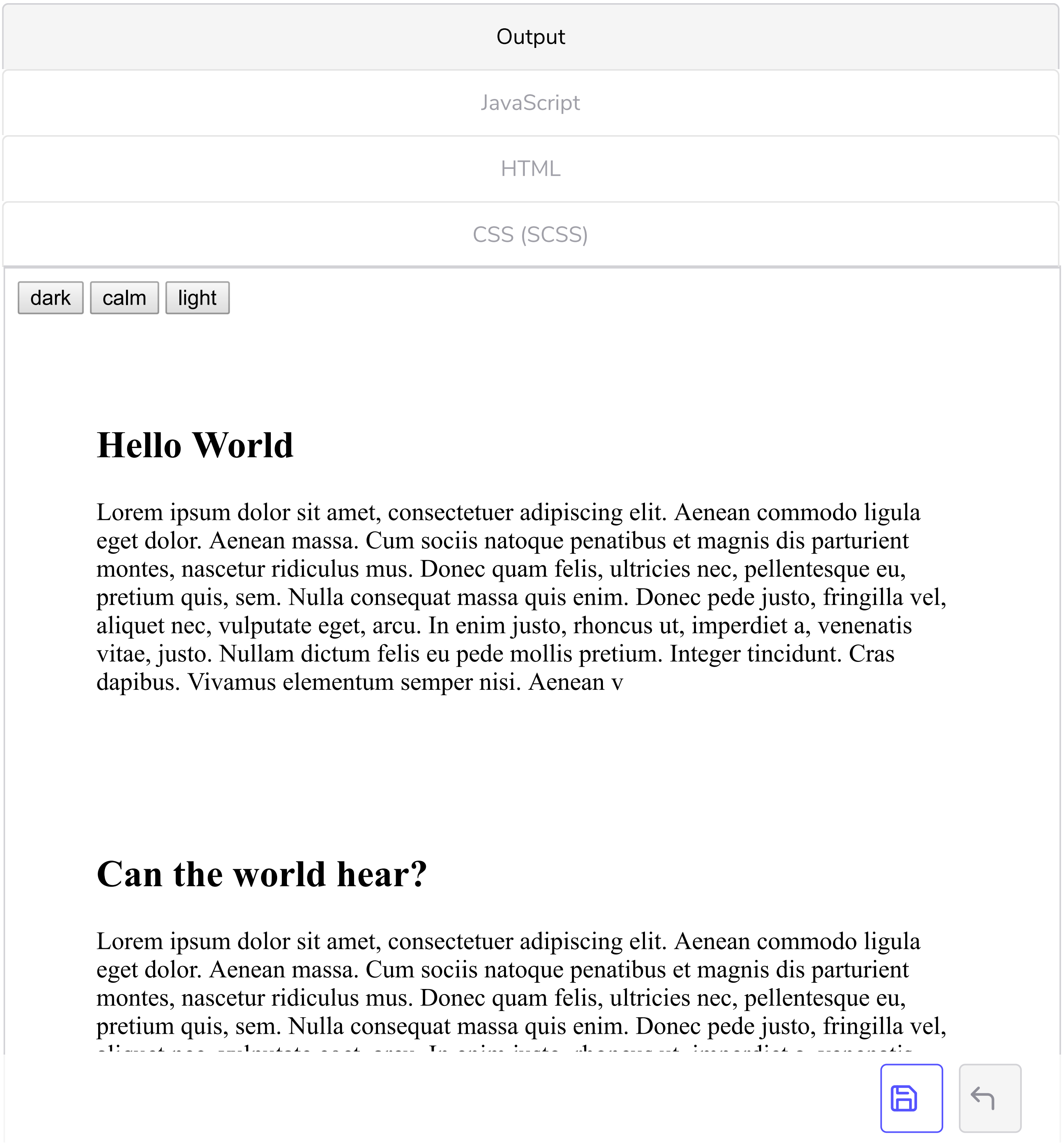
Creating Themed Sites with CSS Variables

I'm sure you've come across them before. Themed sites give the user the feel of customization. Like they are in control.

Below is the basic example we'll build.

Don't forget to click the buttons!



Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean v

Can the world hear?

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, So, how easy do CSS variables make this?

We'll have a look.

Just before that, I wanted to mention that this example is quite important. With this example, I'll introduce the concept of updating CSS variables with Javascript.

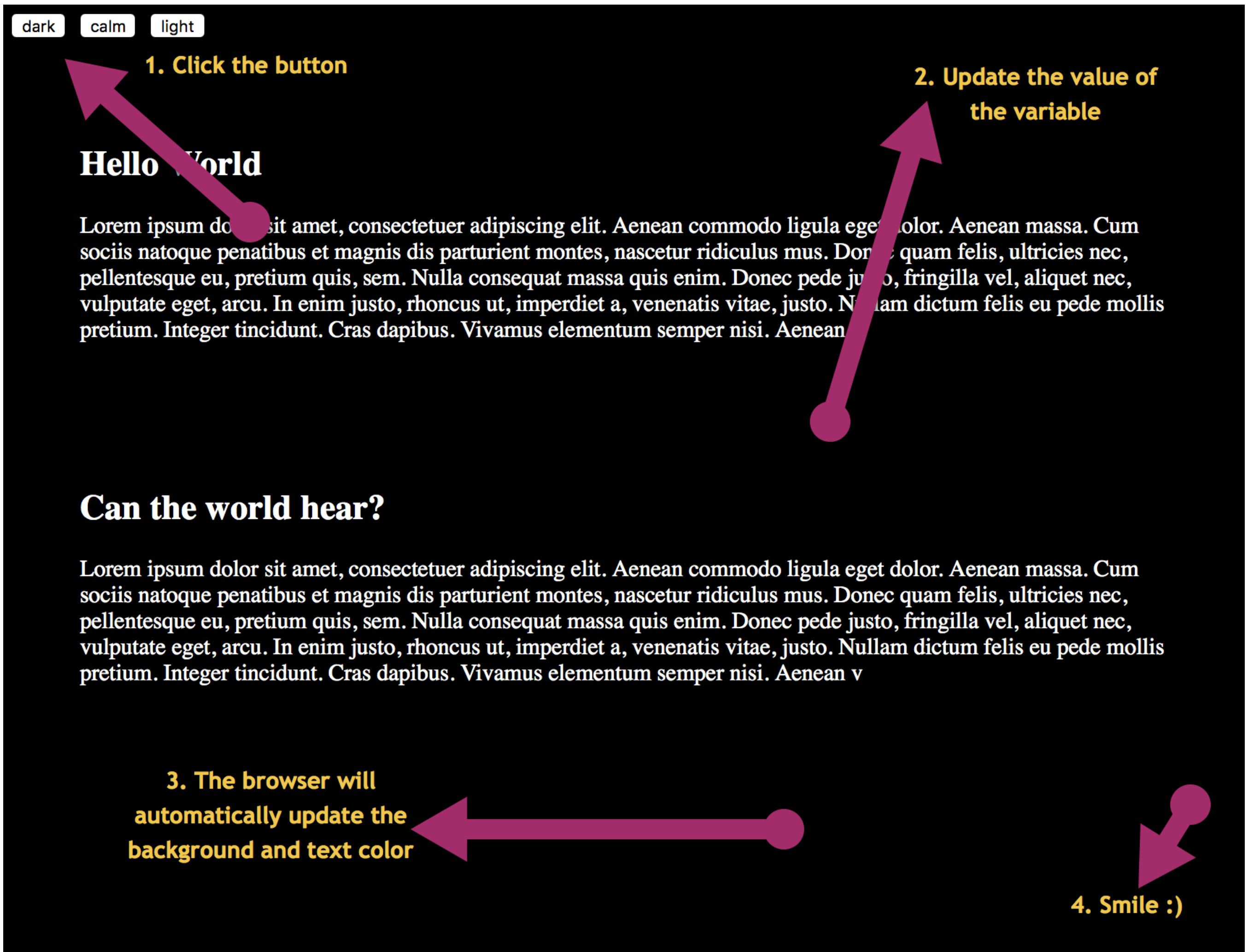
It is fun!

You'll love it.

What we really want to do.

The beauty of CSS variables is their reactive nature . As soon as they are updated, whatever property has the value of the CSS variable gets updated as well.

Conceptually, here's a image that explains the process with regards to the example at hand.


dark
calm
light
1. Click the button
2. Update the value of the variable
3. The browser will automatically update the background and text color
4. Smile :)

Hello World

Lorem ipsum do sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget olor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Don quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede ju o, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. N am dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean

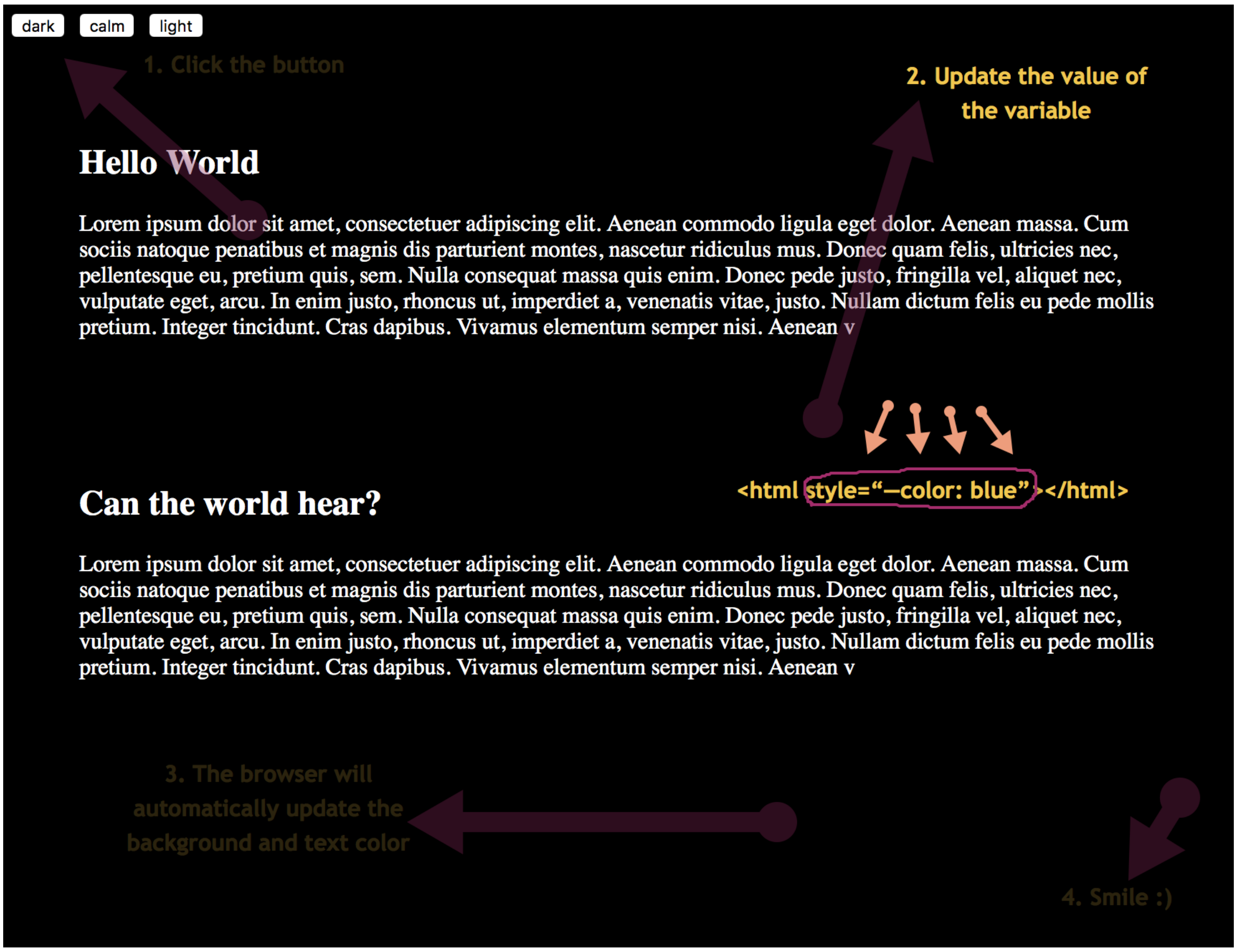
Can the world hear?

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean v

Obviously we need some Javascript for the click listener.
For this simple example, the background and color of the text of the entire page are based on CSS variables. When you click any of the buttons above, they set the CSS variable to some other color. As a result of that, the background of the page is updated.

Hey, that’s all there is to it. Uh, one more thing.

When I say the CSS variable is set to some other value, how’s that done?


dark
calm
light
1. Click the button
2. Update the value of the variable
<html style="–color: blue"></html>
3. The browser will automatically update the background and text color
4. Smile :)

Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean v

Can the world hear?

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean v

CSS variables will take effect even if they are set inline. With Javascript, we get a hold of the root document, and we set the new value for the CSS variable inline.

Got that?

That’s a lot of talking, let’s do the real thing.

The initial markup

The initial markup

The initial markup needed is this:

```
<div class="theme">
  <button value="dark">dark</button>
  <button value="calm">calm</button>
  <button value="light">light</button>
</div>

<article>
...
</article>
```

The markup consists of three buttons within a `.theme` parent element. To keep things short I have truncated the content within the `article` element. Within this `article` element is the content of the page.

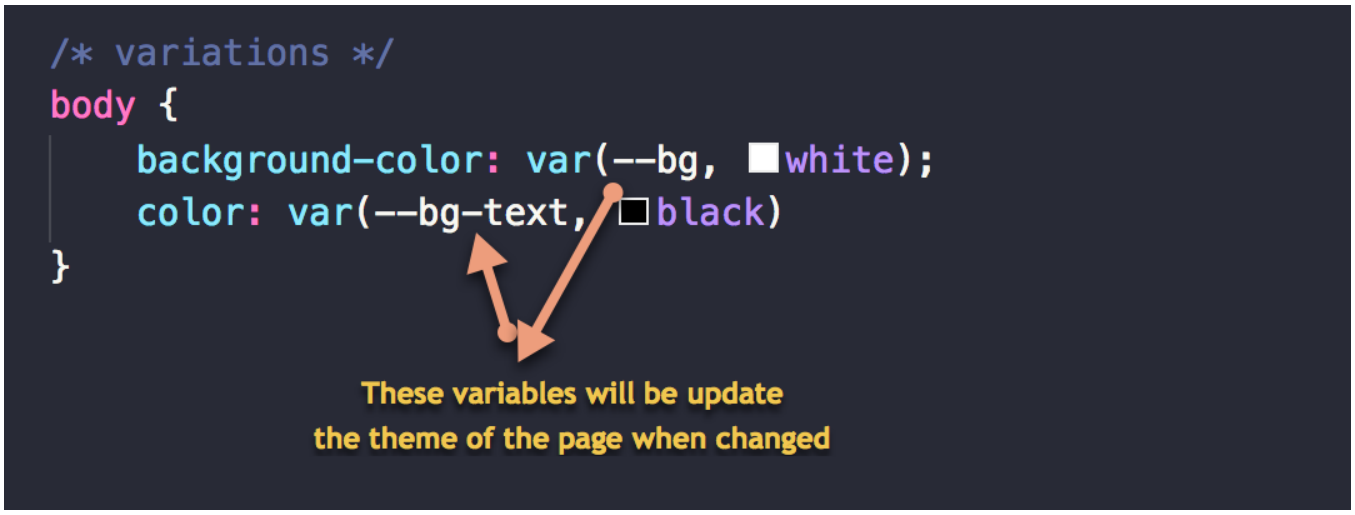
Styling the Page

The success of this project begins with the styling of the page. The trick is simple. Instead of just setting the `background-color` and `color` of the page in stone, we will have them based on variables.

Here's what I mean.

```
body {
   background-color: var(--bg, white);
   color: var(--bg-text, black)
}
```

The reason for this is kind of obvious. Whenever a button is clicked, we will change the value of both variables within the document. Upon this change, the overall style of the page will be updated. Easy-peasy.


```css
/* variations */
body {
    background-color: var(--bg, white);
    color: var(--bg-text, black)
}
```
These variables will be update
the theme of the page when changed

So, let's go ahead and handle the update from Javascript.

Getting into the Javascript

I'll go ahead and spit out all the Javascript needed for this project.

```
const root = document.documentElement
const themeBtns = document.querySelectorAll('.theme > button')

 themeBtns.forEach((btn) => {
   btn.addEventListener('click', handleThemeUpdate)
 })

 function handleThemeUpdate(e) {
   switch(e.target.value) {
     case 'dark':
       root.style.setProperty('--bg', 'black')
       root.style.setProperty('--bg-text', 'white')
       break
     case 'calm':
        root.style.setProperty('--bg', '#B3E5FC')
        root.style.setProperty('--bg-text', '#37474F')
       break
     case 'light':
       root.style.setProperty('--bg', 'white')
       root.style.setProperty('--bg-text', 'black')
       break
   }
 }
```

Don’t let that scare you. It’s a lot easier than you probably thought.

First off, keep a reference to the root element, `const root = document.documentElement` The root element here is the `HTML` element. You’ll see why this is important in a bit. If you’re curious, It is needed to set the new values of the CSS variables.

Also, keep a reference to the buttons too, `const themeBtns = document.querySelectorAll('.theme > button')`

`querySelectorAll` yields an array like data structure we can loop over. Iterate over each of the buttons and add an event listener to them, upon click.

Here’s how:

```
themeBtns.forEach((btn) => {
  btn.addEventListener('click', handleThemeUpdate)
})
```

Where’s the `handleThemeUpdate` function? I’ll discuss that next.

Every button being clicked will have the `handleThemeUpdate` as its callback function. It becomes important to note what button was clicked and then perform the right operation. In the light of that, a switch `operator` is used, and some operations carried out based on the value of the button being clicked.

Go ahead and take a second look at the block of Javascript code. You’ll understand it a lot better now.