**Welcome to the CSS Code Challenges**

In this CSS code challenge, you will complete challenges based on CSS concepts in our [Learn CSS](https://www.codecademy.com/learn/learn-css) lesson. The best way to reinforce these new skills is through practice. We’ve created a series of problems designed to use your new CSS knowledge.

As you complete these challenges, we encourage you to review past lessons, look things up in the [Mozilla Developer Network (MDN) CSS documentation](https://developer.mozilla.org/en-US/docs/Web/CSS). Check out the hints and solution code if you get stuck, and, most of all, have fun!

**Missing Semicolons**

Semicolons separate the declarations in a ruleset.

**Instructions**

**1.**

Check the rulesets #one, #two, #three, and #four and add in the missing semicolons.

Hint

The semicolon at the end of a declaration separates the declarations. For example:

.newClass {

width: 100%;

height: 100%;

}

**Fallback Font**

Add a fallback font in case the first font doesn’t load.

**Instructions**

**1.**

Currently, the body selector is using the font Roboto which is being imported using a CDN. In case the CDN doesn’t work properly add a backup font of Courier.

Hint

To add a backup font, add another font after the initial font and separate the fonts by commas. Remember to keep Roboto as the first font.

body {

  font-family: Roboto,Courier;

}

**Round Image**

Embrace web trends by changing a square image to a circular one.

**Instructions**

**1.**

Change the **style.css** file to display the <img> as a round circle.

Hint

Changing the border-radius property gives an element rounded curved corners. You can increase the percentage of border-radius until the <img> becomes a circle.

img {

border-radius: \_\_%;

}

img {

  height: 150px;

  margin: 0 auto;

  border-radius: 50%;

}

**Combinator**

Use a CSS combinator to make a ruleset more specific.

**Instructions**

**1.**

In the browser, the elements that have a class attribute of .recipe are bolded. In **style.css**, change the .recipe ruleset to only select the .recipe element nested within the <article> element.

Hint

You can use a combinator to select children or siblings of elements. In this case, you want to target a child element of <article> that has a class attribute of .recipe.

Read more about combinators at the [MDN CSS Combinators documentation](https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Combinators_and_multiple_selectors).

article .recipe {

  font-weight: bold;

}

**Inline HTML to CSS**

Take inline CSS and turn it into a proper stylesheet.

**Instructions**

**1.**

Remove all of the inline CSS inside **index.html** and replace it by writing the correct CSS selectors and rules inside **style.css**.

Hint

The following inline HTML:

<p style="font-size:12px;"></p>

Translates to the following CSS:

p {

font-size: 12px;

}

**2.**

To make the styles re-appear, link the stylesheet inside the <head> element in the **index.html** file.

Hint

The syntax for linking a stylesheet looks like:

<link href="style.css" type="text/css" rel="stylesheet">

input{

  font-weight: 300;

  width: 100%;

  font-size: 25px;

  border: 0px;

  color: #4b545f;

  font-family: Helvetica Neue, Helvetica, Sans-serif;

  margin: 5%;

}

form{

  text-align: center;

  width: 90%;

  margin: 0 auto;

}

button{

  border: 2px solid #ddd;

  padding: 3%;

  margin: 5%;

  width: 50%;

}

<head>

  <title>Log In Page</title>

  <link href="style.css" type="text/css" rel="stylesheet">

</head>

**Position**

Positioning an element is an important part of shaping the layout of a web page.

**Instructions**

**1.**

You have two <div> elements adjacent to each other at the moment. Change the positioning of these <div>s to see what color is created when the <div>s overlap.

Hint

To change the positioning of an element, add the position property and set a value in the declaration.

Read more about position and values for position at the [MDN CSS Position documentation](https://developer.mozilla.org/en-US/docs/Web/CSS/position). Some values that would be helpful to look into are: absolute and relative.

#red {

  background-color: rgb(255, 0, 0);

  position: absolute;

}

#blue {

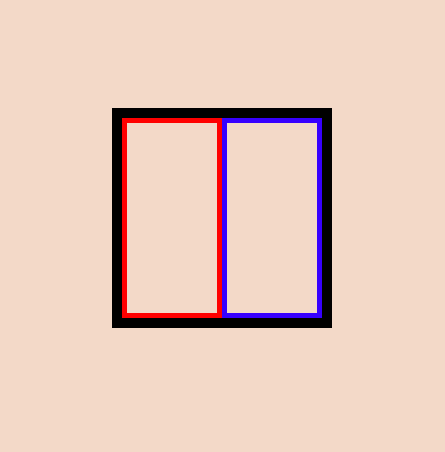
  background-color: rgb(0, 0, 255);

  position: absolute;

}

**Display**

Change the display of an element to render like the image below:



**Instructions**

**1.**

Currently the display property is the reason why the red and blue boxes are aligning next to each other. Fix this alignment by assigning the #left ruleset the correct display value.

Hint

You have to assign a display value that keeps the dimensions of the inner boxes and that also allows other elements to fit in the same line. Sounds like you can use the inline-block element to do just that!

Read more about display properties at the [MDN CSS display documentation](https://developer.mozilla.org/en-US/docs/Web/CSS/display).

#left {

  border: 5px solid red;

  width: 90px;

  height: 190px;

  display: inline-block;

}

#right {

  border: 5px solid blue;

  width: 90px;

  height: 190px;

  float: right;

}

**Clean Up a Stylesheet**

Clean up the unused CSS selectors, selectors that are not being applied to any element, in **index.html**.

**Instructions**

**1.**

Remove the 3 *unused* selectors and their styles inside **style.css** by checking to see the styles applied to the elements inside **index.html**.

Hint

The following selector would classify as an *unused* selector if the **index.html** file had no elements with an ID of description:

#description {

font-size: 12px;

}

Therefore, we could remove it from the stylesheet without affecting anything displayed inside **index.html**.

You’ll have to compare the selectors in **style.css** and the elements in **index.html** to see if there’s any effect. If you see a change in the browser after you delete a selector and run your code, then that selector has a purpose!

**Specificity**

Read more about CSS specificity in the [MDN CSS Specificity documentation.](https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity)

**Instructions**

**1.**

Add a ruleset in **style.css** to make the opacity of the paragraph, containing the text, “What will it be?”, equal to 0.5 — the only element that should change is the second <p>.

Think about the specificity of CSS rules when it pertains to tags, classes, ids, and parent elements.

Hint

Since an id selector is more specific than a tag selector, the following will convert the background color with an <h1> element with an id of hello to yellow.

h1 {

background-color: blue;

}

#hello {

background-color: yellow;

}

**2.**

In **style.css**, change the color of the <h1> to #12130F.

Hint

The following will convert the background color of an <h1> element that is the child of a div to yellow.

div {

background-color: blue;

}

div > h1 {

background-color: yellow;

}

#question{

   opacity: .5;

}

h1 {

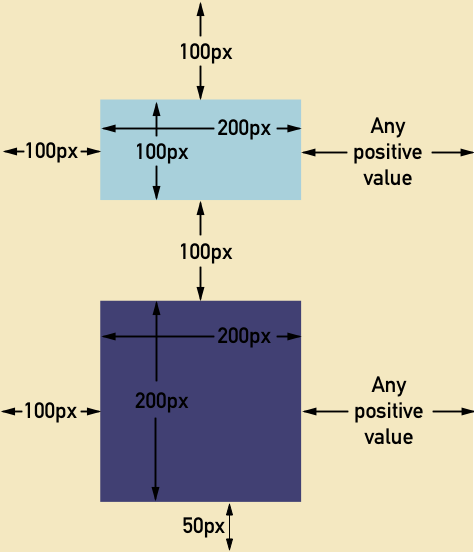
  font-size: 50px;

  font-weight: bold;

  color: #12130F;

}

**Padding vs. Margin**

A padding vs. margin challenge! Take the initial teal and purple boxes that have a height and width of 100px and manipulate the padding and margin properties to get the image below:

**Instructions**

**1.**

Changing **only** the padding and margin properties of #teal and #purple, make the boxes in the browser match the dimensions of the boxes in the image above.

Remember the initial boxes have dimensions of 100px height and width.

Hint

The padding property adds space inside the element. The margin property adds space around the element. Each of these two properties has top, left, bottom and right sub-properties.

#teal {

  background-color: #A8D0DB;

  padding: 0 50px;

  margin: 100px;

}

#purple {

  background-color: #414073;

  padding: 50px;

  margin: 50px 100px;

}

# CSS Code Challenges Review

Great job on completing the CSS code challenges!

To recap on what you just practiced:

* Edited an image to make it round.
* Targeted specific elements’ CSS.
* Transferred inline CSS to a stylesheet.
* Refactored a CSS stylesheet to remove extra selectors.
* Manipulated padding and margin to align elements.
* Added a fallback font just in case a font can’t be displayed.
* Separated declarations using semicolons.
* Changed the display property of an element.
* Assigned the position of elements to overlap.

**Instructions**

If you want to challenge yourself, go back to the challenges and find other solutions to the CSS problems!