Assignment: Content Manipulation

Task description

You are going to create a mobile first responsive web-page about making green energy affordable for people with average budget. The web-page will have nav-bar, masthead, main-content and footer. Nav-bar has 4 items: home, about, contact and blog. Masthead has to have a background image and a tag-line.

For main-content, you will have 3 buttons and 3 contents (your 3 solutions about making a clean energy affordable for majority of people) stored in an object. On the page load, the first content will be hard-coded HTML. This content will be replaced by another content from JavaScript object once a button is clicked. The first button (because it corresponds to the first content displayed on the page-load) will have a different style than other two (indicating active button). As user clicks any button, id "active-button" will move to the clicked button and corresponding content will be displayed. Size of images and fonts change with the screen-width. Break-point for responsive design will be at: 30rem. Max-width of content is 60rem. This means that content has to be centered in the view-port once 60rem of content-width is exceeded.

Footer needs to have links to at least 3 social media networks (use icon-font).

Order of Operations

1. Create directories and files
2. Create wire-frame and mockup
3. Complete HTML
4. Complete CSS
5. Complete JavaScript

Directories - Structure

Directory tree (including names) has to look like this:

/green-energy/

/css/

/main.css

/img/

/wireframe-mockup/

/js/

/script.js

/index.html/

Fonts

You need to use serif/sans-serif Google font pair. Serif font is for headings and sans-serif font is for body text. Fall-back font is going to be default serif/sans-serif font.

References:

* [http://fontpair.co](http://fontpair.co/)
* [https://fonts.google.com](https://fonts.google.com/)

Line height ratio has to be between 1.4 and 1.6. You need to use following font-sizes:

* Mobile View: 0.75rem
* Tablet/Desktop View: 1rem

Colors

Color scheme needs to be analogous. List the hexadecimal values of colors (as CSS comment) inside CSS style-sheet .

HTML elements with background color will expand across the screen while content is centered in the middle of view-port and has limited width.

Reference: [color.adobe.com](https://color.adobe.com/create/color-wheel/?base=2&rule=Analogous&selected=2&name=My%20Color%20Theme&mode=rgb&rgbvalues=0.48828182115766083,0.013612449956360129,1,0.19953300165134202,0.012387329460287717,0.91,0.036387550043639916,0.09611296892434906,1,0.012387329460287717,0.3042358934196418,0.91,0.013612449956360129,0.5939008286677046,1&swatchOrder=0,1,2,3,4)

Images

You could use your own images or choose images from http://www.freeimages.com/ or http://www.morguefile.com/.

All images have to be OPTIMIZED for web.

Set the background image for masthead.

The screen-shots below are not to be replicated in design nor content - they only represent the guidelines for required layout and alignment. The topic is already given in the task description. You will choose your color-scheme, your images and your font-family (font-stack), BUT the content has to be responsive in the way that screen-shots indicate (including order of elements, alignments, horizontal and vertical gutters of white space).

Mobile View < 30rem

The gutter of white space around the content is 2rem. Everything aligned left.

30rem < Tablet/Desktop View < 60rem

Nav-bar items are floated and horizontally aligned. Main-content image-width is 25%. The image is floated - text wraps around.

Coding Guidelines

// Load your images on page-load  
 function preloader() {  
 const imagesList = [  
 "./img/img-1.jpg",  
 "./img/img-2.jpg",  
 "./img/img-3.jpg"  
 ];  
 const images = [];  
 for (let i = 0; i < imagesList.length; i++) {  
 images[i] = new Image();  
 images[i].src = imagesList[i];  
 }  
  
 // Images ready to be used:  
 console.log(`Preloaded images:\n\t${images[0].src}\n\t${images[1].src}\n\t${images[2].src}`);  
 };   
 window.addEventListener("load", preloader);  
   
   
 /\*   
 Get all buttons in a NODE LIST of buttons (array like structure) \*/  
   
 /\*   
 Complete your resource-object that will store the dynamic content.  
 Resource object should 3 sub-objects. Give your sub-objects  
 meaningful names. Every sub-object should have the following  
 properties headingContent, bodyText, imgUrl and imgAlt. \*/  
  
 /\*   
 Get the reference to your HTML-container  
 that will be dynamically loaded from the resource-object. \*/  
   
 /\*   
 The first button in a NODE LIST of buttons will initially   
 have the id: active-button - this will uniquely style   
 the active button (CSS rule).   
   
 The first content from the  
 resource-object will be loaded on the page load:  
 `<h1>${headingContent}</h1>  
 <img src="${imgUrl}" alt="${imgAlt}">  
 <p>${bodyText}</p>` \*/  
   
 /\*   
 Start your handleSelection function here. \*/   
   
 /\*   
 Remove the id active-button from the element that  
 contains it prior to the click-event.   
  
 This will require the loop throught the NODE LIST of buttons.   
 Inside the loop, use conditional and the element object method  
 hasAttribute() to check if the current button in the loop containes the id.  
 If it does, use element-object property removeAttribute()  
 to remove the id. \*/  
  
 /\*  
 Use the element-object method setAttribute() to set the id active-button   
 to the currently clicked button. \*/  
   
 /\*   
 Use conditional and event-object to check which button is clicked  
 and based on that, create HTML with the data inside the backticks:  
 `<h1>${headingContent}</h1>  
 <img src="${imgUrl}" alt="${imgAlt}">  
 <p>${bodyText}</p>`  
 Assign this content to to your HTML-container that will   
 be dynamically loaded (you already got the reference to   
 this container before you started the function handleSelection) \*/   
   
 /\*   
 Close your handleSelection function here. \*/   
   
 /\*   
 Register all buttons to click event. The event-handler handleSelection will listen   
 for this event to happen. \*/

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
The good CSS

/\* Analogous Color Scheme for Green Energy \*/

/\* Color 1: #66BB6A (Green) \*/

/\* Color 2: #43A047 (Dark Green) \*/

/\* Color 3: #1B5E20 (Darker Green) \*/

body {

display: flex;

flex-direction: column;

min-height: 100vh;

font-family: 'Roboto', sans-serif;

line-height: 1.6;

margin: 0;

padding: 0;

box-sizing: border-box;

}

main {

flex: 1;

}

footer {

text-align: center;

padding: 1rem;

background-color: #f8f8f8;

margin-top: 2rem; /\* Optional: Add margin-top to create more space \*/

}

/\* Nav-bar \*/

nav ul {

display: flex;

justify-content: center;

list-style-type: none;

background-color: #f8f8f8;

padding: 1rem;

}

nav ul li {

margin: 0 1rem;

}

nav ul li a {

color: #000;

text-decoration: none;

padding: 0.5rem 1rem;

text-transform: capitalize;

border: 1px solid #ccc;

border-radius: 5px;

}

nav ul li a:hover {

background-color: #66BB6A;

color: #fff;

}

/\* Masthead \*/

.masthead {

position: relative;

height: 300px; /\* Adjust this height as needed \*/

background-color: #66BB6A;

overflow: hidden;

}

.masthead img {

width: 100%;

height: 100%;

object-fit: cover;

}

.masthead h1 {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

color: green;

font-family: 'Merriweather', serif;

font-size: 2rem;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

text-align: center;

}

/\* Main Content \*/

.buttons {

display: flex;

justify-content: center;

margin: 2rem 0;

}

.buttons button {

padding: 1rem;

margin: 0 0.5rem;

background-color: #43A047;

color: #fff;

border: none;

cursor: pointer;

text-transform: capitalize;

}

#active-button {

background-color: #1B5E20;

}

#content {

text-align: center;

padding: 2rem;

max-width: 60rem;

margin: 0 auto;

margin-bottom: 2rem;

}

#content img {

width: 200px;

height: 200px;

max-width: 300px;

object-fit: cover;

display: block;

margin: 0 auto 1rem;

}

/\* Footer \*/

footer {

text-align: center;

padding: 1rem;

background-color: #f8f8f8;

}

footer a {

color: #1B5E20;

text-decoration: none;

margin: 0 1rem;

}

footer a:hover {

color: #66BB6A;

}

@media (min-width: 30rem) {

body {

font-size: 1rem;

}

.masthead h1 {

font-size: 2rem;

}

#content img {

float: left;

margin-right: 1rem;

}

}

/\* @media (max-width: 30rem) {

body {

font-size: 0.75rem;

}

.masthead h1 {

font-size: 1.5rem;

}

#content img {

float: none;

margin: 0 auto 1rem auto;

}

} \*/

/\* Adjustments for Mobile Screens \*/

@media (max-width: 30rem) {

body {

text-align: left; /\* Align everything left \*/

}

nav ul {

flex-direction: column;

padding: 0;

border-bottom: 1px solid #ccc;

text-align: left; /\* Align navigation links left \*/

}

nav ul li {

margin: 0;

}

nav ul li a {

display: block;

padding: 1rem;

border: none;

border-bottom: 1px solid #ccc;

}

.buttons {

display: flex;

justify-content: center; /\* Center buttons horizontally \*/

margin-bottom: 1rem; /\* Add margin bottom for spacing \*/

}

.buttons button {

margin: 0 0.5rem; /\* Adjust button margin \*/

}

#content {

padding: 1rem;

}

#content img {

width: 100%;

height: auto;

max-width: 100%;

margin-bottom: 1rem;

}

}