6 Flex-box Photo Gallery

1

This week, the code we will be writing will also only be an insert for your own site. So, you would place the html code presented here inside of your main content area.

2

We will again be using flexbox to get our images to behave. We will place the entire thing in a container, but this time we will be doing things just a bit differently. That is because our container will not be set to flex. It will be the unordered list that we place inside of that container that we will be setting to display as flex.

3

I did put a background color into the container tag, but that is all I did.

4

Notice above that the ul which is given the tag of image-gallery is also given a flex-wrap of wrap. so that as the screen layout narrows, the images that don’t fit begin to rearrange themselves to accommodate the small screen widths till it finally is just a single column for the cell phone display.

5

In that same CSS rule above, we also justify-content to be centered, and gave a small gap between each image so that they would not appear to be glued to one another.

This is the changes that we made to the list items and then to the image itself that is in the list. We gave the list a pointer, so we can see that it will do something when we hover over each image. The rest of the rules in the following tags are pretty self- explanatory.

6

You can see from the HTML, that the image, and the overlay resides in-between each li opening and closing tag.

7

Here are the rules for the overlay and the hover. You can see we have some interesting transitions going on here. Our background is given some transparency by adding the little **a** after the **rgb** colors.

Setting the **scale** to 0 by using **transform,** is a little convoluted, but it has to do with visibility of the element. Its size will be 0,0, which means nothing, so you cannot see it. When we are not hovering, the overlay will have a size of nothing. But things that you have set up for other elements on the page, such as float or clear, will still work. So, in other words it doesn’t try to break things.

The **transition** here has to do with the speed that the overlay enlarges to cover the image, and then shrinks back in and disappears. It will set things in seconds, which is signified by the little **s** which follows the numbers. The first number is how many seconds it takes for the width to move, and the second number is talking about the height. Ease-in-out, means that it will ease in smoothly and then ease out smoothly, or slowly. The **all** at the beginning of the transition rule, just means that the movement should be applied to all of the properties.

8

You can go HERE to see all of this code.