10.7 Chapter Summary

This chapter provided an overview of the main features of the jQuery framework. While there is plenty of jQuery content that we did not have the space to cover, the chapter did cover selectors, filters, event handling, animation, as well as asynchronous communication and file uploading.

10.7.1 Key Terms

Animation
Asynchronous JavaScript with XML (AJAX)
content delivery network (CDN)
content filters
cross-origin resource sharing (CORS)
easing function
filters
framework
FormData
graceful degredation
jQuery
jqXHR
library
progressive enhancement

10.7.2 Review Questions

1. What is a web framework? What types of features are expected in a typical JavaScript framework?
2. What does the `$()` shorthand stand for in jQuery?
3. Write a jQuery selector to get all the <p> elements that contain the word "hello."
4. jQuery extends the CSS syntax for selectors. Explain what that means.
5. How would you change the text color of all the `<a>` tags using jQuery?
6. What is the difference between the `append()` and `appendTo()` methods?
7. Write a jQuery click event handler for all `<img>` tags within `<div>` elements. In the handler, output the `src` attribute of the image to the console.
8. What are the advantages of using asynchronous requests over traditional synchronous ones?
9. What makes a HTTP method safe?
10. Why would you use jQuery animations over CSS transitions?
11. What is cross origin resource sharing? What relevance does it have for jQuery applications using asynchronous requests?

10.7.3 Hands-On Practice

HANDS-ON EXERCISES
Project 10.1

PROJECT 1: Art Store

DIFFICULTY LEVEL: Easy

Overview

Use jQuery to respond to events and to programmatically modify HTML and CSS as shown in Figure 10.21.

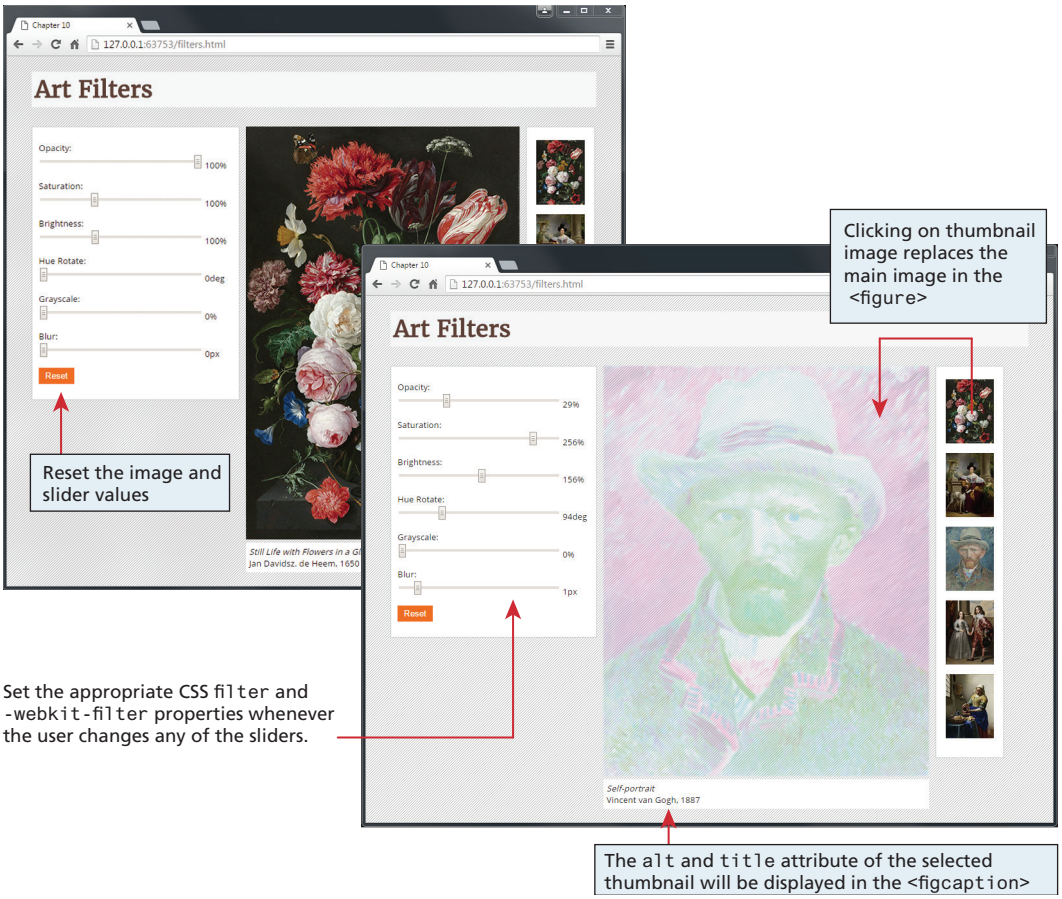

FIGURE 10.21 Project 1

Instructions

1. Examine **lab10-project1.html** in the browser and then editor. You have been supplied with the necessary CSS and HTML.
2. Import jQuery in the `<head>` of the page.
3. Use jQuery to respond to click events on the painting thumbnails. Replace the `src` attribute of the `<img>` element in the `<figure>` so that it is displaying the clicked painting. Hint: get the `src` attribute of the clicked element and then replace the `small` folder name with `medium` folder name.
4. As well, change the `<figcaption>` so that it displays the newly clicked painting's title and artist information. This information is contained within the `alt` and `title` attributes of each thumbnail.

5. Set up event listeners for the `input` event of each of the range sliders. The code is going to set the `filter` and the `-webkit-filter` properties on the image in the `<figure>`. Recall from Chapter 7 that if you are setting multiple filters, they have to be included together separated by spaces.
6. Add a listener for the click event of the reset button. This will simply remove the filters from the image.

Testing

1. To test, click on the thumbnails and verify the correct caption is displayed. Ensure the filters work as expected.

PROJECT 2: Travel

DIFFICULTY LEVEL: Intermediate

HANDS-ON EXERCISES

Project 10.2

Overview

This project will build a photo gallery using jQuery for our travel photo sharing site as shown in Figure 10.22.

Instructions

1. Examine **lab10-project2.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 `<img>` 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 `<img>` 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 `<img>` 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 `<img>` element that displays the larger version of the image. Underneath that `<img>` 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