### **Overview**

Positioning is a fundamental aspect of web development and must be understood to create successful webpage layouts. In this activity, you will gain practice with positioning by building three basic page layouts.

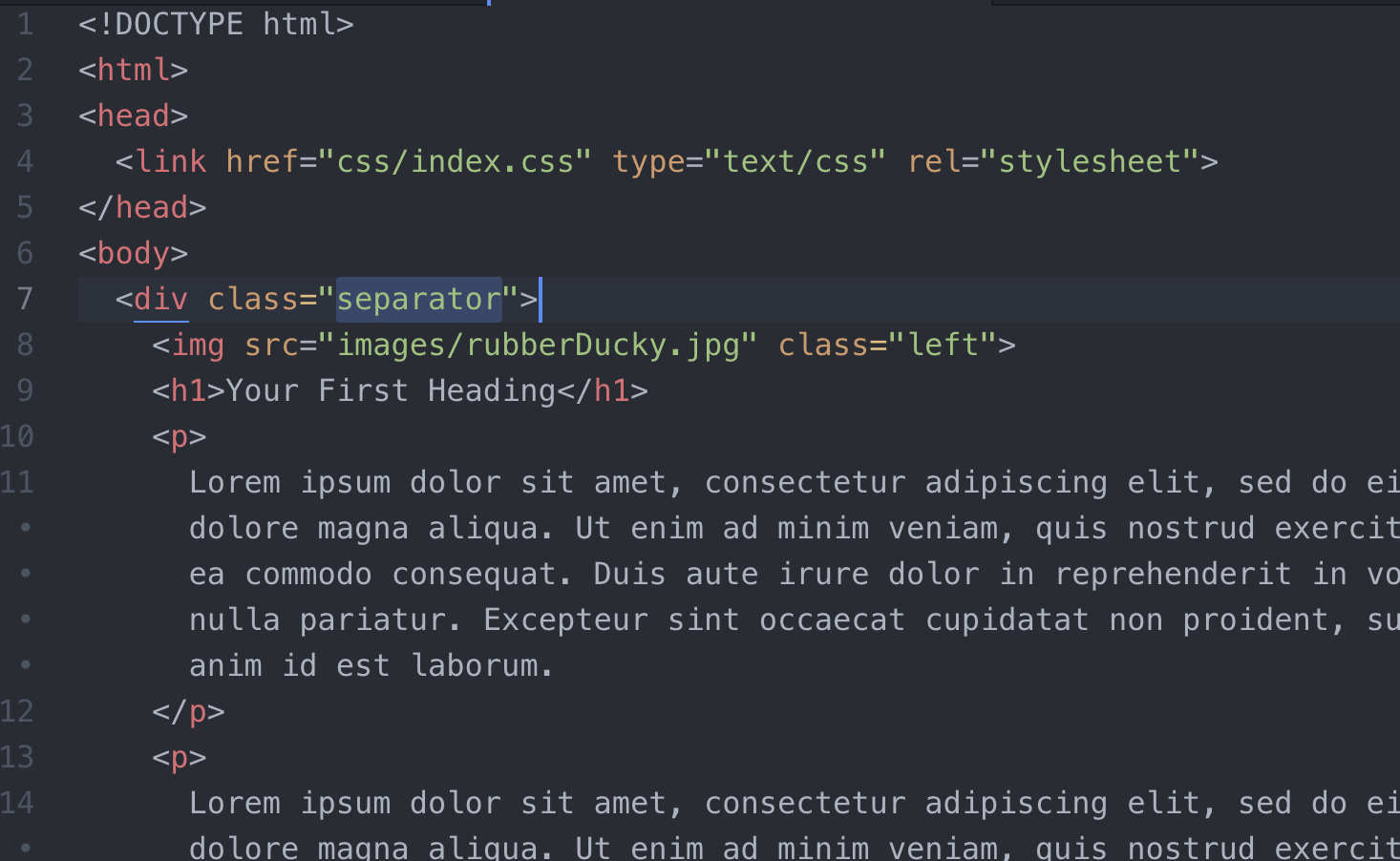
### **Instructions**

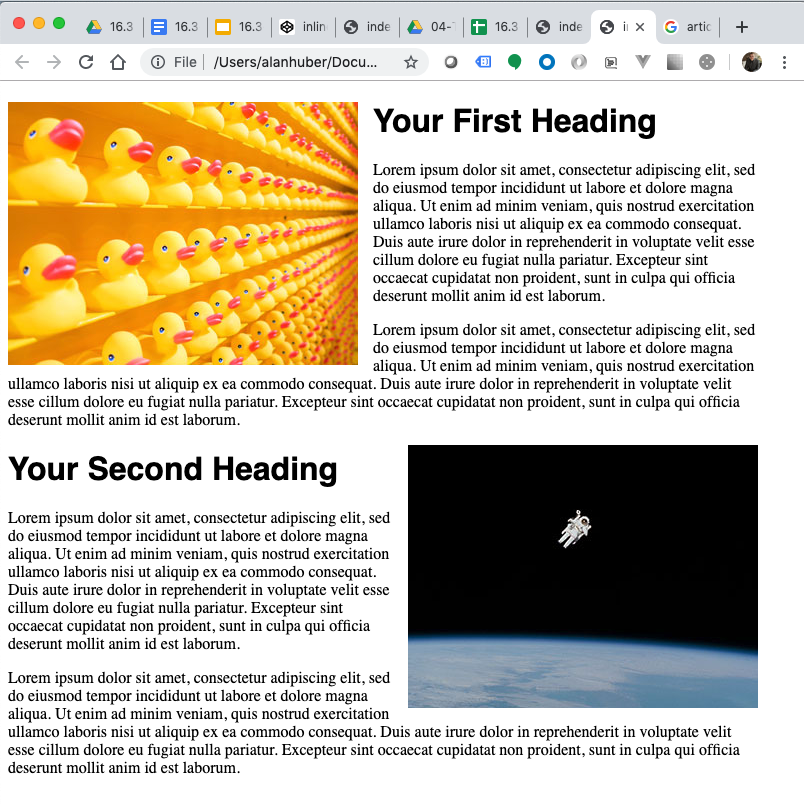
**Part 1: CSS Floats**

CSS Floats are primarily used to flow text around images. Floats also have an additional use case: CSS floats can be used to position elements inside containers to the farthest left or farthest right of the element.

**Note:** CSS Floats can also be used to position elements next to each other, but this can cause unexpected behavior for parent containers.

**Steps**

1. Open index.html contained in the folder named 1\_float.
   1. It’s located in the Unit 17 Activities folder, inside **17.3\_unsolved.zip**.
2. Open this index.html file in your web browser. Notice how the text sits near the bottom of the image? Let’s move it to the side of the first image.
3. Open index.CSS located in /unsolved/CSS/index.CSS.
4. Create a CSS selector that targets the img tag with the class of .left
5. Add the property float: left; to the CSS selector you just wrote.
6. Notice that your text is too close to the image. That’s normal. Your image needs some padding so that the text doesn’t sit so close to it. To fix that, add the property padding-right:15px; to the class of .left that you created in the previous step to space your text farther away from the paragraph content.
7. However, you have another image that needs to be positioned. The second image has a CSS selector of .right. Add .right to your stylesheet and give it the property float: right;
8. This element also needs some padding to move the text away from it. Give .right the CSS property of padding-left: 15px;
9. Notice that your text is spanning the full width of the page. This is because it’s a block-level element. You should give it a defined width so that the text looks better.
10. In your HTML, you have a div with the class of separator that is containing all the contents of your mock article.  
    
11. Write a CSS selector that targets the class of separator.
12. Give this selector the CSS property width: 750px;
13. Your page should look like the following image:



**Note:** You’ve just created a layout that has many potential uses: it could be an informational page, a blog post, or an article about a certain topic, depending on your needs.

**Resources:**

For more information on how floats can be used to position containers, read the article “[How to Clear Floats](https://www.w3schools.com/howto/howto_css_clearfix.asp)” by the W3Schools.

**Part 2: CSS Positioning Block-Level and Inline Elements**

***Block-level elements*** always start on a new line and take up the full width available. They are used to contain inline content. They can also contain other block-level elements if your container needs more structure. Additionally, they can be moved around your page using margin and automatically moved across the page using automatic margins.

***Inline elements*** do not start on a new line and only take up as much width as necessary. They are contained inside block elements when constructing content. Inline content contained inside a div can be controlled by adding the text-align property to the div that holds your inline elements.

**Steps**

1. Open index.html contained in the folder named 2\_block\_and\_inline\_elements, located in Activities/positioning\_elements/2\_block\_and\_inline\_elements/index.html.
2. Open this index.html file in your browser so you can see what it looks like before you style it.
3. In the index.CSS files located in CSS/index.CSS, there is already a style targeting an element with the class of blockElement. This is where most of your CSS is going to be directed. You have already added styles to this document, so that you can see these elements display on your page:

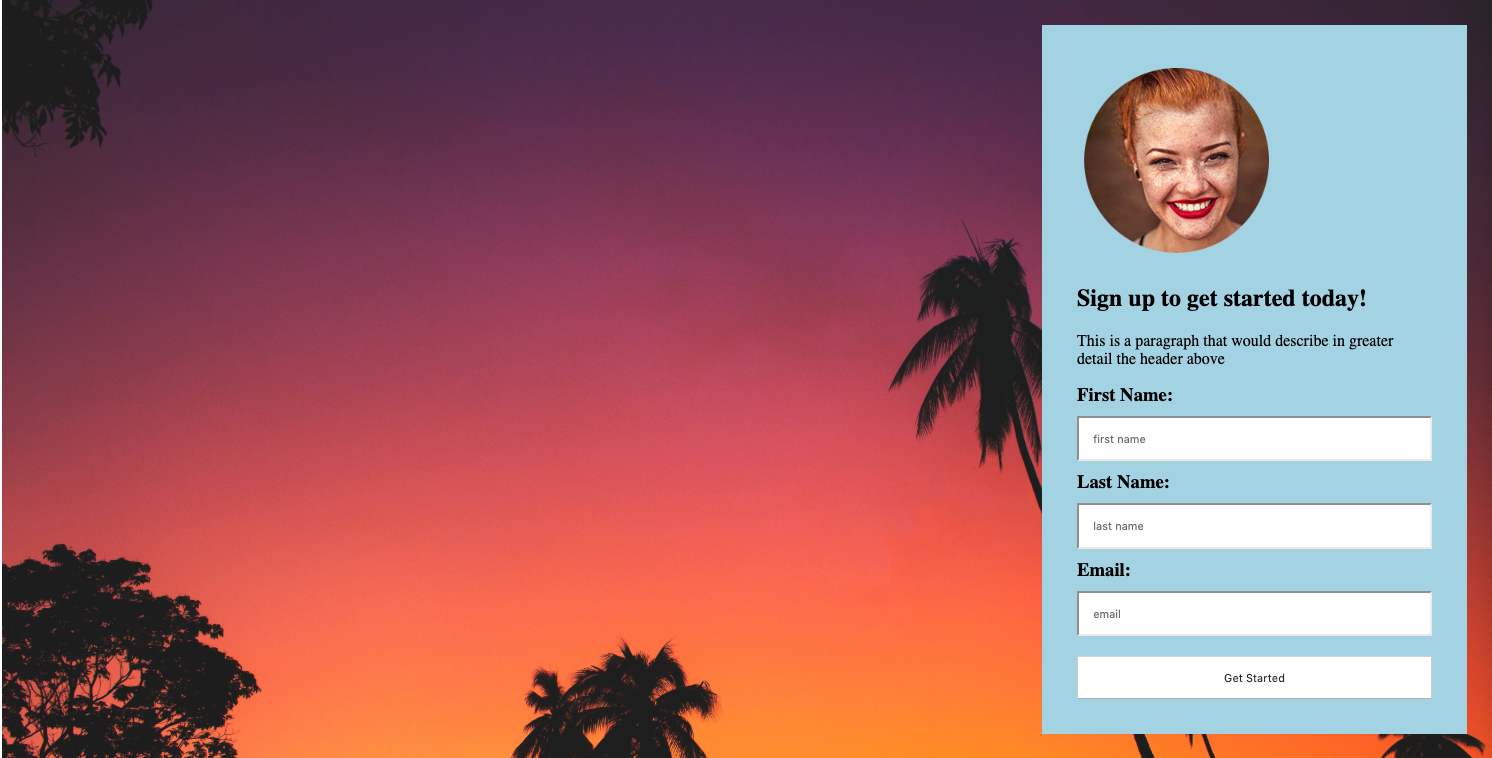
|  |
| --- |
| .blockElement {  } |

* *If you don't want to continually refresh the page, feel free to use Chrome’s Web Inspector to preview how your styles affect your HTML in real time!*

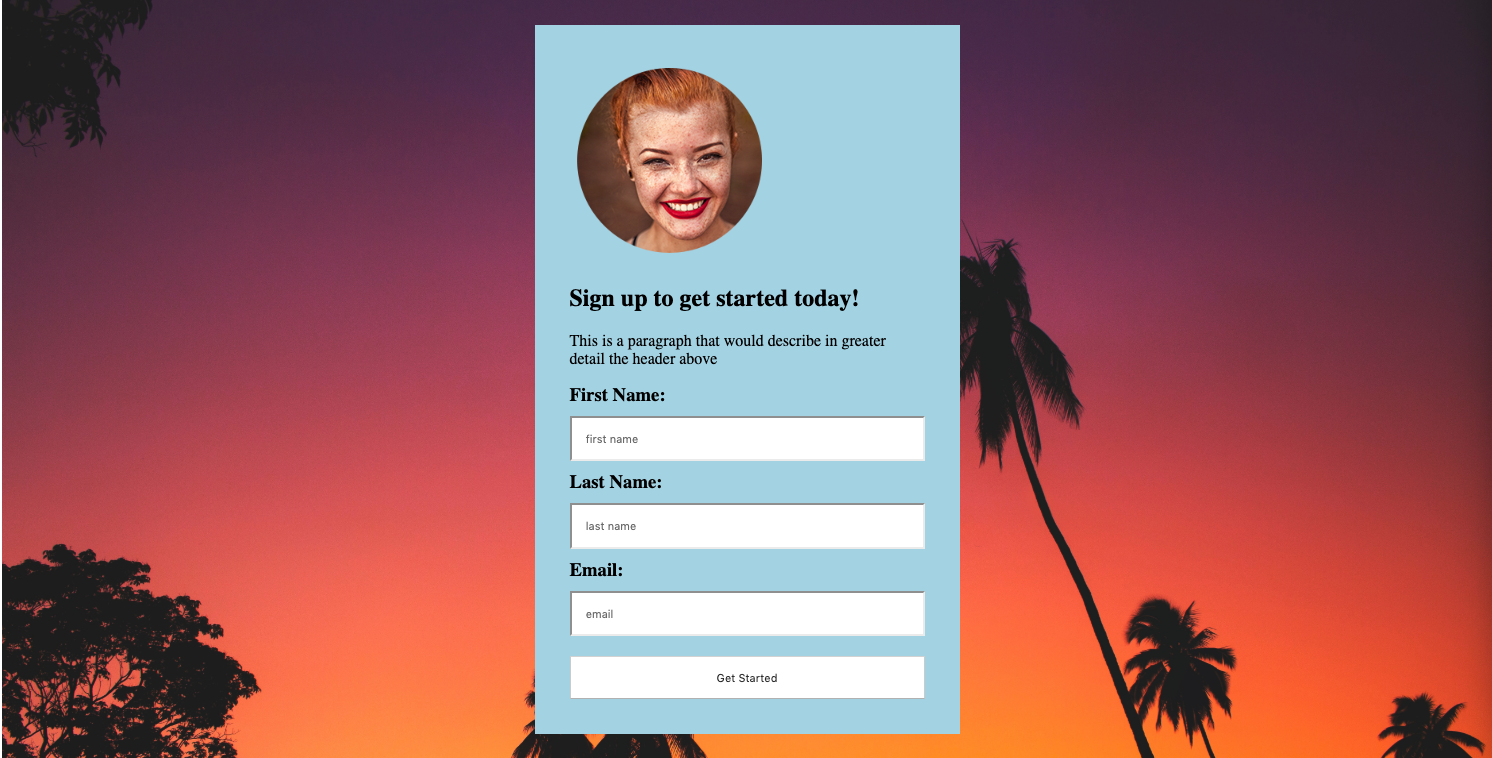
1. Notice how the div (which is a block-level element) that contains your other form already spans the entire width of this document without specifying in your CSS.   
     
   Block-level elements by default take up 100% width of their parent containers.
2. Add the CSS property margin-left: auto; to the class .blockElement.

**Note:** Any kind of auto margin will not work if your div does not have a defined width because you can’t center something that is already taking up 100% of the width.

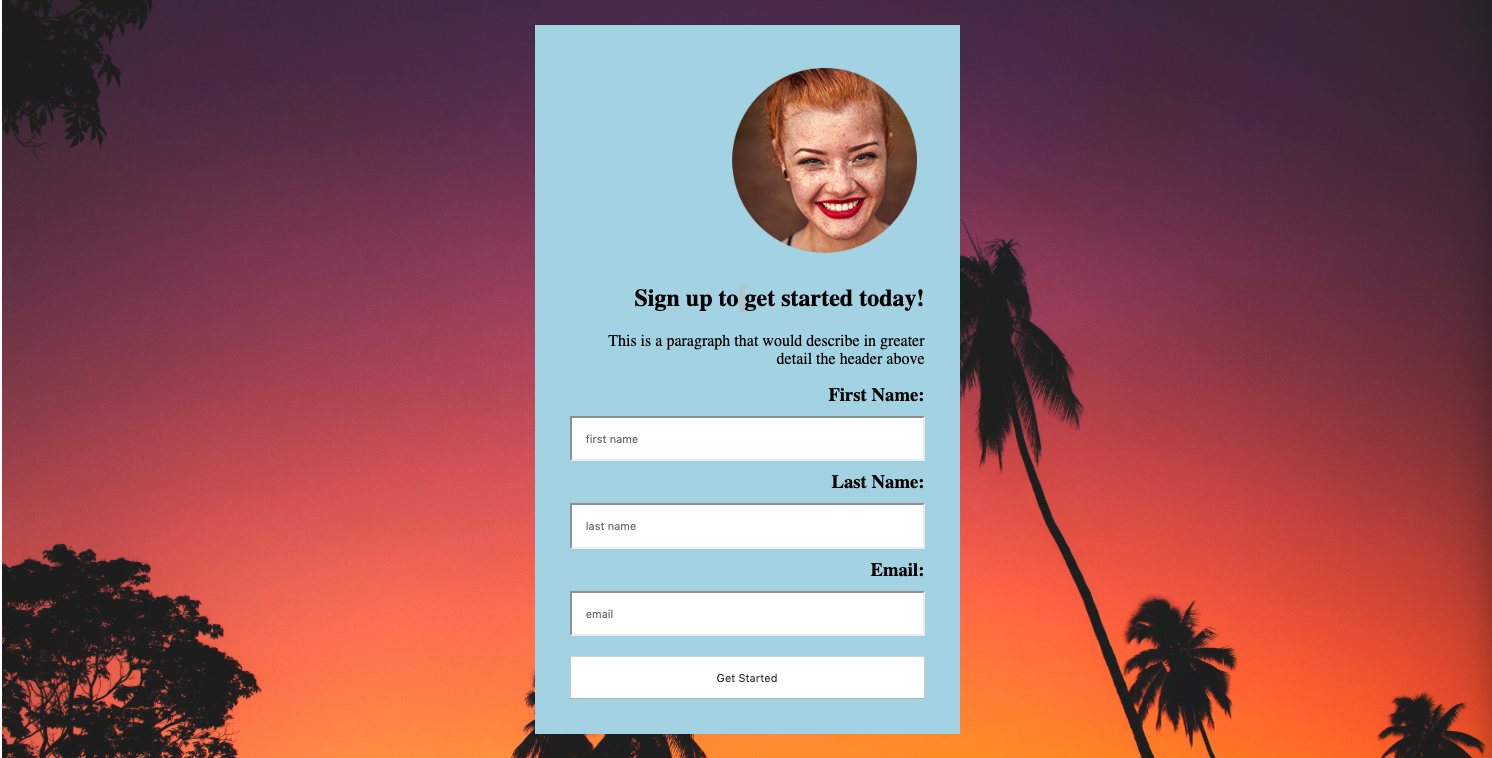
1. Reload the page to see how your element is now pushed to the far right.   
     
   Automatic margins are an important concept that will really help you build interesting layouts, as it allows you to position block-level elements inside a containing element.



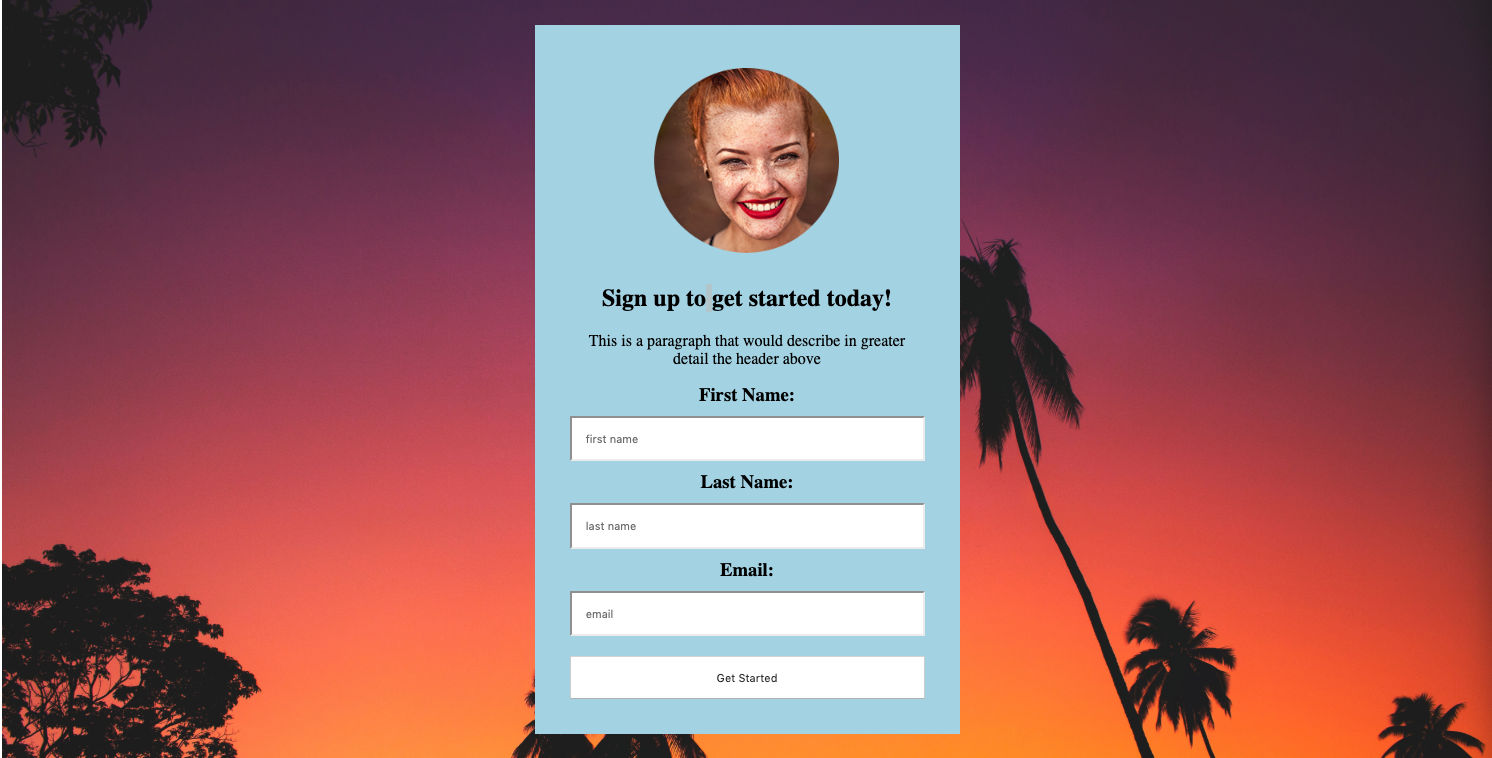
1. Now delete the property margin-left: auto; from the class .block\_element and add margin: 0 auto; to it instead. Notice how the element is now centered in its container:



1. Next, position your text using the CSS property text-align. Text-align is applied to the parent container and affects any inline element nested inside the parent tag (images, etc.).
2. Add the CSS property text-align: right; to the class .blockElement and reload your page (or use the web inspector). Notice how your inline elements are now positioned to the right of your div.



1. Now delete text-align: right; and add text-align: center;
2. Reload the page and see the results. You should see all your text aligned in the center of the element.



**Resources:**

* text-align: [https://www.w3schools.com/CSSref/pr\_text\_text-align.asp](https://www.w3schools.com/cssref/pr_text_text-align.asp)
* margin: [https://www.w3schools.com/CSS/CSS\_margin.asp](https://www.w3schools.com/css/css_margin.asp)

**Part 3: Positioning Inline-Block Elements**

***Inline-block elements*** can be used to create elements that line up next to each other, such as tabs, navigation items, or social media icons that all sit next to each other. They can also be used to create sections that line up next to each other.

**Steps**

1. Open index.html contained in the folder named 3\_inline\_block\_elements, located in Activities/positioning\_elements/3\_inline\_block\_elements/index.html.
2. Open this index.html file so that you can see how it looks before you style it.
3. Write a CSS selector that targets the HTML element with the class of inline-block.
4. In your selector with the class of .inline-block, add the CSS display property of inline-block: display: inline-block;
5. Give it a padding of 15px so your text is centered inside the containing div.

|  |
| --- |
| padding: 15px; |

1. Reload your page.
2. In your CSS file, write a selector that targets the HTML element with the class of left.

|  |
| --- |
| .left {  } |

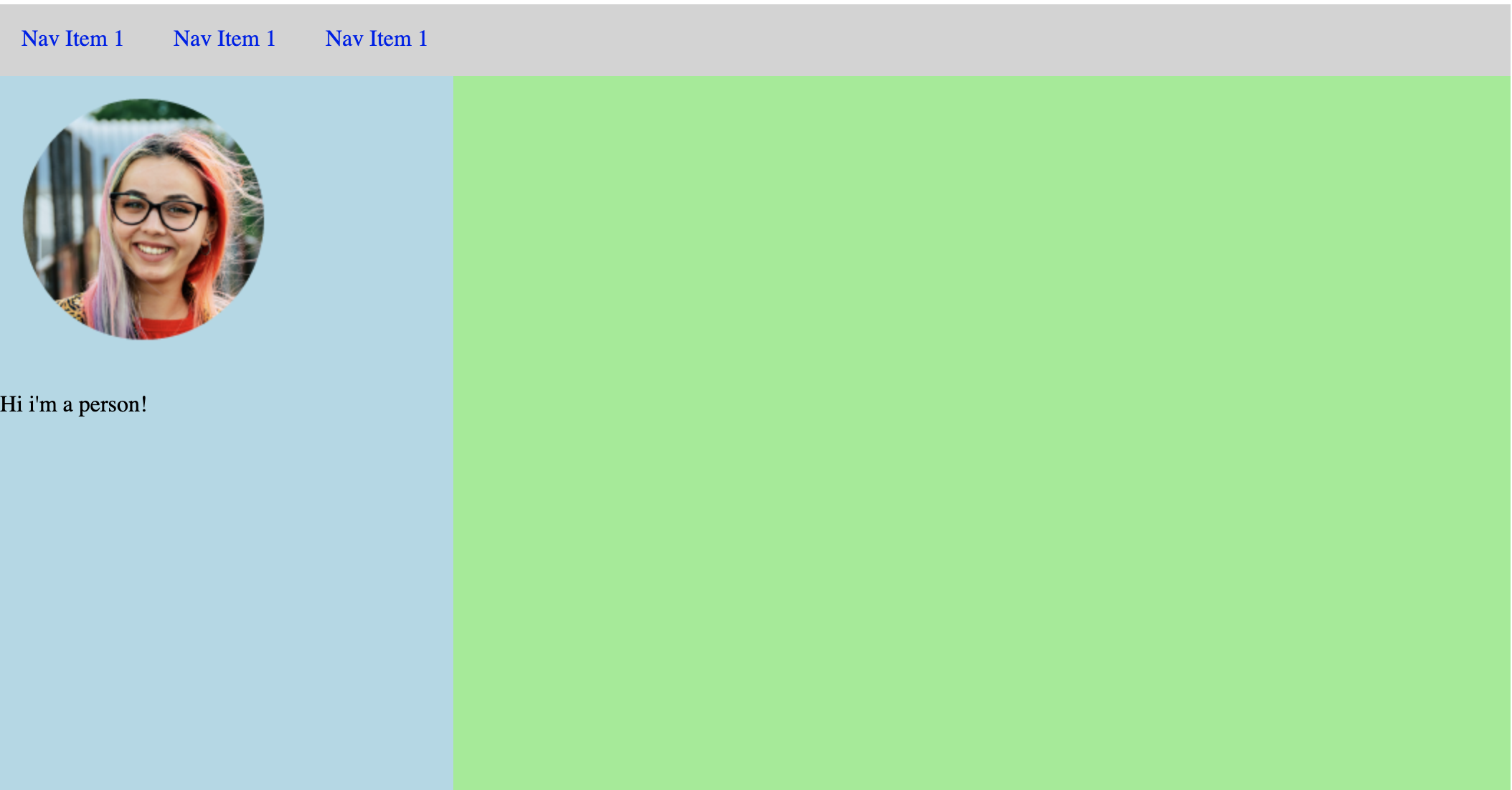
1. In your CSS file, write a selector that targets the HTML element with the class of right.

|  |
| --- |
| .right {  } |

1. Give both classes a height of 500px and float left.

|  |
| --- |
| .left { height: 500px; float: left;  } .right { height: 500px; float: left; } |

1. Give .left the CSS property of background-color with the value of light blue.
2. Give .left the CSS property of width with a value of 30%.
3. Give .right the CSS property of background-color with a value of light green.
4. Give .right the CSS property of width with a value of 70%.
5. Reload the page to preview your container.



1. You may have noticed that the container for .left has an image and p tag with text inside it. Let’s position these elements.
2. Give .left the property of text-align with a value of center.
   * Img tags are inline block-level elements that can be centered with text align just like the text contained inside your p tag.
3. You have some cards in the div with the class of .right. For those, write a CSS selector that targets the class of card. Give it the following properties:
   * Give it the CSS property display with value of inline-block.
   * Give it the CSS property background-color with a value of #C6C6C6.
   * Give it the CSS property of border-radius with a value of 5px.
   * Give it the CSS property of height with a value of 200px.
   * Give it the CSS property of width with a value of 150px.
4. Now that you have styled your cards, let’s position them.
5. In your class of .right, add the following properties:
   * Add the property text-align with a value of center.
   * These will center all of your cards together.
6. In your class of .card, add the following properties:
   * Add the property margin with a value of 35px.
   * Add the CSS property of margin-top with a value of 50px.

You have created three basic page layouts and practiced positioning elements on those pages including text, containers, and images.

**Part 4: Share your work on Slack**

Take screenshots of each of your layouts and post them in the class Slack channel.