

Due: Monday July 15 (11:59PM)

Submission

Put your assignment in a folder named `assign2_userid` (e.g., `assign2_ealmasri`). Put all resources used by your assignment into this folder. Compress this folder into a zip file (call it `assign2_userid.zip`). Submit a copy of this compressed file via Canvas (Assignments → Assignment 2).

Overview

The intent of this assignment is to demonstrate your understanding of JavaScript and jQuery. In this assignment, you will build a photo gallery using jQuery for our travel photo sharing site as shown in Figure 1. The required starting files to complete the assignment can be found in a compressed file called ***assign2-start.zip*** which can be downloaded via Canvas (Assignments → Assignment 2 → `assign2-start.zip`).

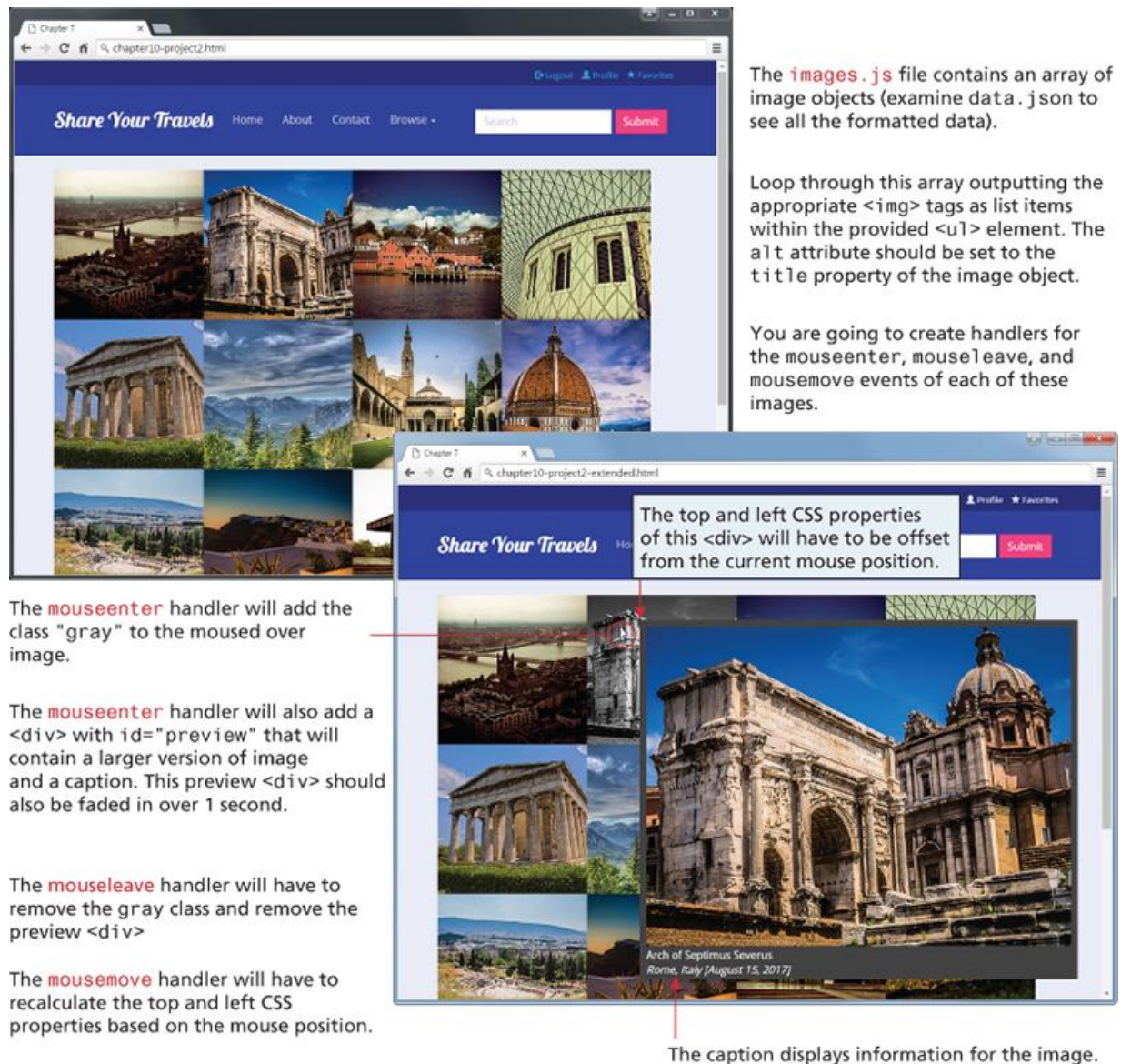


Figure 1: Mockup of Assignment 2

Instructions

1. Examine `assign2.html` in the browser and then editor. You have been supplied with the appropriate CSS (the relevant classes are in `gallery.css`), html, and JavaScript data files (an array of image objects are in `images.js` file). The data is minimized in that file so there is an additional file called `data.json` which contains the data in an easy-to-read format. The images are supplied in two folders: `images/square` (for the gallery) and `images/medium` (for the popup).
2. Loop through the images array and using the appropriate jQuery DOM methods, add the appropriate `` tags to the supplied `<ul class="gallery">` element. The image filenames are contained in the `path` property of each image object. Set the `alt` attribute of each `` to the `title` property of the image object.
3. Use jQuery to attach handlers for the `mouseenter`, `mouseleave`, and `mousemove` events of the square images in the gallery.
4. For the `mouseenter` event, use jQuery to add the “gray” class to the square `` under the mouse. If you examine that class, you will see it sets the `filter` property to `grayscale()`. Hint: remember that `$(this)` within an event handler references the DOM object that generated the event.
5. Also for the `mouseenter` event, use jQuery to generate a `<div>` with an `id="preview"` (the styling for `#preview` is already defined in `gallery.css`). Within that `<div>` add an `` element that displays the larger version of the image. Underneath that `` add a `<p>` element for the caption. The information for the caption and image are contained within the `images` array. The `alt` attribute of the square image under the mouse contains the image title. You can search through the `images` array looking for a match on the title; once a match is found, you have the file path, city, country, and date information.
6. You will need to use jQuery to set the `left` and `top` CSS properties for the `#preview<div>`. You can retrieve the x, y coordinates (via the `pageX` and `pageY` properties) of the current mouse position from the event object that is passed to your event handler. You can calculate the new position by offsetting by some amount from the mouse x, y position.
7. Finally, once the `#preview <div>` is constructed, simply append it to the `<body>`.
8. For the `mouseleave` event, remove the “gray” class from the square image under the mouse. Also remove the `#preview<div>` from the body.
9. For the `mousemove` event, simply set the `left` and `top` CSS properties for the `#preview <div>` using the same approach as described in step 6.

Testing

10. Verify the code works when **mousing** over the images. Be sure that the caption is displaying the correct information.
11. Don't worry if the pop-up image is “off screen” when **mousing** over images on the edges of the browser.