Student name: Zholshiyeva Arailym

ID: 190103028

Class: Web programming fundamentals (practice)

***Article 2***

Creating a photo gallery using CSS

The step-by-step plan for creating this gallery is as follows:

1. Developing a responsive grid.
2. Decoration of miniatures.
3. Styling signatures.
4. Final touches.
5. Optional: plug-in for pop-up windows.

*1. Development of a responsive grid*

The first thing we'll start with is creating the grid for our gallery. A grid is a kind of skeleton that defines the layout of elements on a web page. The container block will have a maximum width of 960 pixels, and the thumbnails will line up in three columns of equal width (remember, we are building on the layout). Each side of the thumbnail will have a padding of 10 pixels.

Based on the words above, let's write the first style to the style.css file:

*container { width: 100%; / \* width of the container block \* /*

*max-width: 960px; / \* maximum container width \* /*

*margin: 0 auto; / \* this style will center the container \* /*

*}*

*.photo {*

*float: left; / \* tell the elements to line up one after the other \* /*

*width: 33.333333%; / \* set the width of the element \* /*

*padding: 10px; / \* add padding on each side \* /*

*box-sizing: border-box; / \* change how the width is calculated \* / }*

Let's check the responsiveness of our grid. Change the width of the browser window several times and observe the behavior of the elements. Elements less than 977 pixels wide begin to misbehave. This is because the thumbnail images do not adjust to the width of the container. Add the following style to make your images responsive:

*.photo img {*

*display: block;*

*max-width: 100%;*

*height: auto;*

*}*

Now everything works as it should, and even on small phones with a screen width of 320 pixels, our gallery is very easy to use and looks beautiful.

*2. Decoration of miniatures*

*Method 1: BORDER PROPERTY*

The first way is to create a border of the desired color and width for each <img> tag.

Add this code to your .photo img selector:

border: 10px solid #eee;

box-sizing: border-box;

The first line sets the border, and the second changes how the thumbnail size is calculated so that our border is included in the total width. Otherwise, the thumbnails would spill out of the container.

Refresh the page in the browser and remember the result.

*Method 2: BACKGROUND-COLOR AND PADDING PROPERTIES*

The second way is to mimic the look of the frame by creating 10-pixel padding and painting the background a light gray. Replace the code from the first method with the following:

padding: 10px;

background-color: #eee;

box-sizing: border-box;

*Hover style*

It's time for us to remember the pseudo-classes and pseudo-elements that we studied in the first part of the tutorial. Today we need the :hover pseudo-class to define the style of the thumbnail when the cursor is hovering over it, and the :after pseudo-element to add specific content after each thumbnail.

*.photo a:hover {*

*opacity: 0.5;*

*}*

And also let's make the <a> tag behave like a block, otherwise we won't see how the previous code works on the thumbnail:

*.photo a {*

*display: block;*

*position: relative;*

*}*

Let's add an eye icon. To do this, we need to use the :after pseudo-element. Add the following styles to your table, and then we'll explain why each is needed:

*.photo a:after {*

*content: '';*

*background: transparent url(../img/icons/eye-icon.png) no-repeat center;*

*width: 52px;*

*height: 35px;*

*position: absolute;*

*margin: auto;*

*top: 0;*

*left: 0;*

*bottom: 0;*

*right: 0;*

*visibility: hidden;*

*}*

Then,

*.photo:hover > a:after {*

*visibility: visible;*

*}*

This seemingly strange selector tells the browser to style the :after pseudo-element of the <a> tag, which is a child of .photo, when hovering over the .photo element. The visibility: visible style itself means that the eye icon becomes visible.

*3. Styling signatures*

Ideally, a caption for a photo should be concise and not overly conspicuous. Therefore, our style for her will be very simple:

*.photo figcaption {*

*font-family: 'Open Sans', sans-serif;*

*color: #999999;*

*text-align: center;*

*margin-top: 20px;*

*}*

*4. Final touches*

We haven't styled the title above the gallery yet. Let's do that:

*h1 {*

*font-size: 36px;*

*text-transform: uppercase;*

*color: #cccccc;*

*text-align: center;*

*margin: 30px 0;*

*}*

*body {*

*font-family: 'Open Sans', sans-serif;*

*}*

And so that the gallery block itself does not rest at the bottom of the web page, add a small margin to it from the bottom:

*#gallery {*

*margin-bottom: 50px;*

*}*

Then,

*#gallery:after {*

*content: '';*

*display: block;*

*height: 0;*

*clear: both;*

*}*

*5. Connecting a plugin for pop-ups*

It's time to use the Lightbox plugin, which will provide a beautiful opening of full-size photos in a pop-up window, without leaving the page, and also add the ability to flip through photos right on the spot.

The Lightbox plugin, powered by the jQuery JavaScript library, is used to overlay images on top of the current page. It's a handy tool that works in any modern browser.

Duplicate the attribute for each of the nine links in our gallery:

*<a href="img/originals/img-01.jpg" data-lightbox="roadtrip">*

*<img src="img/img-01-min.jpg" alt="Eagle" />*

*</a>*

If you wish, you can make the plugin display the caption to the photo in the open window. To do this, add one more attribute to the link - data-title = "", and inside its quotes place the signature text by copying from the <figcaption> tag:

*<a href="img/originals/img-01.jpg" data-lightbox="roadtrip" data-title="Eagle">*

*<img src="img/img-01-min.jpg" alt="Eagle" />*

*</a>*

Repeat these steps for the rest of the links and check the result.

Sources: <https://idg.net.ua/blog/uchebnik-css/ispolzovanie-css/praktika-fotogalereya-css>