## PROJECT 1: Art Store

**DIFFICULTY LEVEL: Intermediate**

### Overview

Demonstrate your proficiency with absolute positioning and floats by modifying **chapter07-project1.css** so that **chapter07-project1.html** looks similar to that shown in Figure 7.55.

### Instructions

1. Examine **chapter07-project1.html** in the browser. The HTML does not need to be modified for this project.



Use absolute positioning to place banner text on top of banner image.

**Dutch Portraits of the Golden Age**
From the Rijks Museum

Start

Float this `<div>` left

**Latest Additions**

Portrait of Maritge Claesdr Vooght
*Frans Hals, 1639*
Maritge Vooght is here portrayed in a traditional pose, proudly sitting upright and looking straight out at the viewer

Portrait of Johannes Wtenbogaert
*Rembrandt, 1633*
Wtenbogaert's face is more realistically modelled than his hands, which may have been done by a pupil in Rembrandt's workshop

Float this container to the left.

Read More

Float this link right

Read More

map of images

| a | b | d |
| e | f g | h |
| i | j | k |
| | | c |

285px × 390px

10px space

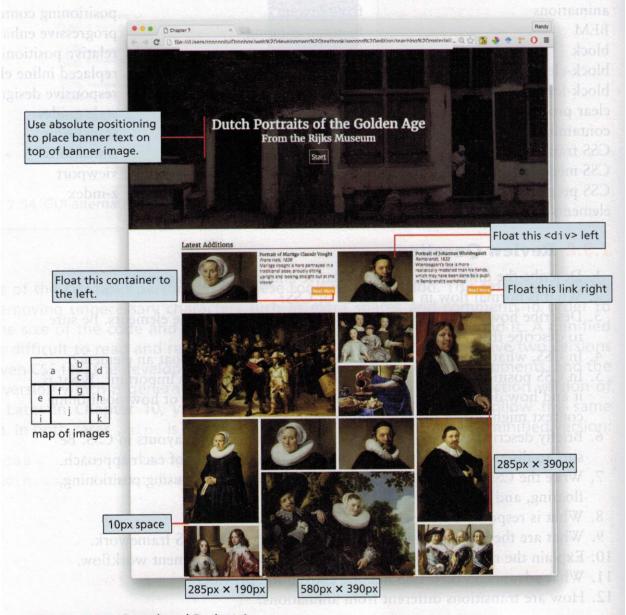285px × 190px    580px × 390px

**FIGURE 7.55** Completed Project 1

2. You have three layout tasks. The first will be the large text on top of the banner image in the header. Use absolute positioning to place the `<div>` with the banner—title class on top of the banner image. The exact position is not important; just try to get it approximately in the center. Also style the rest of the banner title text.

3. The next layout task is the highlights section. You will use a float to move the image within the highlights—media element to the left of the text. The highlights—buttonlink within the highlights—text element will be floated to the right. You will also use a float to move the highlights—container elements to the left.

4. The final layout task will use absolute positioning to construct the mosaic of paintings. Remember that absolute positioning is relative to the last positioned ancestor. We recommend that you use relative positioning on the mosaic container. You can then use absolute positioning for each mosaic image.

## Testing

1. View chapter07-project01.html in the browser. It should look similar to that shown in Figure 7.55. Note that you will need a wide browser window on a desktop machine.

2. Try resizing the browser window and make it smaller. The layout will be a mess! It illustrates one of the main problems with complicated positioning layouts: they don't scale well to smaller browser windows.

## PROJECT 2: Book CRM

**DIFFICULTY LEVEL:** Intermediate

### Overview

Use the flexbox layout mode and media queries to create a responsive layout.

### Instructions

1. Open **chapter07-project02.html** in the browser. You will be modifying the CSS only.
2. Modify **styles.css** and float the `<h1>` in the header to the left and the vertical line menu image to the right.
3. Right now each card fills the entire width of the available space. Change the width of the card class to 24%. By taking less than a quarter of the available space, we will eventually be able to fit four cards on a row.
4. Now we need to use the flexbox mode. You will need to add display:flex to the cards class.
5. Change the max-width property of the `<figure>` image to 100%.
6. Modify the card class by setting its flex property to 0 1 auto. Test in browser. Set the justify-content, align-items, and flex-wrap properties appropriately in order to achieve a layout similar to that shown in Figure 7.56.

7. Trying resizing the browser; notice how the flex containers continue to shrink in width in order to maintain the four columns. Why four columns? Remember back in step three we set the width to 24%, so the browser is trying to maintain that rule.
8. Add a media query for screens 480 px wide and less. In it, change the card width to 100% and test. Now on a small screen, each card will fill the entire width. Also reduce the height of the header as well as the margin and padding of its heading.
9. Add a media query for tablets between 481 px and 768 px. Change the card width so two cards are displayed on each row.
10. Add a 2 second transition on the opacity property when hovering over or off of the See More span with the *button* class This will create the illusion of the span fading into (or out of) visibility. Also, add a drop shadow and a saturation filter of about 130% when hovering over any of the card images.

### Testing

1. View **chapter07-project02.html** in the browser. Be sure to test at different sizes to verify the media queries work as expected (see Figure 7.56).

**4**



Float to left

Float to right

Each card has `width: 24%`

Card container uses `display:flex`

When hovering over the card, add a `transition` on the opacity property of the button.

flexbox shrunk to smaller size keeps shrinking columns so that they take up 24% of available space. This needs to be fixed using media queries!

Tablet width after media query added.

Mobile width–notice that header shrinks in size also.

**FIGURE 7.56** Completed Project 2