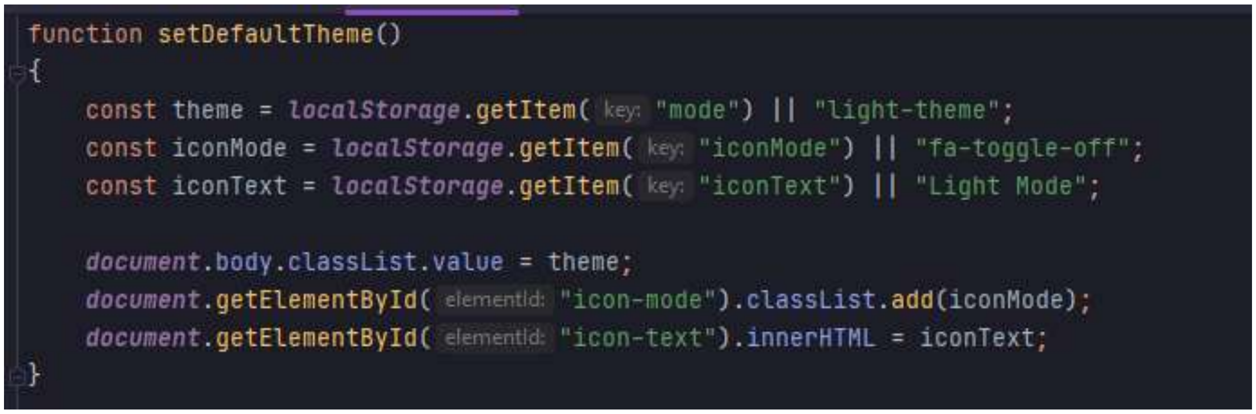
- WEB 330's GitHub Repository
- BRUIN Connect
- W3C HTML Validator
- W3C CSS Validator
- HTML Tutorial
- JavaScript Best Practices

4. Visit the courses GitHub repository under week-0 and look for a file named theme.html. Review the files JavaScript code and add it to the index.html file (your landing page). As a disclaimer, do not just copy/paste the code into your index.html file. Take the time to read through the code and if you have questions, please post them to the course's discussion board forum. At a high-level, when a user selects "Dark Mode" their preferences are added to the

browsers localStorage and the variables we created in the theme.css are applied to the HTML elements. When a user selects "Light Mode" their preferences are updated in the browsers localStorage and the variables we created in the theme.css file are applied to the HTML elements. To propagate the styling in the weekly assignments, you will need to grab the user's preference from localStorage and apply the CSS class "light-theme" or "dark-theme" to the HTML body element. Refer to the "Web 330 HTML, CSS, and JavaScript Requirements" document for more information.

5. Visit the courses GitHub repository under week-0 and look for a file named theme.js. Review the files JavaScript code for setDefaultTheme(). Create a new file in your projects root directory named theme.js and add it to the top of the newly created theme.js file. As a disclaimer, do not just copy/paste the code into your theme.js file. Take the time to read through the code and if you have questions, please post them to the course's discussion board forum. To use this file, you will need to add it to the HTML documents <head> section. The index.html page (your landing page) will need to call the setDefaultTheme() function before all other JavaScript code. For the setSelectedTheme() function, refer to the WEB 330 HTML, CSS, and JavaScript requirements document. By the end of this week you should have two functions in this file setDefaultTheme() and setSelectedTheme().

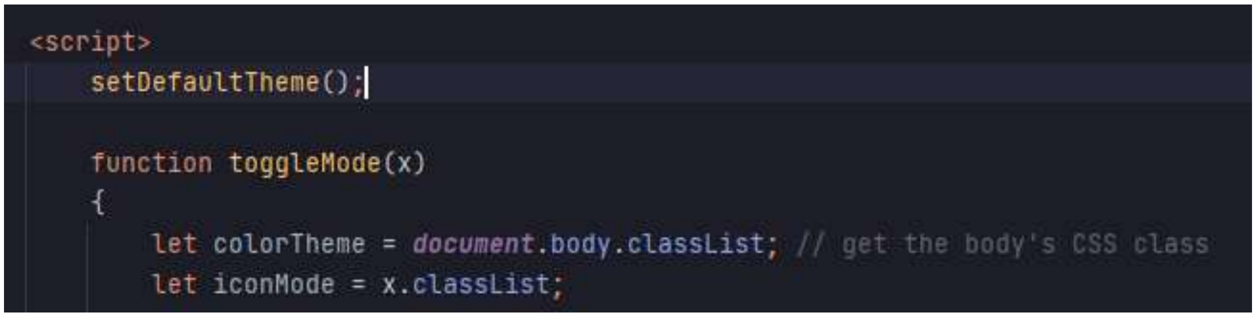
Exhibit F. setDefaultTheme() function



```
function setDefaultTheme()
{
    const theme = localStorage.getItem( key: "mode") || "light-theme";
    const iconMode = localStorage.getItem( key: "iconMode") || "fa-toggle-off";
    const iconText = localStorage.getItem( key: "iconText") || "Light Mode";

    document.body.classList.value = theme;
    document.getElementById( elementId: "icon-mode").classList.add(iconMode);
    document.getElementById( elementId: "icon-text").innerHTML = iconText;
}
```

Exhibit G. index.html setDefaultTheme() function call



```
<script>
    setDefaultTheme();

    function toggleMode(x)
    {
        let colorTheme = document.body.classList; // get the body's CSS class
        let iconMode = x.classList;
```

6. Validate the HTML and CSS by using W3C's HTML and CSS validators. There should not be any errors in the results printout. If there are errors, correct them and rerun the validation tests.
7. Combine all images in a single Word document include your name, date, and assignment number. At minimum, you will need screenshots of the validation tests (CSS and HTML) and a screenshot of the landing page running on GitHub pages.
8. Configure the repository for GitHub pages. A document was provided during WEB 231 with instructions on how to configure a static website for GitHub pages.