5.1 Flexbox

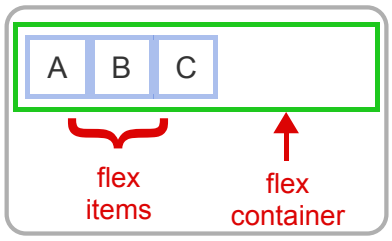
Flexbox container and items

The ***Flexible Box*** or ***flexbox*** is a CSS layout mode that provides an efficient way to lay out elements in a container so the elements behave predictably when the container is resized or viewed on different screen sizes.

A ***flex container*** is an element that has the CSS property `display` set to `flex` to create a block-level flex container or `inline-flex` to create an inline flex container. Ex: `<div style="display: flex">`. Flex containers hold flex items. A ***flex item*** is a child element of a flex container that is positioned and sized according to various CSS flexbox properties.

PARTICIPATION ACTIVITY 5.1.1: Flexbox example renders three div elements on the same row.

```
<div id="container">
   <div>A</div>
   <div>B</div>
   <div>C</div>
</div>
```




```
/* flex container */
#container {
   display: flex;
   border: 1px green solid;
   padding: 5px;
}

/* flex items */
#container > div {
   padding: 10px;
   border: 1px blue solid;
}
```

Animation content:

Two blocks of code are displayed, the first in HTML:

  A
  B

```
  <div>C</div>
</div>,
 the second in CSS:
 /* flex container */
#container {
  display: flex;
  border: 1px green solid;
  padding: 5px;
}
/* flex items */
#container > div {
  padding: 10px;
  border: 1px blue solid;
}
```

A screen is displayed below. The first scene shows A, B, and C vertically aligned on the left without CSS. Next the flex container block is highlighted and A, B, and C display left-aligned in a row. Lastly the flex items block is highlighted and A, B, and C are formatted.

Animation captions:

1. Without any CSS, the A, B, and C div elements display vertically, each filling the browser width.
2. Setting the CSS display property to "flex" makes the outer div the flex container. The flex items now display on the same row.
3. The flex items have padding and blue borders.

PARTICIPATION ACTIVITY 5.1.2: Flexbox container and items.

Refer to the animation above.

1) The flex container has _____ flex item(s).

- ○ 1
- ○ 2
- ○ 3

2) Removing _____ from the `#container` rule displays all flex items vertically on separate rows.

- `display: flex;`
- `border: 1px green solid;`
- `padding: 5px;`

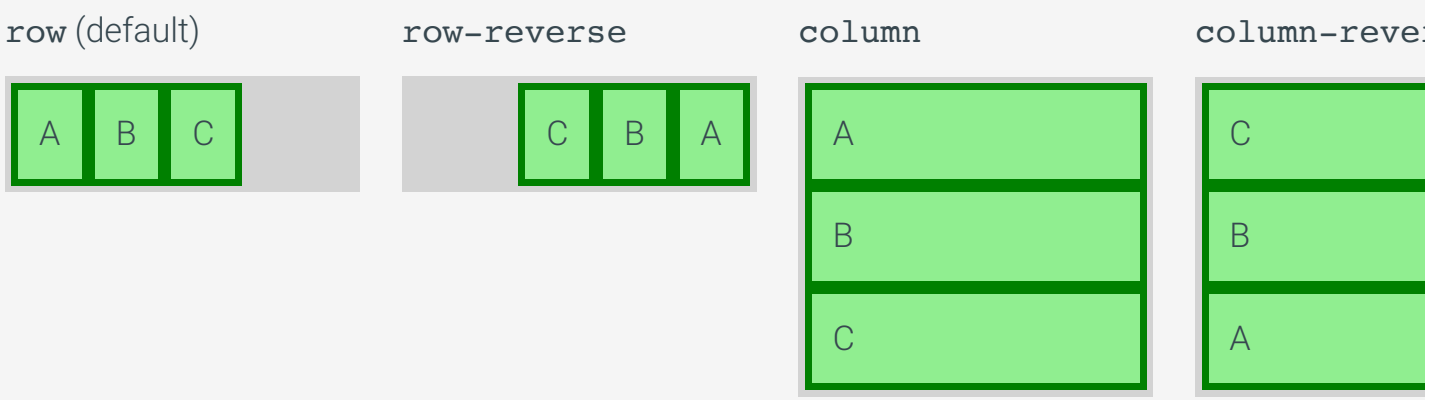
3) Flex items appear _____ within a flex container by default.

- left aligned
- centered
- right-aligned

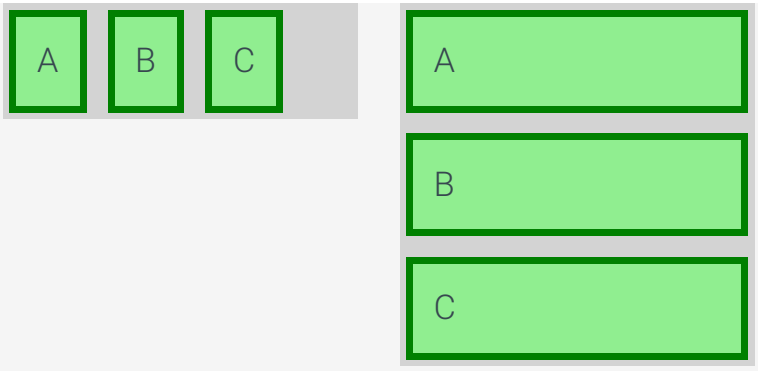
Flex container properties

Several CSS properties modify the default behavior of a flex container:

- The ***flex-direction*** property defines the direction of flex items within the container using values:

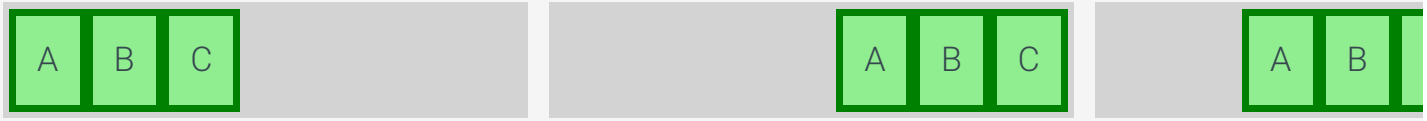


- The ***gap*** property defines the space between flex items. Ex: `gap: 10px;` puts a 10px gap between all items.

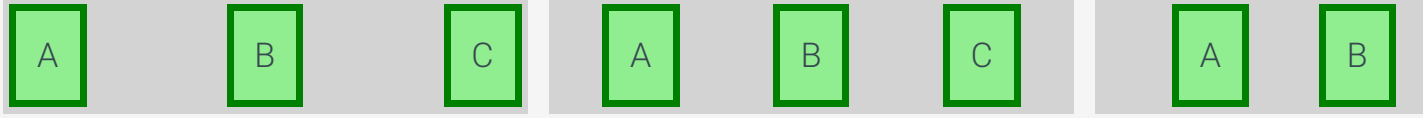


- The ***justify-content*** property justifies the flex items within the container using values:

`flex-start` (default) `flex-end` `center`

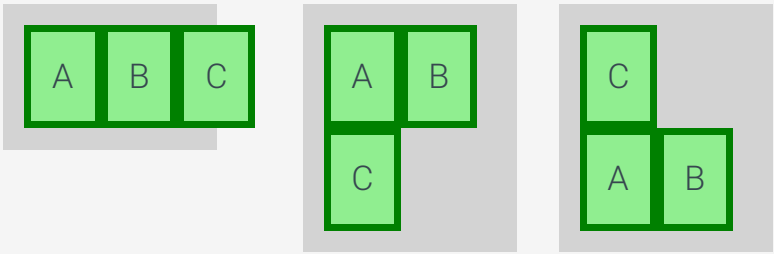


`space-between` `space-around` `space-evenly`



- The ***flex-wrap*** property determines if or how flex items wrap onto multiple rows when the container is not wide enough to hold all items, using values:

`nowrap` (default) `wrap` `wrap-reverse`



PARTICIPATION ACTIVITY 5.1.3: Flex container properties for photos.

The webpage below uses a flexbox to display three photos with captions.

Make the following CSS modifications to the flex container. After each modification, render the webpage to verify the modification works properly.

1. Add a `gap` property to the flex container to put a 20px gap between the photos.

2. Add a `flex-direction` property to the flex container to display the photos in reverse.
3. Add a `justify-content` property to the flex container to display the photos centered in the browser window.
4. Add a `flex-wrap` property to the flex container so the photos wrap to the next row when the browser width is reduced.

Note: The window that renders the webpage can be resized using the small handle in the bottom-right corner.

HTML CSS

```
<div id="container">
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/ducatiSm
      <p>
         Ducati motorcycle
      </p>
   </div>
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/kyotoRes
      <p>
         Japanese restaurant
      </p>
   </div>
   <div class="photo">
      <img src="https://resources.zybooks.com/WebProgramming/lake1v1.
      <p>
```

Render webpage Reset code

Your webpage

Ducati motorcycle

Japanese restaurant

Lake in Colorado

▶ View solution

PARTICIPATION ACTIVITY 5.1.4: Flex container properties.

Refer to the HTML below. Assume a CSS rule exists that makes the `<div>` a flex container.

```
<div id="container">
   <img src="madrid.jpg" alt="Madrid">
   <img src="hanoi.jpg" alt="Hanoi">
   <img src="islamabad.jpg" alt="Islamabad">
</div>
```

1) What `flex-direction` value displays Madrid on the right side of the grid container with all three images on the same row?

- `row`
- `row-reverse`
- `column-reverse`

2) What `justify-content` value puts an equal amount of space on either side of the three images but leaves no space between the images?

- `center`
- `space-between`
- `space-around`

3) What `flex-wrap` value makes the Islamabad image appear below Madrid if the grid container is not wide enough to show all three images on the same row?

- `nowrap`
- `wrap`
- `wrap-reverse`

Flex item properties

A flex item's width is determined by the combination of three CSS properties:

- The ***flex-basis*** property sets the initial length of a flex item. The values can be `auto` (the default), a percentage, or a length unit. The default value `auto` makes the flex item the same initial length as the content.
- The ***flex-grow*** property sets a proportion that determines how much of the available container space should be assigned to the item. The default is 0, meaning the size should be based on the item's content.
- The ***flex-shrink*** property sets a proportion that determines the item's minimum size. The default is 1, meaning the size should shrink at the same rate as other items when the container width shrinks. A value of 0 means the item should not change sizes when the container width shrinks.

PARTICIPATION ACTIVITY 5.1.5: Changing flex item properties.

```
<nav>
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="products.html">Products</a></li>
    <li><a href="about.html">About</a></li>
  </ul>
</nav>
```

```
nav ul {
  display: flex;
  list-style-type: none;
  padding: 0;
}
```

```
nav li {
  flex-grow: 1;
  background-color: gold;
  text-align: center;
}
```

```
nav li {
  flex-basis: 100px;
  flex-shrink: 0;
  background-color: gold;
  text-align: center;
}
```

Animation content:

A code block of HTML is displayed:
<nav>

  Home
  Products
  About

</nav>
Next three code blocks of CSS are shown. The first block shows the following:
nav ul {
 display: flex;
 list-style-type: none;
 padding: 0;
}
The second:
nav li {
 flex-grow: 1;
 background-color: gold;

```
  text-align: center;
}
```

And the third:

```
nav li {
  flex-basis: 100px;
  flex-shrink: 0;
  background-color: gold;
  text-align: center;
}
```

Below each CSS code block there is a screen displaying the output. The first shows Home, Product, and About in a row across the top. The second first shows the three nav links left-aligned in a row when flex-grow is 0, then portions the nav links evenly when flex-grow is 1. The third box shows the three nav links changing container size when the browser is resized when flex-shrink is 0, then flex-shrink is changed to 1, preventing the containers from resizing.

Animation captions:

1. A website's navigation links are displayed in an unordered list.
2. Making the ul element a flex container places the nav links on the same row.
3. By default, the li elements have flex-basis:auto and flex-grow:0, so li elements are only as wide as the item's content.
4. Changing flex-grow from the default 0 to 1 gives all li elements the same proportion. The elements fill the flex container.
5. Replacing "flex-grow:1" with "flex-basis:100px" makes each li element 100px wide.
6. Resizing the browser changes the container size. When the container shrinks, the li elements shrink to fill the available space.
7. Changing flex-shrink from the default 1 to 0 prevents the li elements from shrinking when the browser is resized.

PARTICIPATION ACTIVITY 5.1.6: Flex item properties.

Refer to the webpage below.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>ACME Widgets</title>
   <style>
      body {
```

```
            font-family: Arial;
        }

        header {
            display: flex;
            justify-content: space-between;
        }

        header h1 {
            margin: 6px;
        }

        nav ul {
            display: flex;
            list-style-type: none;
            padding: 0;
            margin: 0;
            width: 500px;
        }

        nav li {
            background-color: gold;
            margin: 5px;
            padding: 10px;
            text-align: center;
        }

        nav a {
            color: black;
            text-decoration: none;
        }
    </style>
</head>
<body>
    <header>
        <h1>ACME Widgets</h1>
        <nav>
            <ul>
                <li><a href="index.html">Home</a></li>
                <li><a href="products.html">Products</a></li>
                <li><a href="services.html">Services</a></li>
                <li><a href="about.html">About</a></li>
            </ul>
        </nav>
    </header>
</body>
</html>
```

1) How many flex containers exist in the webpage?

- 0
- 1
- 2

2) The heading "ACME Widgets" appears _____ the navigation links in the browser.

- to the left of
- to the right of
- above

3) What is the `flex-basis` for the `<nav>` flex item?

- `auto`
- 0
- 1

4) How wide is the `<nav>` flex item?

- Just wide enough to hold all the navigation links
- Half the width of the webpage
- Entire width of the webpage

5) In addition to the CSS below, _____ must be added to the `nav li` rule so the li elements fill half the row.

```
nav {
   flex-basis: 50%;
}
```

- `flex-grow: 0;`
- `flex-grow: 1;`
- `flex-grow: 50%;`

6) Adding _____ to the `header h1` rule prevents the heading from shrinking when the browser is resized.

- `flex-shrink: none;`
- `flex-shrink: 1;`
- `flex-shrink: 0;`

The flex property

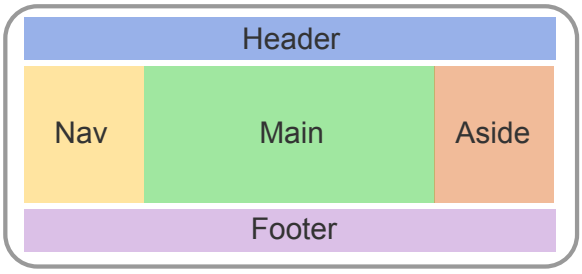
The shorthand property ***flex*** specifies `flex-grow`, `flex-shrink`, and `flex-basis` together. Ex: `flex: 0 1 auto;` is the same as `flex-grow: 0; flex-shrink: 1; flex-basis: auto;`.

PARTICIPATION ACTIVITY 5.1.7: Flexbox layout using the flex property.

```
<body>
   <header>Header</header>

   <!-- Flexbox layout -->
   <div id="container">
      <nav>Nav</nav>
      <main>Main</main>
      <aside>Aside</aside>
   </div>

   <footer>Footer</footer>
</body>
```




```
#container {
   display: flex;
}

nav {
   flex: 0 1 20%;
}

main {
   flex: 0 1 60%;
}

aside {
   flex: 0 1 20%;
}
```

flex-grow flex-shrink flex-basis

Animation content:

Two blocks of code are shown. The HTML:

```
<body>
  <header>Header</header>
  <!-- Flexbox layout -->
  <div id="container">
    <nav>Nav</nav>
    <main>Main</main>
    <aside>Aside</aside>
  </div>
  <footer>Footer</footer>
</body>
```

and the CSS:

```
#container {
  display: flex;
}
nav {
  flex: 0 1 20%;
}
main {
  flex: 0 1 60%;
}
aside {
  flex: 0 1 20%;
}
```

In the CSS flex property, the 0 controls flex-grow, the 1 controls flex-shrink, and the percentage controls flex-basis. Below is a screen displaying header spanning the width at the top of body, footer spanning the width at the bottom of body, and between the two is <nav> occupying 20% of the left side of body, main occupying 60% of the middle of body, and aside occupying the remaining 20% of body on the right side.

Animation captions:

1. <header> and <footer> span the entire width of <body>, but the is a flex container that displays the flex items on the same row.
2. <nav>, <main>, and <aside> all have flex-grow = 0, so all three flex items' width should be based on each item's content.
3. <nav>, <main>, and <aside> all have flex-shrink = 1, so all three flex items shrink at the same rate when the browser is resized.
4. <nav> occupies 20% of the row, <main> occupies 60%, and <aside> occupies 20%. 20% +

60% + 20% = 100% of the row.

PARTICIPATION ACTIVITY 5.1.8: Flex item properties.

Refer to the animation above.

1) Which property makes the Main flex item wider than Nav and Aside?

- `flex-grow`
- `flex-basis`
- `flex-shrink`

2) How do the changes below affect the webpage?

```
nav   { flex: 1   1 auto; }
main  { flex: 3.5 1 auto; }
aside { flex: 1   1 auto; }
```

- All three flex items are equal width.
- Nav and Aside are wider than Main.
- The layout will be nearly the same as before.

3) How do the changes below affect the webpage?

```
#container { display: flex;
             flex-direction:
row;
             justify-content:
center; }
nav   { flex: 0 1 auto; }
main  { flex: 0 1 auto; }
aside { flex: 0 1 auto; }
```

- All three flex items are default width and centered horizontally in the flex container.
- Nav and aside are wider than Main.
- All three flex items will be default width, aligned to the left.

4) How does the change below affect the webpage?

```
#container { display: flex;
flex-direction: column; }
```

- No changes.
- All three flex items are stacked on top of one another.
- The direction of the flex items is reversed.

PARTICIPATION ACTIVITY 5.1.9: Flexbox practice.

The webpage displays three years of Grammy Award nominations for Album of the Year. Each year's nomination is placed in a `<section>` element.

Alter the webpage to display the three sections in a single row:

1. Add a single `<div>` element that surrounds all three sections.
2. Add the proper CSS so the `<div>` becomes a flex container and displays each section on a single row.
3. Add a `flex` property to the `section` rule that sets flex-grow to 0, flex-shrink to 1, and flex-basis to 250px.
4. Add a `flex-wrap` property that makes the sections wrap to the next row when the browser is resized.

After adding the HTML and CSS above, render the webpage. Resize the rendered webpage's window and verify that the sections wrap to the next row when the window isn't wide enough to display the sections side-by-side.

HTML CSS

```
<h1>Grammy Nominees for Album of the Year</h1>

<section>
   <h2>2020</h2>
   <ul>
      <li><cite>Folklore</cite> by Taylor Swift</li>
      <li><cite>Future Nostalgia</cite> by Dua Lipa</li>
      <li><cite>Hollywood's Bleeding</cite> by Post Malone</li>
      <li><cite>Chilombo</cite> by Jhené Aiko</li>
      <li><cite>Black Pumas (Deluxe Edition)</cite> by Black Pumas</l
      <li><cite>Everyday Life</cite> by Coldplay</li>
      <li><cite>Djesse Vol.3</cite> by Jacob Collier</li>
      <li><cite>Women In Music Pt. III</cite> by HAIM</li>
   </ul>
</section>
<section>
```

Render webpage Reset code

Your webpage

Grammy Nominees for Album of the Year

2020

- *Folklore* by Taylor Swift
- *Future Nostalgia* by Dua Lipa
- *Hollywood's Bleeding* by Post Malone
- *Chilombo* by Jhené Aiko
- *Black Pumas (Deluxe Edition)* by Black Pumas
- *Everyday Life* by Coldplay
- *Djesse Vol.3* by Jacob Collier
- *Women In Music Pt. III* by HAIM

2019

▶ View solution

CHALLENGE ACTIVITY 5.1.1: Flexbox.

550544.4142762.qx3zqy7

Start

For the element with class container, set display to flex and width to 95px.

SHOW EXPECTED

CSS HTML

```
1  .container {
2
3     /* Your solution goes here */
4
5  }
6  .container > div {
7     background-color: lightblue;
8     border: 2px solid blue;
9     padding: 6px;
10 }
```

1 2 3

Check Next

View your last submission

Exploring further:

- HTML Layouts from W3Schools
- CSS Flexbox from W3Schools

5.2 Grid layout

Grid container and grid items

Grid layout is a CSS layout mode that divides a webpage into a rectangular grid in which to position page elements. Grid layout is ideal for designing two-dimensional webpage layouts.

A ***grid container*** is an element that has the CSS property `display` set to `grid` to create a block-level grid container or `inline-grid` to create an inline grid container. Ex: `<div style="display: grid">`. A ***grid item*** is a child element of a grid container that is by default placed into a single grid cell.

The ***grid-template-columns*** property defines the grid container's number of columns and optionally the width of each column. Ex: `grid-template-columns: 50px 90px auto auto;` specifies 4 values that create 4 columns: the first is 50px wide, the second is 90px wide, and the third and fourth columns are automatically sized to fit the remainder of the grid width.

PARTICIPATION ACTIVITY 5.2.1: Grid layout example.

```
<div id="grid-container">
   <div>1</div>
   <div>2</div>
   <div>3</div>
   <div>4</div>
   <div>5</div>
   <div>6</div>
</div>
```

```
#grid-container {
   display: grid;
   grid-template-columns: auto auto auto;
}

#grid-container > div {
   text-align: center;
   background: lightgreen;
   border: 3px solid green;
   padding: 20px;
}
```

1	2	3
4	5	6

Animation content:

Two blocks of code are shown. The HTML:

```
<div id="grid-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

and the CSS:

```
 #grid-container {
  display: grid;
  grid-template-columns: auto auto auto;
}
#grid-container > div {
  text-align: center;
  background: lightgreen;
  border: 3px solid green;
  padding: 20px;
}
```

The CSS grid-template-columns property determines how many columns will be displayed based on how many property values there are. A screen is shown divided initially into 3 columns based on grid-template-columns, then the 6 div elements are displayed as 3 columns and 2 rows.

Animation captions:

1. The "display: grid;" declaration makes the with id grid-container a block-level grid.
2. The grid-template-columns property is assigned 3 "auto" values, so the grid container will contain 3 equally-sized columns.
3. Six elements are children of the grid container, so each element becomes a grid item. The div child selector puts a green border around each grid item.

PARTICIPATION ACTIVITY 5.2.2: Grid layout basics.

Refer to the animation above.

1) If `#grid-container` is modified, how many columns will the grid have?

```
#grid-container {
   display: grid;
   grid-template-columns: auto
auto;
}
```

- 2
- 3
- 4

2) If `#grid-container` is modified, how wide is the second column?

```
#grid-container {
   width: 600px;
   display: grid;
   grid-template-columns:
200px auto;
}
```

- 200px
- 400px
- 600px

3) If `#grid-container` is modified, how wide is the second column?

```
#grid-container {
   width: 600px;
   display: grid;
   grid-template-columns: 20%
auto;
}
```

- 120px
- 480px
- 600px

4) If the grid container has 10 grid items, what `grid-template-columns` value creates a grid with 2 rows and 5 columns?

```
#grid-container {
   width: 600px;
   display: grid;
   grid-template-columns:
_____;
}
```

- 5
- 200px auto
- 200px auto auto auto 200px

PARTICIPATION ACTIVITY 5.2.3: Arrange the photos with a grid layout.

Edit the CSS so the outer div becomes a grid layout container that displays the photos and captions in two columns.

HTML CSS

```
<body>
   <h1>Photo Album</h1>
   <div id="grid-container">
      <figure>
         <img src="https://resources.zybooks.com/WebProgramming/ducat
         <figcaption>One way to scare Mom</figcaption>
      </figure>
      <figure>
         <img src="https://resources.zybooks.com/WebProgramming/small
         <figcaption>Say hello to Daisy</figcaption>
      </figure>
      <figure>
         <img src="https://resources.zybooks.com/WebProgramming/lake1
         <figcaption>Last summer we hiked to a beautiful lake</figcap
      </figure>
      <figure>
```

Render webpage Reset code

Your webpage

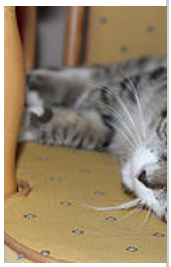
Photo Album

One way to scare Mom

Expected webpage

Photo Album

One way to scare Mom



Say hello to

▶ View solution

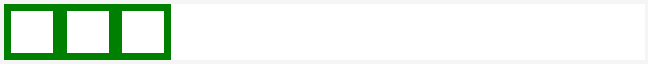
Controlling the grid container

The default behavior of a grid container can be modified with various CSS properties:

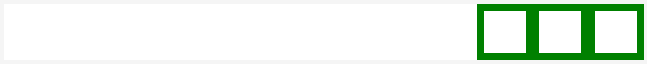
- The ***gap*** property defines the gap between each grid row and column. Ex: `gap: 5px 25px;` puts a 5px gap between each row and a 25px gap between each column.

- The ***grid-template-rows*** property defines the height of each row. Ex: `grid-template-rows: 20px 40px;` makes the first row 20px tall and the second row 40px tall.

- The ***justify-content*** property horizontally aligns the grid items inside the grid container using values:
  - `start` - Aligns grid flush with the grid container's starting edge.



  - `end` - Aligns grid flush with the grid container's ending edge.



  - `center` - Aligns grid in the center of the grid container.

  - `stretch` - Stretches the grid items to fill the grid container width.

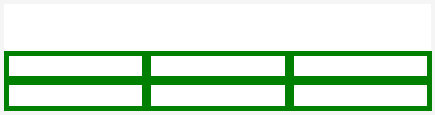
  - `space-around` - Places equal spacing between grid items with half the space on either side of the grid container.

  - `space-between` - Places equal spacing between grid items with no space on either side of the grid container.

- `space-evenly` - Places equal spacing between grid items, including the sides of the grid container.

- The ***align-content*** property vertically aligns the grid items inside the grid container using values:
  - `start` - Aligns grid flush with the grid container's starting edge.

  - `end` - Aligns grid flush with the grid container's ending edge.



  - `center` - Aligns grid in the center of the grid container.

  - `stretch` - Stretches the grid items to fill the grid container height.

  - `space-around` - Places equal spacing between grid items with half the space on either side of the grid container.

  - `space-between` - Places equal spacing between grid items with no space on either side of the grid container.

  - `space-evenly` - Places equal spacing between grid items, including the sides of the

grid container.

The `justify-content` and `align-content` properties have no effect unless the grid width or height is less than the grid container's width or height.

PARTICIPATION ACTIVITY 5.2.4: Modify the grid container.

Refer to the HTML and CSS below:

```
<style>
#grid-container {
   width: 300px;
   height: 70px;
   background: mediumseagreen;
   display: grid;
   grid-template-columns: auto auto auto;
}

#grid-container > div {
   background: lightgreen;
}
</style>

<body>
   <div id="grid-container">
      <div>A</div>
      <div>B</div>
      <div>C</div>
      <div>D</div>
      <div>E</div>
      <div>F</div>
   </div>
</body>
```

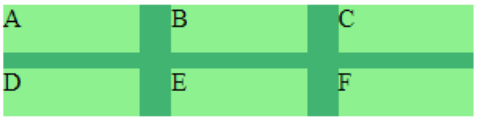
1) Add a CSS declaration to `#grid-container` so the first row is 20 pixels tall and the second row is 30 pixels tall.

:

```
20px 30px;
```

Check Show answer

2) Put a 10 pixel gap between each row and 20 pixel gap between each column.

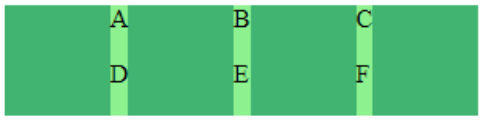


gap:

;

Check Show answer

3) Place an equal amount of horizontal space between each grid item, including the grid edges.



justify-content:

;

Check Show answer

4) Center each grid item vertically in the grid.

A B C
D E F

:

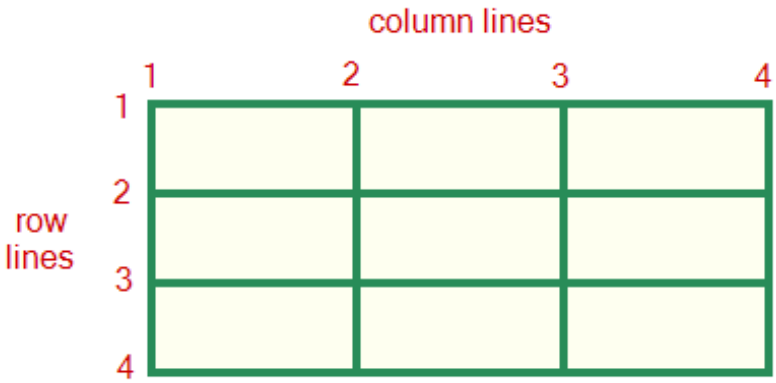
```
center;
```

Check Show answer

Controlling grid item placement

A grid item by default appears in a single row and column based on the ordering of the grid item within the grid container. However, grid items may be positioned at specific grid locations using the column line and row line numbers as illustrated in the figure below.

Figure 5.2.1: Row and column lines.



A grid item may be placed in a specific row or column or span multiple rows and/or columns using various following CSS properties:

- The ***grid-row*** property lists the grid item's starting and ending row line numbers. Ex: `grid-row: 1 / 3;` makes the grid item start at row line 1 and end at row line 3, so the grid item spans 2 rows.
- The ***grid-column*** property lists the grid item's starting and ending column line numbers. Ex: `grid-column: 1 / 4;` makes the grid item start at column line 1 and end at column line 4, so the grid item spans 3 columns.

- The ***grid-area*** property lists the grid item's starting and ending row and column numbers. Ex: `grid-area: 1 / 2 / 3 / 4;` makes the grid item start at row line 1 and column line 2 and end at row line 3 and column line 4, so the grid item spans 2 rows and 2 columns.

PARTICIPATION ACTIVITY 5.2.5: Rearrange the grid.

Add the following CSS modifications to alter the grid:

1. Add the following CSS to move the A item to the 1st column and 3rd row:

```
#grid-item-a {
   grid-row: 3 / 4;
}
```

   Render the webpage and observe the A item is previously where the G item was located.

2. Add the following CSS to make the B item occupy the first 3 rows of the 3rd column:

```
#grid-item-b {
   grid-column: 3 / 4;
   grid-row: 1 / 4;
}
```

   Render the webpage and observe that B is on the right edge of the grid, and a 4th row is created with an empty grid cell below B.

3. Add the following CSS to make the C item occupy the first 3 rows and 2 columns:

```
#grid-item-c {
   grid-area: 1 / 1 / 4 / 3;
}
```

   Render the webpage and observe the webpage matches the expected webpage.

HTML CSS

```
<body>
   <div id="grid-container">
      <div id="grid-item-a">A</div>
      <div id="grid-item-b">B</div>
      <div id="grid-item-c">C</div>
      <div>D</div>
      <div>E</div>
      <div>F</div>
      <div>G</div>
      <div>H</div>
      <div>I</div>
   </div>
</body>
```

Render webpage Reset code

Your webpage

A B C

D E F

G H I

Expected webpage

C B

F G H

▶ View solution

PARTICIPATION ACTIVITY 5.2.6: Modify the grid layout.

Refer to the HTML and CSS below:

```
<style>
#grid-container {
   width: 300px;
   background: mediumseagreen;
   display: grid;
   grid-template-columns: auto auto auto;
   gap: 5px 5px;
}

#grid-container > div {
   background: lightgreen;
}
</style>

<body>
   <div id="grid-container">
      <div id="grid-item-a">A</div>
      <div id="grid-item-b">B</div>
      <div id="grid-item-c">C</div>
      <div>D</div>
      <div>E</div>
      <div>F</div>
   </div>
</body>
```

1) Make the A item occupy the entire top row.


```
#grid-item-a {
   grid-column:
                ;
}
```

Check Show answer

2) Make the C item occupy the top 2 rows in the 2nd column.


```
#grid-item-c {
   grid-column: 2 / 3;
   grid-row:
                          ;
}
```

Check Show answer

3) Make the B item occupy 2 rows and 2 columns in the bottom-right grid corner.

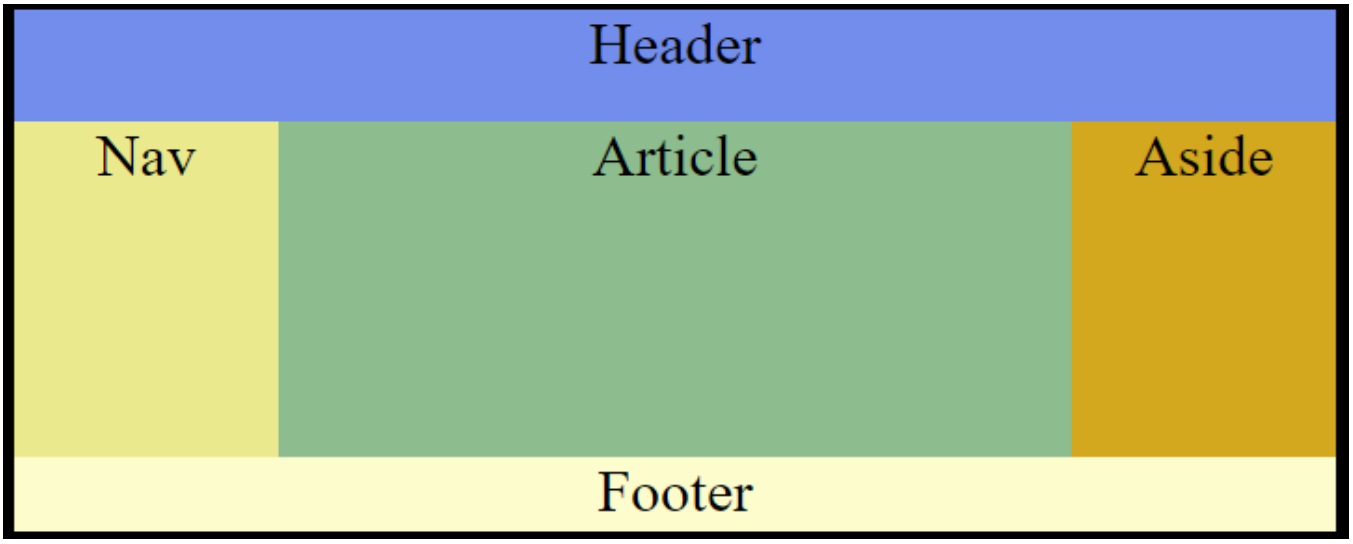

```
#grid-item-b {
   grid-area:
                          ;
}
```

Check Show answer

Naming grid items

Grid items may be assigned names with the `grid-area` property. The grid container's ***grid-template-areas*** property specifies the grid layout using the named grid items.

Figure 5.2.2: Layout goal for the Participation Activity below.



PARTICIPATION ACTIVITY 5.2.7: Grid layout using named grid items.

Add the following CSS modifications to create a grid layout for the webpage that matches the figure above:

1. Add the following CSS to the `body` selector to change the `<body>` into a grid container with 3 columns and 3 rows:

```
body {
   ...
   display: grid;
   grid-template-columns: 150px auto 150px;
   grid-template-rows: 60px 180px 40px;
}
```

   Render the webpage and observe that the five grid items occupy 2 rows.

2. Use the `grid-area` property to name each of the 5 grid items:

```
header {
   background: cornflowerblue;
   grid-area: head;
}

nav {
   background: khaki;
   grid-area: nav;
}

article {
   background: darkseagreen;
   grid-area: article;
}

aside {
   background: goldenrod;
   grid-area: aside;
}

footer {
   background: lemonchiffon;
   grid-area: foot;
}
```

3. Use the `grid-template-areas` property on the grid container to layout the grid items. Each row is specified in a single string with the grid item names.

```
body {
   ...
   grid-template-areas:
      "head head head"
      "nav article aside"
      "foot foot foot";
}
```

Render the webpage, which should look like the figure above. The Header occupies all 3 columns on the top row; the Nav, Article, and Aside occupy the 3 columns in the second row; and the Footer occupies all 3 columns on the bottom row.

HTML CSS

```
<body>
    <header>Header</header>
    <nav>Nav</nav>
    <article>Article</article>
    <aside>Aside</aside>
    <footer>Footer</footer>
</body>
```

Render webpage Reset code

Your webpage

Header
Nav
Article
Aside
Footer

▶ View solution

PARTICIPATION ACTIVITY 5.2.8: Named grid items.

Refer to the Participation Activity above.

1) What `grid-template-areas` value makes the Article span 2 columns?

-
```
"head head head"
"nav article aside"
"foot article foot"
```

-
```
"head article head"
"nav article article
aside"
"foot foot foot"
```

-
```
"head head head"
"nav article article"
"foot foot foot"
```

2) A single period can be used in a `grid-template-areas` value to represent an empty grid cell. Which `grid-template-areas` value places an empty grid cell under Nav?

-
```
". head head"
"nav article aside"
"foot foot foot"
```

-
```
"head head head"
"nav . aside"
"foot foot foot"
```

-
```
"head head head"
"nav article aside"
". foot foot"
```

3) What is the correct way to name the new grid item `<section>`?

```
section {
   grid-area: ____;
}
```

- `"mysection"`
- `mysection`
- `'mysection'`

CHALLENGE ACTIVITY 5.2.1: Grid layout.

550544.4142762.qx3zqy7

Start

For the element with class container, set the display and grid-template-columns properties to display a grid with 2 columns that are both automatically sized to fit the grid width. **SHOW EXPECTED**

CSS HTML

```
1  .container {
2
3     /* Your solution goes here */
4
5  }
6  .container > div {
7     background-color: lightblue;
8     border: 2px solid blue;
9     padding: 6px;
10 }
```

1 2 3 4

Check Next

View your last submission

Exploring further:

- CSS Grid Layout Module from W3Schools
- CSS Grid Layout Browser Support from caniuse.com

5.3 Positioning elements

The position property

The CSS ***position*** property gives developers more control over where elements should appear in the browser. `position` has four possible values:

- `static` - ***Static positioning*** is the default positioning
- `relative` - ***Relative positioning*** positions the element relative to the element's default position
- `fixed` - ***Fixed positioning*** positions the element relative to the viewport in a fixed location
- `absolute` - ***Absolute positioning*** positions the element relative to the nearest positioned ancestor

PARTICIPATION ACTIVITY 5.3.1: Relative positioning.

```
#content {
  border: solid 2px blue;
  color: blue;
  position: relative;
  left: 20px;
  top: 10px;
}
```


```
<div>Before</div>
<div id="content">Content</div>
<div>After</div>
```

Animation content:

Two blocks of code are shown. The CSS:
#content {

```
  border: solid 2px blue;
  color: blue;
  position: relative;
  left: 20px;
  top: 10px;
}
```

and the HTML:

```
<div>Before</div>
<div id="content">Content</div>
<div>After</div>
 }
```

A screen is displayed with the Content div. When the property position is added to the CSS the div element's position becomes relative to its normal position and changes based on the properties left and top. A negative value in the property left makes the div move to the left and a positive value moves div to the right. A negative value in the property top makes the div move down and a positive value moves div up.

Animation captions:

1. The "Content" displays in the default location.
2. Adding relative positioning to #content does not change the "Content" position until "left" and/or "top" properties are specified.
3. "left: -20px" moves the left edge 20 pixels left from the default location.
4. "left: 20px" moves the left edge 20 pixels to the right of the default location.
5. Negative values for "top" move the element up, and positive values move the element down.

PARTICIPATION ACTIVITY 5.3.2: Relative and static positioning.

1) Where is the image located relative to the image's default location?

```
<img src="world.jpg"
style="position:relative;
left:30px">
```

- 30 pixels to the right
- 30 pixels to the left
- No change

2) Where is the image located relative to the image's default location?

```
<img src="world.jpg"
style="position:relative;
top:-30px">
```

- 30 pixels higher
- 30 pixels lower
- No change

3) Where is the image located relative to the image's default location?

```
<img src="world.jpg"
style="position:static;
left:-20px; top:30px">
```

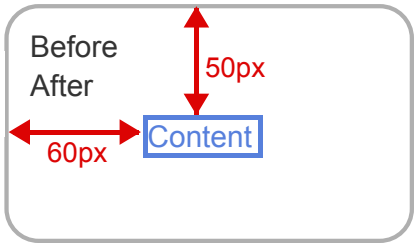
- 20 pixels to the right and 30 pixels higher
- 20 pixels to the left and 30 pixels lower
- No change

Fixed positioning

Fixed positioning places the element at a fixed location in the viewport, and scrolling does not move the element. A ***viewport*** is the visible area of a webpage. The fixed element is detached from the normal flow of elements in the page and is layered on top of the page contents.

PARTICIPATION ACTIVITY 5.3.3: Fixed positioning.

```
#content {
  border: solid 2px blue;
  color: blue;
  position: fixed;
  left: 60px;
  top: 50px;
}
```




```
<div>Before</div>
<div id="content">Content</div>
<div>After</div>
```

Animation content:

Two blocks of code are shown. The CSS:
#content {
border: solid 2px blue;
color: blue;
position: fixed;
left: 60px;
top: 50px;
}
and the HTML:
Before
&ly;div id="content">Content
After
}
A screen is displayed with the Content . When the property position is added to the CSS the element's position becomes relative to the browser window. Using the properties left and top is positioned on the screen. A negative value in the property left makes the move to the left and a positive value moves to the right. A negative value in the property top makes the move down and a positive value moves up.

Animation captions:

1. The "Content" displays in the default location.
2. Adding fixed positioning to #content detaches the "Content" so the is layered on top of the underlying content.
3. "left: 60px" moves the 's left edge 60 pixels to the right of the browser's left edge.
4. "top: 50px" moves the 's top edge 50 pixels below the browser's top edge.

PARTICIPATION ACTIVITY 5.3.4: Fixed positioning.

Refer to the CSS below.

```
.special {
  position: fixed;
  left: 100px;
  top: 25px;
}
```

1) All elements using the "special" class are displayed 100 pixels from the browser's left edge and 25 pixels from the browser's top edge.

   - True
   - False

2) All elements using the "special" class scroll with the page contents.

   - True
   - False

3) The text "123" is displayed on top of "ABC".

```
<span
class="special">ABC</span>
<span
class="special">123</span>
```

   - True
   - False

Absolute positioning

Absolute positioning is similar to fixed positioning except:

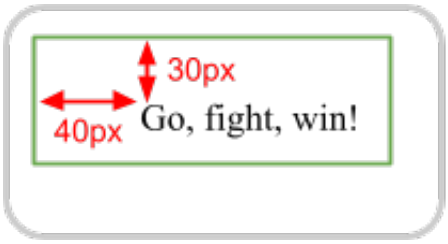
1. The position is based on the nearest positioned ancestor element that uses fixed, absolute, or relative positioning. If no positioned ancestor element exists, the element is positioned relative to the document body.
2. An absolute-positioned element scrolls with the document unless an ancestor element is using fixed positioning.

Figure 5.3.1: Cheer is absolute positioned inside a positioned ancestor (left) and relative to the document body (right).

```
#container {
  border: solid 2px green;
  position: relative;
  height: 60px;
  width: 150px;
}

#cheer {
  position: absolute;
  left: 40px;
  top: 30px;
}
```

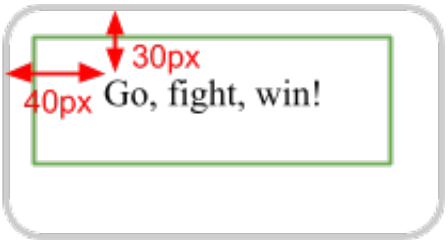
```
<div id="container">
  <div id="cheer">Go, fight,
win!</div>
</div>
```




```
#container {
  border: solid 2px green;
  /* No positioning */
  height: 60px;
  width: 150px;
}

#cheer {
  position: absolute;
  left: 40px;
  top: 30px;
}
```

```
<div id="container">
  <div id="cheer">Go, fight,
win!</div>
</div>
```



PARTICIPATION ACTIVITY 5.3.5: Absolute positioning.

Refer to the CSS below.

```
.special {
  position: absolute;
  left: 100px;
  top: 25px;
}
```

1) The is displayed 100 pixels from the browser's left edge and 25 pixels from the browser's top edge.

```
<body>
  <span
class="special">Special</span>
</body>
```

- ○ True
- ○ False

2) Elements using the "special" class that do not have a positioned ancestor will scroll with the page contents.

- ○ True
- ○ False

3) If the "container" class uses fixed positioning, the will not scroll with the page contents.

```
<div class="container">
  <span
class="special">Special</span>
</div>
```

- ○ True
- ○ False

4) If the "container" class uses static positioning, the `<span>` is positioned relative to the `<div>`.

```
<body>
  <div class="container">
    <span
class="special">Special</span>
  </div>
</body>
```

- True
- False

z-index property

When a relative, absolute, or fixed element is placed on top of another positioned element, the element that is specified last in the HTML is placed on top. However, the CSS ***z-index*** property is used to specify a relative distance that orders the appearance of elements. Elements with higher `z-index` values are placed on top of elements with lower `z-index` values.

On the left side of the figure below, the browser renders the square elements in the order the elements appear in the HTML: The orange square is rendered first, and the green square is rendered last. The right side of the figure shows how the ordering changes using the `z-index` property: The orange square has the largest `z-index` and therefore appears on top.

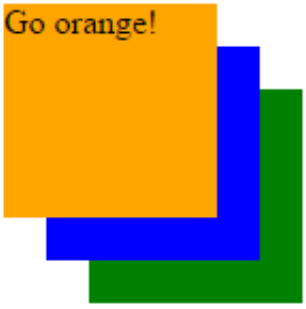
Figure 5.3.2: No z-index is used on the left, but z-index changes the rendered order on the right.

```
div {
  width: 100px;
  height: 100px;
  position: absolute;
}
#orange {
  background-color: orange;
  left: 10px;
  top: 10px;
}
#blue {
  background-color: blue;
  color: white;
  left: 30px;
  top: 30px;
}
#green {
  background-color: green;
  left: 50px;
  top: 50px;
}
```

```
<div id="orange">Go orange!
</div>
<div id="blue">Go blue!</div>
<div id="green">Go green!
</div>
```


```
div {
  width: 100px;
  height: 100px;
  position: absolute;
}
#orange {
  background-color: orange;
  z-index: 3;
  left: 10px;
  top: 10px;
}
#blue {
  background-color: blue;
  color: white;
  z-index: 2;
  left: 30px;
  top: 30px;
}
#green {
  background-color: green;
  z-index: 1;
  left: 50px;
  top: 50px;
}
```

```
<div id="orange">Go orange!
</div>
<div id="blue">Go blue!</div>
<div id="green">Go green!
</div>
```



PARTICIPATION ACTIVITY 5.3.6: z-index.

Refer to the figure above.

1) In the example on the right, what `z-index` value would make the green square appear on top of the orange and blue squares?

   - 1
   - 2
   - 4

2) If all three squares are given the same `z-index` value of 5, which square appears on top?

   - orange
   - blue
   - green

PARTICIPATION ACTIVITY 5.3.7: Positioning practice.

The webpage below displays the iconic "I ♥ NY" logo. Use the `position` and `z-index` properties to make the webpage render like the expected webpage:

1. Use relative positioning in the `img` CSS rule to place the t-shirt image 10 pixels further to the right of the image's default location.
2. Use absolute positioning in the `.first`, `.heart`, and `.last` CSS rules to place "I", "♥" and "NY" in the correct configuration on top of the t-shirt.

HTML CSS

```
<span class="first words">I</span>
<span class="heart">&hearts;</span>
<span class="last words">NY</span>

<div>
   <img src="https://resources.zybooks.com/WebProgramming/tshirtv1.png
</div>

```

Render webpage Reset code

Your webpage

Expected webpage

▶ View solution

CHALLENGE ACTIVITY 5.3.1: Positioning elements.

550544.4142762.qx3zqy7

Start

For the element with id of content, set the position to relative, the top to 9 pixels, and the left to -6 pixels. **SHOW EXPECTED**

CSS HTML

```
#content {

   /* Your solution goes here */

}
```

1 2 3 4

Check Next

View your last submission

Exploring further:

- CSS Layout - The position Property from W3Schools

5.4 Special effects

Text shadows

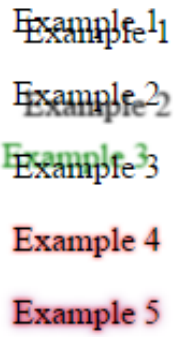
Shadows are added to text using the CSS property ***text-shadow***, which accepts four values:

- `offset-x` - Horizontal pixel offset of shadow
- `offset-y` - Vertical pixel offset of shadow
- `blur-radius` - Optional shadow blur (default is 0)
- `color` - Optional shadow color (default is usually the current CSS color)

The figure below illustrates how the four property values change the shadow.

Figure 5.4.1: Examples of different text-shadow values.

```
<p style="text-shadow: 5px 5px">Example 1</p>
<p style="text-shadow: 5px 5px 1px;">Example 2</p>
<p style="text-shadow: -5px -5px 1px green;">Example 3</p>
<p style="text-shadow: 0 0 3px red;">Example 4</p>
<p style="text-shadow: 0 0 3px red, 0 0 6px purple;">Example
5</p>
```



PARTICIPATION ACTIVITY 5.4.1: Text shadows.

1) Positive `offset-x` and `offset-y` make the shadow appear to the right and below the text, but negative values make the shadow appear to the left and above the text.

   - True
   - False

2) The `offset-x` and `offset-y` must be a non-zero value.

- True
- False

3) A shadow with `blur-radius:4px` is less blurry than a shadow with `blur-radius:2px`.

- True
- False

4) Multiple shadows can apply to the same text.

- True
- False

Box shadows

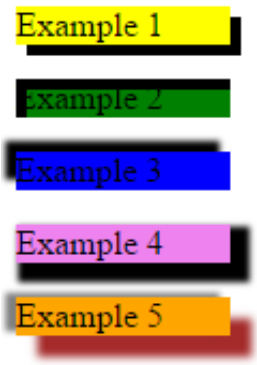
The CSS property ***box-shadow*** adds a shadow to the box around an element using the following properties:

- `inset` - Optional value that draws the shadow inside the box (default is outside the box)
- `offset-x` - Horizontal pixel offset of shadow
- `offset-y` - Vertical pixel offset of shadow
- `blur-radius` - Optional shadow blur (default is 0)
- `spread-radius` - Positive value causes shadow to grow, negative values to shrink (default is 0)
- `color` - Optional shadow color (default is usually the current CSS color)

Figure 5.4.2: Examples of different box-shadow values.

```
p {
   width: 100px;
}
#example1 {
   background-color: yellow;
   box-shadow: 5px 5px;
}
#example2 {
   background-color: green;
   box-shadow: inset 5px 5px;
}
#example3 {
   background-color: blue;
   box-shadow: -5px -5px 3px;
}
#example4 {
   background-color: violet;
   box-shadow: 5px 5px 3px 4px;
}
#example5 {
   background-color: orange;
   box-shadow: -5px -2px 3px gray, 10px 10px 5px brown;
}
```

```
<p id="example1">Example 1</p>
<p id="example2">Example 2</p>
<p id="example3">Example 3</p>
<p id="example4">Example 4</p>
<p id="example5">Example 5</p>
```



PARTICIPATION ACTIVITY 5.4.2: Box shadows.

1) The `box-shadow` property creates a shadow for text.

   - True
   - False

2) If the `box-shadow` uses the value `inset`, then the shadow appears inside the box.

   - True
   - False

3) A zero `spread-radius` makes the shadow the same size as the box.

   - True
   - False

PARTICIPATION ACTIVITY 5.4.3: Shadow practice.

The webpage below displays three flash cards with web history questions and answers. Modify the CSS to add shadows to the cards, question text, and answer text. Make the shadows use different colors, offsets, and blur radiuses.

HTML CSS

```
<div class="card">
  <p class="question">Q: Who invented the WWW?</p>
  <p class="answer">A: Tim Berners-Lee</p>
</div>

<div class="card">
  <p class="question">Q: When was the first website published?</p>
  <p class="answer">A: 1991</p>
</div>

<div class="card">
  <p class="question">Q: What web browser did most people use in the
  <p class="answer">A: Internet Explorer</p>
</div>

```

Render webpage Reset code

Your webpage

Q: Who invented the WWW?

A: Tim Berners-Lee

Q: When was the first website published?

A: 1991

Q: What web browser did most people use in the early 2000s?

▶ View solution

Rounded corners

An element border's corners can be rounded using the CSS property ***border-radius***, which is

assigned one to four radius values.

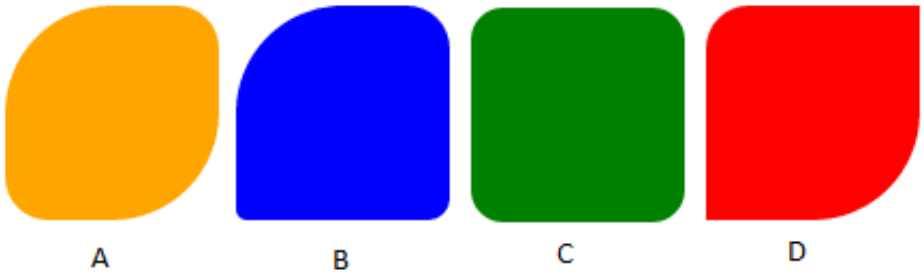
- Single value - All four corners are equally rounded
- Two values - First value is top-left and bottom-right corners, second value is top-right and bottom-left corners
- Three values - First value is top-left, second is top-right and bottom-left, third is bottom-right
- Four values - First value is top-left, second is top-right, third is bottom-right, fourth is bottom-left

Each corner may also be assigned a radius using four CSS properties:

- `border-top-left-radius`
- `border-top-right-radius`
- `border-bottom-left-radius`
- `border-bottom-right-radius`

PARTICIPATION ACTIVITY 5.4.4: Rounded corners.

Match the square with the CSS that produces the square's rounded corners.



If unable to drag and drop, refresh the page.

C B A D

```
border-radius: 40px 20px 10px 5px;
```

```
border-radius: 40px 20px;
```

```
border-top-left-radius: 20px;
border-bottom-right-radius: 50px;
```

```
border-radius: 15px;
```

Reset

Border images

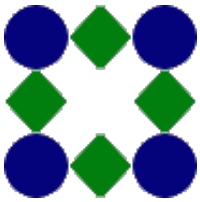
The CSS property ***border-image*** renders an element's border using sections of an image. The border image takes the place of any border properties specified by `border-style`. The following CSS properties are specified by `border-image` all at once:

- `border-image-source` - Image URL
- `border-image-slice` - Image section size
- `border-image-repeat` - "repeat" to repeat the image section, "round" to repeat the image section but resize the image if needed to fit, or "stretch" to stretch an image section

A border image will not appear unless the `border-width` property is non-zero and `border-style` is set to any legal value except `none` and `hidden`.

PARTICIPATION ACTIVITY 5.4.5: Try different border-image values.

The borderv2.png image is used to display a border image around the `<div>` in the webpage below. The blue circles and green diamonds in the image are each 30 × 30 pixels.



Change the following CSS property values to see the effect on the border image:

1. Change the image slice size from 30 to 15 in the CSS property `border-image`. Render the page and observe how half the circle and half the diamond is used to render the border.
2. Change the `border-image` slice size from 15 to 60. Render the page and observe how a 60 × 60 pixel section (2/3 of the image) is used to render the border corners. Since borderv2.png is only 90 × 90 pixels, an unused 60 × 60 pixel section does not exist, so the border sides are empty.

3. Change the `border` size from 15px to 30px. Render the webpage and observe how the border size increased.
4. Change the `border-image` slice size back to 30, and change "round" to "repeat". Render the page and observe how the green diamonds are repeated but do not fit perfectly on the left and right sides.
5. Finally, change the `border-image` "repeat" to "stretch". Render the page and observe how the green diamonds stretch to fill the border.

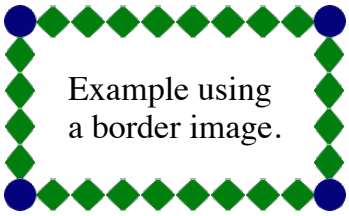
HTML CSS

```
<div id="example">
   Example using a border image.
</div>

```

Render webpage Reset code

Your webpage



▶ View solution

PARTICIPATION ACTIVITY 5.4.6: Border images.

Refer to the CSS below.

```
border-image: url(some-border.png) 50 repeat;
```

1) 50 × 50 pixel sections of some-border.png are used to create the border image.
   - ○ True
   - ○ False

2) If some-border.png is 50 × 50 pixels, then the border will have empty sides.
   - ○ True
   - ○ False

3) If some-border.png is 150 × 150 pixels, the border image section is stretched on the sides.

- True
- False

4) To display the 50 × 50 pixel sections evenly around an element's border image, the element's border should be specified as: `border: 50 solid;`.

- True
- False

CSS3 browser support

*Most modern browsers support CSS3, but some CSS3 properties require vendor prefixes to work on certain browsers. A **vendor prefix** is a prefix added to an experimental or nonstandard CSS property that only works on a specific browser type. Typical vendor prefixes are:*

- ***-webkit-*** *for Chrome, Safari, and newer versions of Opera*
- ***-moz-*** *for Firefox*
- ***-ms-*** *for Internet Explorer*
- ***-o-*** *for older versions of Opera*

The following CSS specifies a ***border-image*** *property for WebKit and Opera browsers:*

```
#borderimg {
   -webkit-border-image: url(border.png) 30 round; /* Safari 3.1-5
*/
   -o-border-image: url(border.png) 30 round; /* Opera 11-12.1 */
   border-image: url(border.png) 30 round;
}
```

Linear gradients

A CSS background may use gradient colors that transition from one color to another. Two CSS gradients exist:

1. Linear gradient - A gradient that follows a straight line
2. Radial gradient - A gradient that radiates outward into an ellipse

The CSS function ***linear-gradient(color1, color2)*** creates a linear gradient that transitions from `color1` to `color2` when moving from the top edge to the bottom edge. Additional colors can be supplied to the function. Ex: `linear-gradient(red, green, blue, yellow)` transitions from red to green to blue to yellow when moving from top to bottom.

To change the gradient's direction, the first argument to `linear-gradient` can be a direction or an angle:

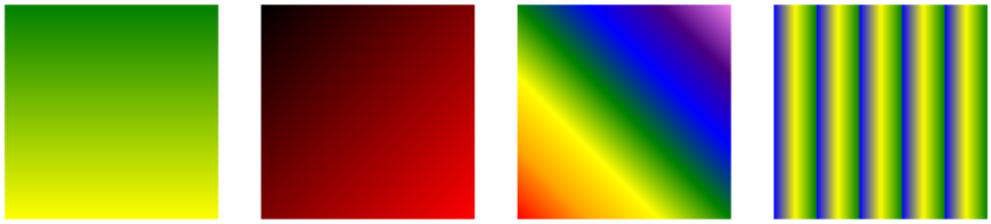
- **Direction** - A direction of `left`, `right`, `top`, or `bottom` with the word `to` in front. Ex: `to left` creates a linear gradient that moves from right to left, and `to bottom right` goes from the top-left corner to the bottom-right corner.
- **Angle** - A CSS angle that points in the direction of the linear gradient. The angles `0deg`, `90deg`, `180deg`, and `270deg` correspond to `to top`, `to right`, `to bottom`, and `to left`, respectively.

The ***repeating-linear-gradient()*** function repeats a linear gradient where the color values are supplied an optional percent. The percentage value after the last color is the percent of the gradient's total length the repeating gradient should occupy. Ex: `repeating-linear-gradient(red, yellow 10%)` means the red to yellow gradient occupies 10% of the gradient's total length and is repeated to fill the entire background.

Figure 5.4.3: Examples of linear gradients.

```
#example1 {
   background: linear-gradient(green, yellow);
}
#example2 {
   background: linear-gradient(to bottom right, black, red);
}
#example3 {
   background: linear-gradient(45deg, red, orange, yellow, green, blue,
indigo, violet);
}
#example4 {
   background: repeating-linear-gradient(to right, blue, yellow, green
20%);
}
```

```
<div id="example1"></div>
<div id="example2"></div>
<div id="example3"></div>
<div id="example4"></div>
```



PARTICIPATION ACTIVITY 5.4.7: Linear gradients.

1) What direction creates the gradient below?

```
background: linear-gradient(_______, orange, red);
```

Check Show answer

2) What angle (direction of red arrow) creates the gradient below?

```
background: linear-gradient(_______, blue, green);
```

Check Show answer

3) What color and percent creates the repeating linear gradient that ends in white?

```
background: repeating-linear-gradient(black, _______);
```

Check Show answer

Radial gradients

A radial gradient is created with the CSS function ***radial-gradient(color1, color2)***, which creates an ellipse-shaped gradient that begins with `color1` in the center and ends with `color2` on the perimeter. More than two colors may be specified. A percentage or length can be placed after a color to give more emphasis to the color. Ex: `radial-gradient(red 10%, yellow 30%)` gives more emphasis to red and yellow than the default rendering.

The ellipse shape of a radial gradient fits the gradient's bounding rectangle. However, a circular radial gradient can be created with the `circle` argument. Ex: `radial-gradient(circle, red, yellow)` creates a circle gradient.

Figure 5.4.4: Examples of radial gradients.

```
#example1 {
   background: radial-gradient(red, orange);
}
#example2 {
   background: radial-gradient(red, orange 50%);
}
#example3 {
   background: radial-gradient(red 20%, orange 50%);
}
#example4 {
   background: radial-gradient(circle, red 20%, orange 50%);
}
```

PARTICIPATION ACTIVITY 5.4.8: Radial gradient.

1) A radial gradient is always an ellipse or circle.

   - True
   - False

2) The radial gradient below has a blue interior and a green exterior.

```
radial-gradient(green, blue);
```

   - True
   - False

3) What arguments to `radial-gradient()` create the radial gradient below?

- black, white, red
- black 40%, white, red

Positioning radial gradients

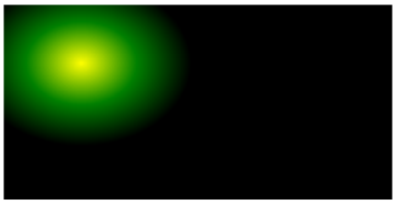
A radial gradient's ellipse or circle is centered by default in the enclosing rectangle, but the center position can be specified using "`at centerX centerY`" where `centerX` and `centerY` specify a distance or percentage. Ex: `radial-gradient(at 50px 10px, yellow, green)` specifies a center that is 50px from the left edge and 10px from the top.

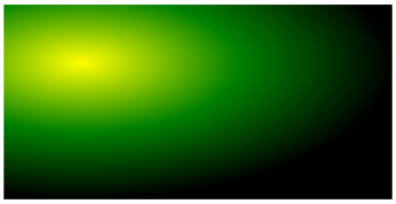
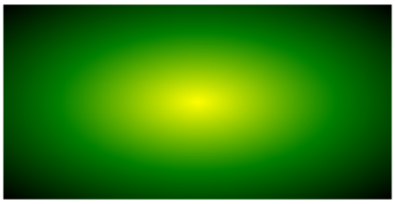
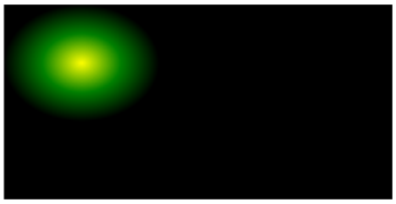
By default, a radial gradient's shape reaches to the farthest corner of the containing rectangle. An extent keyword describes the size of the radial gradient's shape:

- `closest-side` - Circle touches the rectangle's side closest to the circle's center. Ellipse touches the vertical and horizontal sides closest to the ellipse's center.
- `farthest-side` - Circle touches the rectangle's side farthest from the circle's center. Ellipse touches the vertical and horizontal sides farthest from the ellipse's center.
- `closest-corner` - Circle or ellipse touches the corner closest to the shape's center.
- `farthest-corner` - Circle or ellipse touches the corner farthest from the shape's center. (Default behavior.)

PARTICIPATION ACTIVITY 5.4.9: Positioned radial gradients.

Match the background with the radial gradient CSS that produced the background.

A. 
B. 
C. 

D. 
E. 
F. 

If unable to drag and drop, refresh the page.

E C A F B D

	radial-gradient(at 60px 100px, yellow, green, black)
	radial-gradient(yellow, green, black)
	radial-gradient(closest-corner at 20% 30%, yellow, green, black)
	radial-gradient(farthest-side at 20% 30%, yellow, green, black)
	radial-gradient(closest-side at 20% 30%, yellow, green, black)
	radial-gradient(farthest-corner at 20% 30%, yellow, green, black)

Reset

PARTICIPATION ACTIVITY 5.4.10: Gradient practice.

The webpage below displays an advertisement with a background produced by the CSS function `repeating-radial-gradient()`. Make the following modifications to the HTML and CSS so the rendered webpage resembles the expected webpage:

1. Add a radial gradient background to the `<body>` using any colors you prefer, and position the ellipse close to the bottom-right corner.
2. Create two more advertisements like the ads in the expected webpage. Choose whatever fonts and colors you prefer. One ad should have a linear gradient background and the other a repeating linear gradient background.

HTML CSS

```
1 <div id="ad1">Vote this Tuesday!</div>
2
```

Render webpage Reset code

Your webpage

Vote this Tuesday

Expected webpage

Vote this Tuesday

Buy some chocolate!

UPGRADE YOUR PHONE

▶ View solution

CHALLENGE ACTIVITY 5.4.1: Special effects.

550544.4142762.qx3zqy7

Start

For the <p> tag, set the text-shadow to be green with an offset-x of 3px and offset-y of 10px. **SHOW EXPECTED**

CSS HTML

```
1 p {
2
3    /* Your solution goes here */
4
5 }
```

1 2 3 4

Check Next

View your last submission

Exploring further:

- CSS3 text-shadow Property from W3Schools
- CSS3 box-shadow Property from W3Schools
- CSS3 Rounded Corners from W3Schools
- CSS3 border-image Property from W3Schools
- CSS3 Gradients from W3Schools

5.5 Animation

Animations and keyframes

Developers often use JavaScript and JavaScript libraries like jQuery to produce animations. However, CSS can also be used to produce animations. A ***CSS animation*** transforms an element's styles over a set time period, producing an animation. CSS animations have three advantages over JavaScript animations:

1. CSS animations do not require any JavaScript code.
2. CSS animations often put less load on the computer and can use techniques to produce smoother animations when the computer's resources are limited.
3. CSS animations allow the browser to more efficiently control animations and stop animations from running in browser tabs that are not visible.

A CSS animation's behavior is defined with the ***@keyframes*** rule, which contains a keyframe list. A ***keyframe list*** has a name and contains the keyframes or the properties and values that will be animated. A keyframe list contains two keyframe selectors:

- ***from*** - The animation starting state that lists the CSS properties and values that apply when the animation begins
- ***to*** - The animation ending state that lists the CSS properties and values that the "from" values become by the time the animation ends

Percentages may be used to specify keyframes at various points during the animation. Ex: `0%` is equivalent to `from` and `100%` is equivalent to `to`. The value `50%` indicates the animation state at the halfway point.

To create an animation, two CSS properties must be defined:

- ***animation-name*** - Names the keyframe list associated with the animation
- ***animation-duration*** - Length of the animation in seconds (s) or milliseconds (ms)

An animation begins immediately when the browser renders the webpage unless an ***animation-delay*** is used to delay the start of the animation.

PARTICIPATION ACTIVITY 5.5.1: Animating the background color.

```
div {
  width: 100px;
  height: 100px;
  background-color: orange;
  animation-name: changeColors;
  animation-duration: 1s;
  animation-delay: 2s;
}

@keyframes changeColors {
  from { background-color: blue;  }
  to   { background-color: green; }
}
```

```
<div></div>
```

Animation content:

Two blocks of code are shown. The CSS:
div {
 width: 100px;
 height: 100px;
 background-color: orange;
 animation-name: changeColors;
 animation-duration: 1s;
 animation-delay: 2s;
}
@keyframes changeColors {
 from { background-color: blue; }
 to   { background-color: green; }
}
and the HTML:

A screen shows the div element, displaying as an orange box. The div container includes animation properties. The browser waits 2 seconds before playing the change color animation because of the animation-delay property. Then the animation will play over 1 second, as assigned

by the animation-duration, and in that time will change from blue to green. The div element is initially orange, so when the animation starts it becomes blue.

Animation captions:

1. "animation-name" names the keyframe list associated with the animation.
2. "animation-duration" specifies the animation will last 1 second.
3. "animation-delay" tells the browser to wait 2 seconds before starting the animation.
4. After the 2 second delay, the animation begins with "from", so the background is initially blue.
5. During the 1 second duration, "to" indicates the background color becomes green.
6. The background becomes orange again after the animation completes.

PARTICIPATION ACTIVITY 5.5.2: Keyframes and animation.

1) A valid keyframe list must include the `from` and `to` keyframe selectors.
   - True
   - False

2) A property that is listed in the `from` keyframe selector but not the `to` keyframe selector will still animate.
   - True
   - False

3) An animation without an `animation-delay` property begins immediately.
   - True
   - False

4) If `animation-duration` is assigned the value 0s, the animation occurs very quickly.

- True
- False

PARTICIPATION ACTIVITY 5.5.3: Percentages for keyframes.

The webpage below shows a smiley face that moves to the right while changing colors to blue, then moves back to the left while changing colors back to red. Add two keyframes to the animation:

1. 25% through the animation, make the smiley face appear 200 pixels to the right and 100 pixels below the smiley face's starting location. The smiley face should also become yellow.
2. 75% through the animation, make the smiley face appear against the left side of the webpage and 100 pixels below the smiley face's starting location. The smiley face should also become green.

Rendering the webpage should animate the smiley face down and to the right, up, down and to the left, and up again while changing colors.

HTML CSS

```
1 <div id="smiley">&#9786;</div>
2
```

Render webpage Reset code

Your webpage

▶ View solution

Timing, iteration count, and direction

In the smiley face animation above, each transition from one keyframe to the next began with a

slow start, then fast, then a slow end. The ***animation-timing-function*** property controls an animation's speed between keyframes. Several timing functions are available:

- `ease` - Slow start, then fast, then slow end (default)
- `linear` - Same speed throughout
- `ease-in` - Slow start
- `ease-out` - Slow end
- `ease-in-out` - Slow start and end
- `cubic-bezier(n1,n2,n3,n4)` - Specify numbers that control speed based on a Bezier curve

Other animation properties include:

- ***animation-iteration-count*** - Indicates the number of times the animation will run. The value `infinite` runs the animation repeatedly without stopping. Ex: `animation-iteration-count: 3` runs the animation three times.
- ***animation-direction*** - Indicates animation direction
  - `normal` - Normal direction (default)
  - `reverse` - Reverse direction
  - `alternate` - Alternate between normal and reverse
  - `alternate-reverse` - Alternate between reverse and normal
- ***animation*** - Shorthand property indicating the animation name, duration, timing function, delay, iteration count, and direction. Ex: `animation: move 3s linear 2s infinite normal`.

PARTICIPATION ACTIVITY 5.5.4: Keyframes and timing functions.

1) Which keyframe selector is equivalent to `to`?
   - 0%
   - 100%
   - 50%

2) Which keyframe selector specifies the animation state when the animation is three quarters finished?

- 0%
- 50%
- 75%

3) Which timing function makes the animation progress at the same speed the entire time?

- linear
- ease
- cubic-bezier

4) How many times will the animation below run?

```
animation: move 4s ease 1s 2 reverse;
```

- once
- twice
- infinite

5) What color is `#thing`'s font right as the animation completes?

```
#thing {
  animation: changeColors 4s ease 1s 2 reverse;
}
@keyframes changeColors {
  0%   { color:red; }
  50%  { color:blue; }
  100% { color:green; }
}
```

- red
- blue
- green

Transitions

A ***CSS transition*** animates an element's transition from one state to another when an element's CSS property changes value. Ex: A transition may animate an element getting wider when the element's width is increased. Transitions are commonly used with the `:hover` pseudo-class to trigger an animation when the user mouses over an element.

Transitions differ from CSS animations in two ways:

1. Transitions execute when an element's property values are changed, unlike CSS animations that execute at a particular time.
2. Transitions provide less control over the animation than CSS animations.

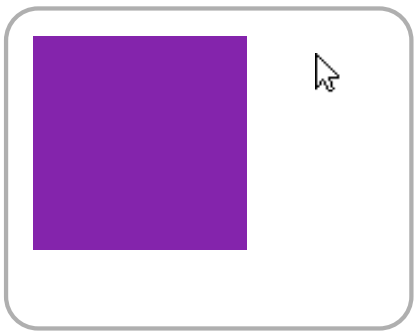
The ***transition*** property defines a transition by specifying one or more CSS properties and each property's transition duration.

PARTICIPATION ACTIVITY 5.5.5: Transitioning the width and height when hovering.

```
div {
  width: 100px;
  height: 100px;
  background-color: purple;
  transition: width 1s, height 1s;
}

div:hover {
  width: 120px;
  height: 120px;
}
```

```
<div></div>
```



Animation content:

There are 2 blocks of code. The CSS:
div {
  width: 100px;
  height: 100px;

```
  background-color: purple;
  transition: width 1s, height 1s;
}
div:hover {
  width: 120px;
  height: 120px;
}
```

and the HTML:

```
<div></div>
```

A screen shows the div element, a purple box that is 100 by 100 pixels. When the mouse hovers over the box it expands to 120 by 120 pixels over 1 second. When the mouse stops hovering, the div returns to 100 by 100 pixels over 1 second. The 1 second time interval is determined by the transition property. The div:hover class controls the hover animation.

Animation captions:

1. When the width property or height properties are changed, the transition to the new values will be animated over 1 second.
2. The width and height properties are increased to 120px when the mouse hovers over the .
3. So, when the mouse cursor hovers over the , the width and height transitions from 100px to 120px, animated over 1 second.
4. When the cursor no longer hovers over the , the transition from 120px to 100px is animated over 1 second.

The ***transition-timing-function*** property controls the speed of the transition. Several timing functions are available, and all complete in the same amount of time:

- `ease` - Slow start, then fast, then slow end (default)
- `linear` - Same speed throughout
- `ease-in` - Slow start
- `ease-out` - Slow end
- `ease-in-out` - Slow start and end
- `cubic-bezier(n1,n2,n3,n4)` - Specify numbers that control speed based on a Bezier curve

The ***transition-delay*** property delays the transition's start.

PARTICIPATION ACTIVITY 5.5.6: Transitions.

1) A transition can animate one or more CSS properties.

   - True
   - False

2) The `transition` property below makes the width take 3 seconds longer than the height to complete the transition.

```
transition: width 3s, height 1s;
```

   - True
   - False

3) The `#div1` transition takes longer to complete than the `#div2` transition.

```
div {
  width: 100px;
  height: 100px;
  transition: width 1s;
}

#div1 { transition-timing-function: ease-in; }
#div2 { transition-timing-function: linear; }
```

   - True
   - False

4) According to the CSS below, a div element would not decrease in size until 500 milliseconds after the cursor hovered over the div.

```
div {
  width: 100px;
  height: 100px;
  transition: width 1s;
  transition-delay: 500ms;
}

div:hover {
  width: 80px;
}
```

- ○ True
- ○ False

5) The CSS below causes a paragraph to disappear when the mouse hovers over the paragraph.

```
p {
  transition: opacity 500ms;
}

p:hover {
  opacity: 0;
}
```

- ○ True
- ○ False

Transformations

The ***transform*** property applies a 2D or 3D transformation to an element. A ***transformation*** is a graphical operation that alters the position, shape, or orientation of an object. The `transform` property is assigned a transformation function. A selected number of 2D transformation functions are summarized in the table below.

Transformations are used in animations and transitions to create engaging webpages.

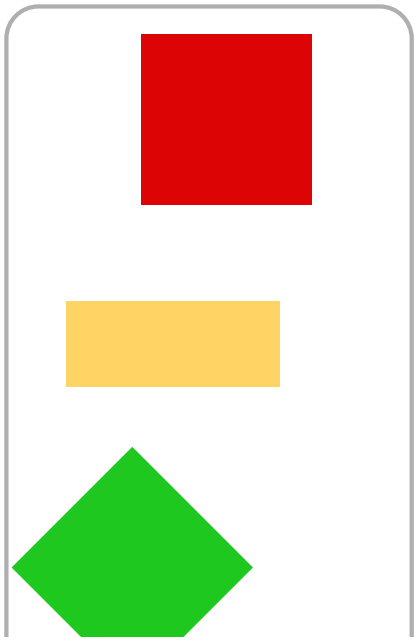
Table 5.5.1: Selected 2D transformation functions.

Function	Description	Example
translate(x, y)	Moves an element on the x-axis `x` distance and along the y-axis `y` distance	`/* Moves right 10px and up 20px */` `translate(10px, -20px)`
scale(x, y)	Increases (values > 1) or decreases (values < 1) the width and height by the `x` and `y` multiplier	`/* Halves the width, doubles the height */` `scale(0.5, 2)`
rotate(angle)	Rotates clockwise by `angle`	`/* Rotates clockwise 45 degrees */` `rotate(45deg)`

PARTICIPATION ACTIVITY 5.5.7: translate(), scale(), and rotate() transformation functions.

```
#example1 {
  background-color: red;
  transform: translate(30px, -10px);
}
#example2 {
  background-color: yellow;
  transform: scale(1.2, 0.5);
}
#example3 {
  background-color: green;
  transform: rotate(45deg);
}
```

```
<div id="example1"></div>
<div id="example2"></div>
<div id="example3"></div>
```



Animation content:

There are two blocks of code. The CSS:

```
#example1 {
  background-color: red;
  transform: translate(30px, -10px);
}
#example2 {
  background-color: yellow;
  transform: scale(1.2, 0.5)
}
#example3 {
  background-color: green;
  transform: rotate(45deg)
}
```

and the HTML:

```
<div id="example1"></div>
<div id="example2"></div>
<div id="example3"></div>
```

A screen shows a red, yellow, and green box aligned vertically. The translate() function moves the red div up and to the right. The scale() function widens and flattens the yellow div. The rotate() function turns the green div to the right.

Animation captions:

1. All three squares are displayed at default locations.
2. translate() moves the square 30 pixels to the right and 10 pixels up.
3. scale() multiplies the yellow square's width by 1.2 and the height by 0.5, so the square is 20% wider and 50% shorter.
4. rotate() rotates the green square 45 degrees clockwise.

PARTICIPATION ACTIVITY 5.5.8: Animations, transitions, and transformations practice.

The webpage below displays a welcome message and a large button. When the cursor hovers over the button, the font color turns red, and the button grows larger. When the button is pressed, the background color is darkened.

The button transition is defined in the `.button` class and specifies that "all" properties should be animated in a transition in 100 milliseconds using the "ease-in-out" timing function. The `:hover` pseudo-class is used to scale the button 5% larger and change the font color, and the `:active` pseudo-class is used to change the background color.

Make the following modifications to the CSS so the rendered webpage behaves like the expected webpage:

1. Add an animation that runs as soon as the page is rendered that moves the text onto the screen. Use the `translate()` function to move the welcome message to -300px initially, then to 60px half way through the animation, then back to 0px at the end.
2. Add an animation that wiggles the button one second after the page renders. Use the `rotate()` function to initially rotate 0 degrees, then 3 degrees, then -3 degrees, and then 0 degrees. The animation should run twice.
3. Experiment with different timing functions. The expected webpage uses linear timing for the two animations listed above.
4. Finally, add a transformation to the `:active` pseudo-class that uses the `scale()` function to scale the button size down 5%.

HTML CSS

```
1 <div id="welcome">Enter the exciting world of fantasy sports!</div>
2 <div class="button">Play Now!</div>
3
```

Render webpage Reset code

Your webpage

Enter the exciting world of fantasy sports!

Play Now!

Expected webpage

Enter the exciting world of fantasy sports!

Play Now!

▶ View solution

PARTICIPATION ACTIVITY 5.5.9: Transformations in transitions and animations.

1) `translate()`, `scale()`, and `rotate()` are ____ functions.

- transition
- transformation
- translation

2) Which function moves an element 20 pixels to the left and 5 pixels down?

- `translate(-20, 5)`
- `translate(20px, -5px)`
- `translate(-20px, 5px)`

3) What does the element using the keyframes below look like at the end of the animation?

```
@keyframes example {
   from { transform:
scale(0.5, 0.5); }
   to   { transform:
rotate(45deg) scale(1.5, 1.5);
}
}
```

- Scaled smaller
- Rotated 45 degrees only
- Rotated 45 degrees and scaled larger

4) What does the <p> look like when the mouse clicks the element?

```
p {
   transition: all 1s;
}
p:hover {
   transform: translate(10px,
0);
}
p:active {
   transform: scale(2, 2);
}
```

- Scaled larger
- Translated 10 pixels to the right
- No change

Exploring further:

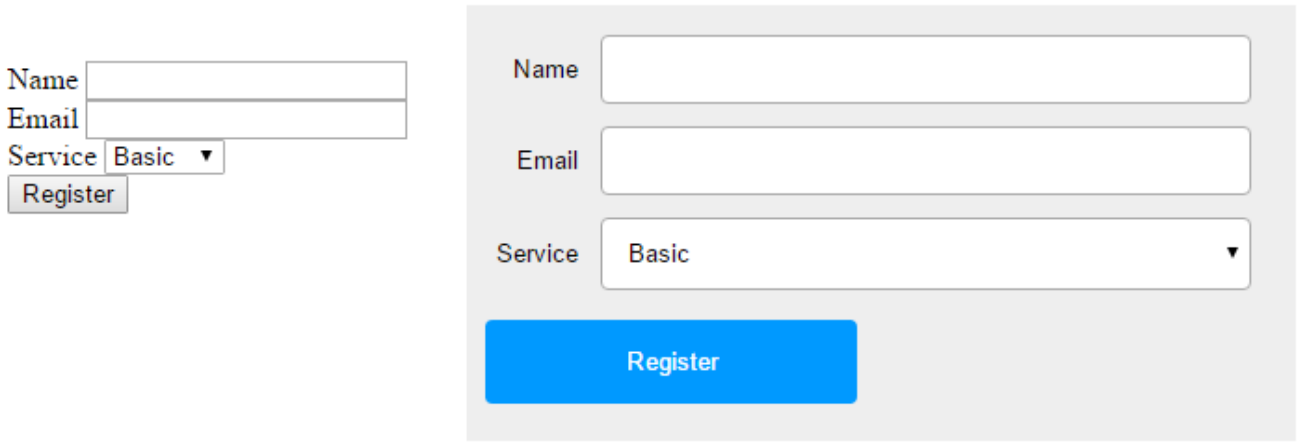
- CSS3 Animations from W3Schools
- CSS3 Transitions from W3Schools
- CSS3 2D Transforms from W3Schools
- CSS3 3D Transforms from W3Schools

5.6 Styling forms

Styling forms with CSS

Web forms are an important part of many websites. A usable form allows the user to quickly and painlessly enter data. Forms require CSS formatting to improve usability.

Figure 5.6.1: HTML form without CSS styling and an improved form with styling.



PARTICIPATION ACTIVITY 5.6.1: Create a styled form.

The webpage below displays a simple web form with little styling. Add the following CSS to create a more usable web form:

1. Add a `label` selector that makes all labels have the same width and margin. Since a label is an inline element, the label's width cannot be changed without making a label an inline-block. Also, right-align the label text to improve the reader's ability to mentally link the label to the input field.

```
label {
   width: 50px;
   display: inline-block;
   text-align: right;
   margin-right: 8px;
}
```

   Render the webpage and observe the labels are equal length and right aligned.

2. Add a selector that gives the text input, email input, and drop-down menu the same consistent width with some padding to increase the size of the inputs. Also, change the border color and radius to give the inputs a softer look.

```
input[type=text], input[type=email], select {
   box-sizing: border-box;
   width: 250px;
   padding: 10px;
   border: 1px solid #aaa;
   border-radius: 4px;
}
```

Render the webpage and observe the inputs are equal size and are spaced out.

3. Add styling to the submit button to create a blue button that looks less like a traditional browser button. Also, change the default mouse cursor to a pointer icon to give the user a visual cue that the button is pressable.

```
input[type=submit] {
   width: 200px;
   background-color: #09f;
   color: white;
   padding: 15px;
   margin: 10px 0;
   border: none;
   border-radius: 4px;
   cursor: pointer;
}
```

Render the webpage and observe the Register button is blue and much larger. Moving the mouse over the button changes the pointer icon.

4. To give the user another visual cue that the button is clickable, darken the button color when the mouse hovers over the button.

```
input[type=submit]:hover {
   background-color: #07d;
}
```

Render the webpage and verify the Register button's color gets darker when the mouse hovers on the button.

HTML CSS

```
<form action="https://wp.zybooks.com/form-viewer.php" target="_blank'
   <p>
      <label for="name">Name</label>
      <input type="text" id="name" name="fullname">
   </p>
   <p>
      <label for="email">Email</label>
      <input type="email" id="email" name="email">
   </p>
   <p>
      <label for="service">Service</label>
      <select id="service" name="service">
        <option>Basic</option>
        <option>Prime</option>
        <option>Deluxe</option>
      </select>
```

Render webpage Reset code

Your webpage

Name

Email

Service Basic

Register

Expected webpage

Name

Email

Service

Basic

Register

▶ View solution

PARTICIPATION ACTIVITY 5.6.2: Form styles.

1) In the example above, the label's width could not be changed until which CSS property/value was set?

   - `display: inline-block`
   - `display: block`
   - `display: none`

2) What CSS selector selects only text inputs?

   - `input`
   - `input[type=text]`
   - `input[type=text], select`

3) In the example above, what happens to the text and email inputs if `box-sizing: border-box;` is removed from the `input[type=text], input[type=email], select` selector's rule?

   - Inputs are no longer visible
   - Inputs are no longer editable
   - Inputs become wider

4) In the example above, what visual cues help the user to know that the blue rectangle with "Register" in the middle is a button that can be pressed?

- The button has rounded corners.
- The pointer icon appears when hovering over the button.
- The pointer icon appears, and the button color changes when hovering over the button.

Paragraphs vs. divs

The W3C recommends putting form elements in `<p>` tags, but many leading web frameworks like Bootstrap use `<div>` tags instead. Either container is acceptable, and this material uses both styles.

```
<!-- Example div container -->
<div>
   <label for="nickname">Nickname?</label>
   <input type="text" id="nickname" name="nickname">
</div>

<!-- Example p container -->
<p>
   <label for="nickname">Nickname?</label>
   <input type="text" id="nickname" name="nickname">
</p>
```

Proper use of field labels

Form field labels should be placed uniformly in the same location on a website's web forms. The best places for labels are immediately above or to the left of an input field. Some developers use only the ***`placeholder`*** *HTML attribute in place of labels to save screen space and reduce clutter, especially on mobile devices. However, usability experts warn that placeholders used as labels can create a number of problems for users and should be avoided.*

Improving text input fields

An input field can be further improved. Changing an input's border color or background color focuses the user's attention to the input. Ex: Changing the input border color to red may indicate an error with the input. Adding a commonly recognized icon to an input field can improve the user's ability to recognize the purpose of the input. Ex: Adding a search icon to a search input.

PARTICIPATION ACTIVITY 5.6.3: Augmenting an input.

The webpage below displays a web form on the left and a search box on the right. A partial email address is entered in the email input field. Add the following CSS:

1. Add an `:invalid` selector at the *end* of the CSS stylesheet that sets an invalid input's border to red to indicate an error:

```
input:invalid {
   border: 2px solid red;
}
```

   Render the webpage and observe the email input now has a red border.

2. Add a `:focus` selector that applies styles to an input that has the focus, and change the background color to light blue:

```
input[type=text]:focus, input[type=email]:focus {
   background-color: lightblue;
}
```

Render the webpage and observe that the name and email inputs become light blue when the inputs have the focus. The browser automatically places a border around an input that has the focus.

3. Add a search icon to the search input by adding a `background-image` that is positioned with `background-position`. Set `background-repeat:no-repeat` so the background image only displays once:

```
input[type=search] {
   float: right;
   background-image:
url("https://resources.zybooks.com/WebProgramming/searchiconv1.png");
   background-position: 5px 5px;
   background-repeat: no-repeat;
   padding-left: 40px;
}
```

Render the webpage and observe the search icon in the search input.

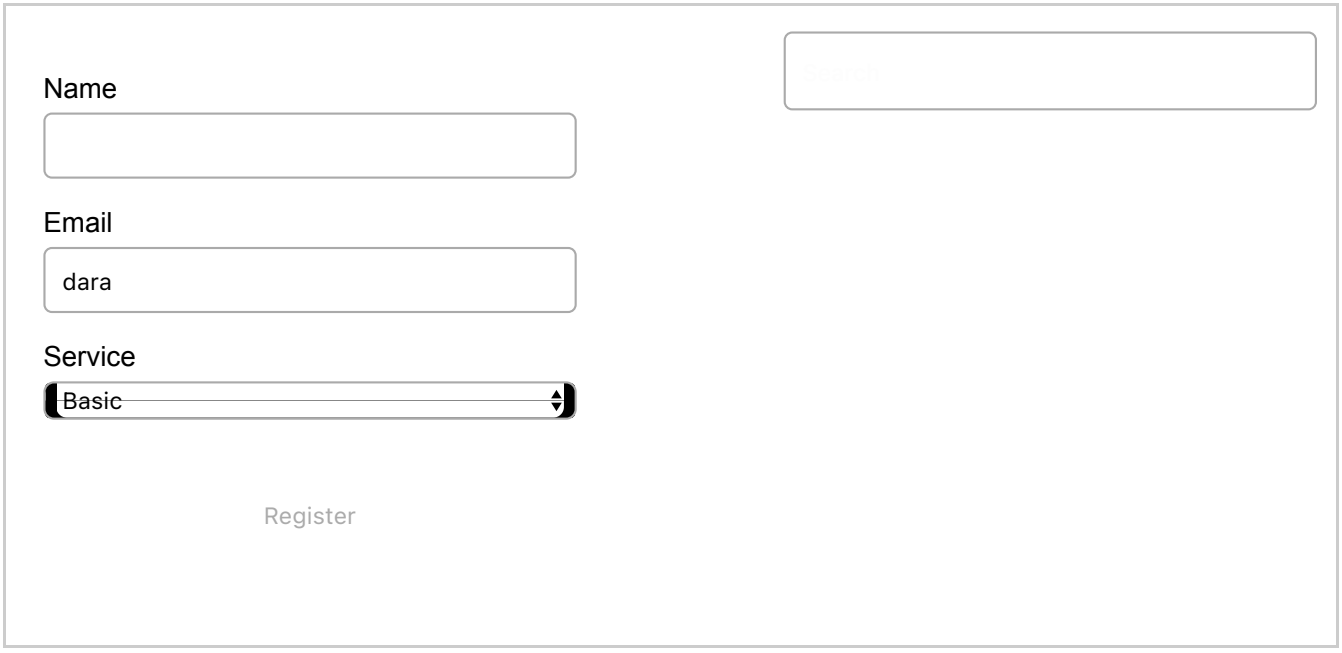
HTML CSS

```
<input type="search" placeholder="Search">

<form action="https://wp.zybooks.com/form-viewer.php" target="_blank"
   <p>
      <label for="name">Name</label>
      <input type="text" id="name" name="fullname">
   </p>
   <p>
      <label for="email">Email</label>
      <input type="email" id="email" name="email" value="dara">
   </p>
   <p>
      <label for="service">Service</label>
      <select id="service" name="service">
        <option>Basic</option>
        <option>Prime</option>
```

Render webpage Reset code

Your webpage



▶ View solution

PARTICIPATION ACTIVITY 5.6.4: Augmented inputs.

1) Many browsers add a border around an input when the input has the focus.
   - True
   - False

2) The `:focus` selector normally selects more than one element at a time.
   - True
   - False

3) Changing the email address to dara@gmail.com removes the red border.
   - True
   - False

Styling radio buttons and checkboxes

Radio buttons and checkboxes use the styling properties supplied by the browser and may differ between browsers. Styling radio buttons and checkboxes requires hiding the default widget, then displaying a custom widget before each label that changes appearance when checked or focused.

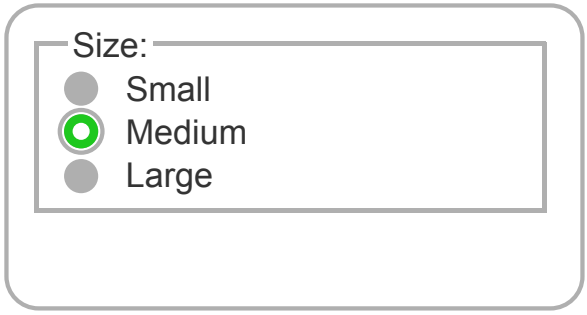
The CSS property ***appearance*** is used to control a widget's appearance based on the operating system's theme. Setting `appearance` to `none` hides the widget.

To display a custom radio button or checkbox, the `::before` pseudo-element selector and `content` property are used to insert content before the label's content that looks like a radio button or checkbox.

PARTICIPATION ACTIVITY 5.6.5: Styling radio buttons.

```
input[type=radio] + label {
   cursor: pointer;
}
input[type=radio] {
   appearance: none;
   margin: 0;
}
input[type=radio] + label::before {
   content: "\00a0";  /* Space */
   border-radius: 10px;
   display: inline-block;
   width: 1em;  height: 1em;
   text-align: center;
   line-height: 1em;
   font-size: 0.8em;
   margin-right: 10px;
   background-color: #aaa;
   border: 0.125em solid white;
}
input[type=radio]:checked + label::before {
   content: "\2022";  /* Bullet */
   color: white;
   background: green;
}
input[type=radio]:focus + label::before {
   box-shadow: 0 0 0 1px #888;
}
```

```
<fieldset>
  <legend>Size:</legend>
  <div>
    <input id="small" type="radio" name="si
    <label for="small">Small</label>
  </div>
  <div>
    <input id="medium" type="radio" name="s
    <label for="medium">Medium</label>
  </div>
  <div>
    <input id="large" type="radio" name="si
    <label for="large">Large</label>
  </div>
</fieldset>
```



Animation content:

There are two blocks of code. The CSS:

```
input[type=radio] + label {
  cursor: pointer;
}
input[type=radio] {
  position: absolute;
  height: 1px; width: 1px;
  clip: rect(0 0 1 1);
}
input[type=radio] + label::before {
  content: "\00a0";  /* Non-breaking space */
  border-radius: 10px;
  display: inline-block;
  width: 1em;  height: 1em;
  text-align: center;
```

```
  line-height: 1em;
  font-size: 0.8em;
  margin-right: 10px;
  background-color: #aaa;
  border: 0.125em solid white;
}
input[type=radio]:checked + label::before {
  content: "\2022";  /* Bullet */
  color: white;
  background: green;
}
input[type=radio]:focus + label::before {
  box-shadow: 0 0 0 1px #888;
}
```

and the HTML:

```
<p>Size:</p>
<div>
 <input id="small" type="radio" name="size">
 <label for="small">Small</label>
</div>
<div>
 <input id="medium" type="radio" name="size">
 <label for="medium">Medium</label>
</div>
<div>
 <input id="large" type="radio" name="size">
 <label for="large">Large</label>
</div>
```

A screen shows three radio buttons labeled Small, Medium, and Large. cursor: pointer makes the cursor change when the user hovers over the radio button or the label. The input[type=radio] block positions the radio buttons off of the screen. To create custom radio buttons, add an empty space before the labels with a circular border, this is shown in the input[type=radio] + label::before CSS block. The input[type=radio]:checked + label::before CSS block changes the custom button to green empty space with a white dot in the middle when clicked. When a radio button has the focus, input[type=radio]:focus + label::before shows a grey shadow around the empty space.

Animation captions:

1. The default radio buttons are plain looking and cannot be styled.

2. The + selector selects <label> elements that are immediately after <input type="radio"> elements.
3. Changing the label's cursor to a pointer helps the user know the label is clickable.
4. Changing setting appearance to none hides the radio buttons, but screen readers still "see" the radio buttons.
5. ::before inserts the content \00a0, a non-breaking space, before the radio button label's content.
6. Custom radio buttons are created by displaying a single empty space within a circular border.
7. When a radio button is checked, a white bullet with green background replaces the previous content (the space).
8. When a radio button has the focus, a gray shadow displays around the content before the label.

PARTICIPATION ACTIVITY 5.6.6: Style the checkboxes.

The webpage below displays three styled radio buttons followed by four unstyled checkboxes. Add the following CSS to style the checkboxes:

1. Add a rule that adds a pointer cursor to the checkboxes' labels:

```
input[type=checkbox] + label {
   cursor: pointer;
   font-size: 1.3em;
}
```

   Render the webpage and verify that the cursor changes to a pointer when mousing over the checkbox labels.

2. Add a rule that hides the checkboxes:

```
input[type=checkbox] {
   appearance: none;
   margin: 0;
}
```

   Render the webpage and verify the default checkboxes are no longer visible.

3. Add a rule that adds a gray square in front of the checkbox labels:

```
input[type=checkbox] + label::before {
   content: "\00a0";   /* Non-breaking space */
   display: inline-block;
   width: 16px;
   height: 16px;
   text-align: center;
   line-height: 1em;
   font-size: 0.8em;
   margin-right: 10px;
   border: 0.125em solid white;   /* Push focus shadow away */
   background-color: #aaa;
}
```

Render the webpage and verify the checkboxes appear as squares.

4. Add a rule to display a white checkmark with green background when a checkbox is checked:

```
input[type=checkbox]:checked + label::before {
   content: "\2713";  /* Checkmark */
   color: white;
   background: green;
}
```

Render the webpage and verify that clicking on a checkbox displays a checkmark in the box.

5. Add a rule to display a gray border around the checkbox that has the focus:

```
input[type=checkbox]:focus + label::before {
   box-shadow: 0 0 0 1px gray;
}
```

Render the webpage and verify that the last clicked checkbox has a gray border.

HTML CSS

```
<fieldset>
   <legend>Size:</legend>
   <div>
      <input id="small" type="radio" name="size" value="small">
      <label for="small">Small</label>
   </div>
   <div>
      <input id="medium" type="radio" name="size" value="medium">
      <label for="medium">Medium</label>
   </div>
   <div>
      <input id="large" type="radio" name="size" value="large">
      <label for="large">Large</label>
   </div>
</fieldset>
```

Render webpage Reset code

Your webpage

Size:
- Small
- Medium
- Large

Flavors:
- Raspberry
- Grape
- Cherry
- Root Beer

Expected webpage

Size:
- Small
- Medium
- Large

Flavors:
- Raspberry
- Grape
- Cherry
- Root Beer

▶ View solution

PARTICIPATION ACTIVITY 5.6.7: Styling radio buttons and checkboxes.

1) Which label is selected by `input[type=radio] + label`?

   - `<input type="checkbox"> <label></label>`
   - `<input type="radio"> <label></label>`
   - `<label></label><input type="radio">`

2) In the exercise above, the default radio buttons and checkboxes could have been hidden using the CSS `display:none`. Why is hiding the radio buttons and checkboxes with `display:none` not a good idea?

   - Cross-browser support issues.
   - Screen readers will think the radio buttons or checkboxes are not visible.
   - The `display` property should be avoided.

3) Which CSS selector selects only checkboxes that are checked?

   - `input:checked`
   - `input[type=checkbox]:focus`
   - `input[type=checkbox]:checked`

4) What does the DOM look like after the CSS and HTML below are rendered?

```
span::before {
   content: "Before";
}
```

```
<span>Test</span>
```

- BeforeTest
- BeforeTest
- TestBefore

CHALLENGE ACTIVITY 5.6.1: Styling forms.

550544.4142762.qx3zqy7

Start

For the <label> tag, set display to inline-block, use a width of 45px, align the text to the right, and add a margin on the right of 7px. **SHOW EXPECTED**

CSS HTML

```
label {

   /* Your solution goes here */

}
form {
   font: 10pt Arial;
   background-color: #eee;
   padding: 10px;
}
```

1 2 3 4

Check Next

View your last submission

Exploring further:

- CSS Forms from W3Schools
- An Extensive Guide To Web Form Usability from Smash Magazine
- Placeholders in Form Fields Are Harmful from Nielsen Norman Group
- Replacing Radio Buttons Without Replacing Radio Buttons from SitePoint

5.7 Sass

CSS preprocessors

Sass is a popular CSS preprocessor that uses CSS-like syntax to build complex CSS stylesheets. Other popular CSS preprocessors, like Less and Stylus, offer similar and unique features with different syntax.

The Sass website has instructions on installing the Sass preprocessor on a variety of operating systems. Some developers prefer to run the Sass preprocessor from the command line or from an application like Koala. The Sass preprocessor compiles a Sass file (.scss) into a CSS (.css) file.

Sass version 3 introduced a new syntax called ***Sassy CSS*** (***SCSS***), which uses semicolons and brackets like CSS. Some online references still refer to the old Sass syntax which relies on indentation and has no brackets.

PARTICIPATION ACTIVITY 5.7.1: Compiling SCSS into CSS.

Animation content:

There are two blocks of code. The SCSS:
$font-face: Arial, sans-serif;
$font-color: #0ff;

```
body {
  font: 12pt $font-face;
  color: $font-color;
}
```

called styles.scss and the CSS

```
 body {
  font: 12pt Arial, sans-serif;
  color: #0ff;
}
```

called styles.css.

The command-line tool sass styles.scss styles.css compiles styles.scss into styles.css. During compilation the SCSS variables that begin with $ are inserted into the CSS.

Animation captions:

1. A .scss file contains SCSS syntax.
2. The sass command-line tool compiles styles.scss and outputs the resulting CSS to styles.css.
3. Variables begin with a $ and are set like CSS properties.
4. The value of the variables $font-face and $font-color are inserted into the resulting CSS.

PARTICIPATION ACTIVITY 5.7.2: Sass CSS preprocessor.

1) SCSS is syntactically different than the original Sass syntax.

   - True
   - False

2) The sass command-line tool creates a .scss file from a .css file.

   - True
   - False

3) The SCSS below results in CSS that sets a webpage's background color to blue.

```
$theme-color: blue;
body {
  background-color: $theme-
color;
}
```

- True
- False

4) An advantage to using an SCSS variable to store a color value used multiple times in a stylesheet is that if the color needs to be changed in the future, only the variable needs to be changed.

- True
- False

Nesting

Selectors may be nested in Sass to create child selectors that only apply to the parent selector. In the figure below, the `strong` child selector is nested in a `.notes` parent selector, creating a `.notes strong` selector in the resulting CSS.

Figure 5.7.1: Selector nesting.

```
// SCSS

.notes {
  font-size: smaller;

  strong {
    color: green;
  }
}
```

```
/* Resulting CSS */

.notes {
  font-size: smaller;
}

.notes strong {
  color: green;
}
```

The `&` character is used to reference the parent selector from a child selector's properties.

Figure 5.7.2: Referencing the parent with &.

```
// SCSS

a {
  text-decoration: none;
  &:hover {
    color: blue;
  }
}
```

```
/* Resulting CSS */

a {
  text-decoration: none;
}
a:hover {
  color: blue;
}
```

A number of CSS properties begin with the same prefix. Ex: `font-family`, `font-size`, and `font-weight` all begin with the same `font` prefix. Sass allows properties that share the same prefix to be nested under the prefix.

Figure 5.7.3: Property nesting.

```
// SCSS

p {
  font: {
    family: Arial;
    size: 12pt;
    weight: bold;
  }
}
```

```
/* Resulting CSS */

p {
  font-family: Arial;
  font-size: 12pt;
  font-weight: bold;
}
```

PARTICIPATION ACTIVITY 5.7.3: Nested selectors and properties.

Select the CSS that results from the given SCSS.

1)

```
p {
  em {
    color: red;
  }
}
```

- ```
  p em {
    color: red;
  }
  ```
- ```
  p + em {
    color: red;
  }
  ```
- ```
  p, em {
    color: red;
  }
  ```

2)

```
section {
  article {
    &:hover {
      color: blue;
    }
  }
}
```

- ```
  section:hover {
    color: blue;
  }
  ```
- ```
  section hover {
    color: blue;
  }
  ```
- ```
  section article:hover
  {
    color: blue;
  }
  ```

3)

```
.highlight {
  border: {
    width: 2px;
  }
}
```

```
.highlight {
  border: 2px;
}
```

```
.highlight {
  border-width: 2px;
}
```

```
.highlight border {
    width: 2px;
}
```

Variables and arithmetic

SassScript is a set of extensions to CSS that allow properties to use variables, arithmetic, and functions. SassScript also provides basic control directives for performing conditional logic and looping.

A SassScript variable begins with a `$` and can store one of the following data types:

- Number - Any number that is optionally followed by a CSS unit. Ex: `3`, `5.1`, `20px`
- String - "Double", 'single', or unquoted strings. Ex: `"red"`, `'red'`, `red`
- Color - Color name or value. Ex: `green`, `#00ff00`, `rgb(0,255,0)`
- Boolean - `true` or `false`
- Null - `null`
- List of values - Separated by spaces or commas. Ex: `10px 20px 30px 40px`, `Helvetica, Arial, sans-serif`
- Map - Key/value pairs. Ex: `(111:red, 222:blue)`

Basic arithmetic like addition, subtraction, multiplication, and division may be performed on numbers and numeric variables. Ex: `20px + 15` = `35px`. Arithmetic on color values results in the red, green, and blue values being added, subtracted, multiplied, or divided one at a time. Ex: `#0011aa + #bb2244` results in 00 + bb = bb, 11 + 22 = 33, and aa + 44 = ee; so the final value is `#bb33ee`.

PARTICIPATION ACTIVITY 5.7.4: Performing SassScript arithmetic.

SCSS

```
$width: 300;
$size: 20pt;
$color: #220055;

p {
  width:  300  - 50px;
  font-size:  20pt  * 0.9;
  line-height:  20pt  / 2 + 14;
  color:#220055 * 2;
}
```

CSS

```
p {
  width: 250px;
  font-size: 18pt;
  line-height: 24pt;
  color: #4400aa
}
```

Animation content:

There are two blocks of code. The SCSS:
$width: 300;
$size: 20pt;
$color: #220055;
p {
  width: $width - 50px;
  font-size: $size * 0.9;
  line-height: $size / 2 + 14
  color: $color * 2;
}
and the CSS
p {
  width: 250px;
  font-size: 18pt;
  line-height: 24pt;
  color: #4400aa;
}
When the SCSS is compiled to the CSS, the calculations for properties are performed. The width property in CSS is calculated by $width - 50px or 300 - 50px to get 250px. The font-size property

in CSS is calculated by multiplying $size * 0.9 or 20pt * 0.9 to get 18pt. The line-height property in CSS is calculated by $size / 2 + 14 or 20pt / 2 + 14 to get 24pt. The color property in CSS is calculated by multiplying $color * 2 or #220055 * 2 to get #4400aa. The multiplication is done by multiplying each color value by 2, to get red 2 * 22, to get green 2 * 00, and to get blue 2 * 55.

Animation captions:

1. $width and $size store numbers, but $color stores a color.
2. 50px is subtracted from the value of the $width variable.
3. The variable $size is multiplied by 0.9.
4. $size is divided by 2, and then 14 is added to the result.
5. Multiplying $color by 2 results in red, green, and blue components each being multiplied by 2.

PARTICIPATION ACTIVITY 5.7.5: Variables and arithmetic.

What is `$value`?

1)
```
$value: 20px - 15;
```

Check Show answer

2)
```
$value: 20pt + (10 / 2) ;
```

Check Show answer

3)
```
$value: #ff1150 -
#001120;
```

Check Show answer

4) `$value: #ff1150 + 2;`

Check Show answer

Functions

SassScript includes a large number of utility functions.

Table 5.7.1: Some SassScript functions.

Function	Description	Example
`lighten(color, amount)`	Returns a `color` lightened by an `amount` between 0% and 100%	`/* Returns #c00 */` `$color: lighten(#a00, 10%);`
`invert(color)`	Returns the inverse (negative) of a `color`	`/* Returns #5ff */` `$color: invert(#a00);`
`to-upper-case(string)`	Returns `string` using all uppercase characters.	`/* Returns "BEHOLD!" */` `$message: to-upper-case("Behold!");`
`round(number)`	Returns a `number` rounded to the nearest whole number	`/* Returns 21px */` `$width: round(20.5px);`
`random(limit)`	Returns a random integer between 1 and `limit` (inclusive)	`/* Returns a number between 1 and 5 that is multiplied by 20px */` `$width: random(5) * 20px;`

PARTICIPATION ACTIVITY 5.7.6: SassScript functions.

What is `$value`?

1) `$value: lighten(black, 20%);`
   - #000
   - white
   - #333

2) `$value: invert(white);`
   - black
   - white
   - gray

3) `$value: round(16.4pt);`
   - 16.4pt
   - 16pt
   - 17pt

4) `$value: random(3) * 100px;`
   - 0, 1, 2, or 3
   - 1, 2, or 3
   - 100px, 200px, or 300px

Mixins

A ***mixin*** is set of reusable styles and is defined by the ***@mixin*** directive. A ***directive*** is an extension to the CSS at-rules, which are statements that begin with the `@` character. Mixins may take arguments, which give mixins the ability to customize the styles that the mixin defines. Mixins are included in a document using the ***@include*** directive.

PARTICIPATION ACTIVITY 5.7.7: Including mixins.

SCSS

```
@mixin cool-font {
  font: italic small-caps 16pt cursive;
  color: darkgreen;
}

@mixin highlight($color, $width) {
  border: $width solid $color;
  box-shadow: 4px darken($color, 20%);
}

.special {
  background-color: #eee;
  @include cool-font;
  @include highlight(red, 2px);
}
```

CSS

```
.special {
  background-color: #eee;
  font: italic small-caps 16pt cursive;
  color: darkgreen;
  border: 2px solid red;
  box-shadow: 4px #990000;
}
```

Animation content:

There are two blocks of code. The SCSS:
@mixin cool-font {
  font: italic small-caps 16pt cursive;
  color: darkgreen;
}
@mixin highlight($color, $width) {
  border: $width solid $color;
  box-shadow: 4px darken($color, 20%);
}
.special {
  background-color: #eee;
  @include cool-font;
  @include highlight(red, 2px);
}
and the CSS:
.special {
  background-color: #eee;
  font: italic small-caps 16pt cursive;
  color: darkgreen;
  border: 2px solid red;
  box-shadow: 4px #990000;

}

When the SCSS is compiled to the CSS, cool-font's attributes are transferred directly to the CSS, and highlight's attributes are first filled with the two arguments then transferred to the CSS.

Animation captions:

1. Two mixins are defined: cool-font and highlight.
2. The special class includes the cool-font mixin.
3. The highlight mixin is included with two arguments, red and 2px, which are assigned to $color and $width.

PARTICIPATION ACTIVITY 5.7.8: Mixins.

Given the mixins below, match the SCSS with the resulting CSS.

```
@mixin shadow-font {
  font-size: 12pt;
  text-shadow: 2px 2px blue;
}

@mixin pretty-border($img, $size) {
  border: 10px solid transparent;
  padding: 20px;
  border-image: url($img) $size round;
}
```

If unable to drag and drop, refresh the page.

```
div {
  @include pretty-border(
          "border.png", 30);
}
```

```
div {
  @include shadow-font;
}
ol {
  @include shadow-font;
}
```

```
div {
  @include shadow-font;
}
```

```
div {
  @include pretty-border(
          "border.png", 30);
  @include shadow-font;
}
```

```
div {
  font-size: 12pt;
  text-shadow: 2px 2px blue;
}
```

```
div {
  border: 10px solid transparent;
  padding: 20px;
  border-image: url("border.png") 30 round
}
```

```
div {
  border: 10px solid transparent;
  padding: 20px;
  border-image: url("border.png") 30 round
  font-size: 12pt;
  text-shadow: 2px 2px blue;
}
```

```
div {
  font-size: 12pt;
  text-shadow: 2px 2px blue;
}
ol {
  font-size: 12pt;
  text-shadow: 2px 2px blue;
}
```

Reset

Control directives and expressions

Sass contains other features including:

- *Control directives, like* `@if` *and* `@for`*, that support conditional styling and looping*
- *Ability to import SCSS and Sass files using the* `@import` *directive*
- *Ability to extend the styles in a class with the* `@extend` *directive*
- *Ability to write custom functions*

See the Sass website's documentation for more details.

Exploring further:

- Sass
- Koala
- Less
- Stylus

5.8 Example: Styled Restaurant Reviews

Styling the home page

This section adds CSS to the example restaurant review website from an earlier section. The restaurant review website displays user reviews of local restaurants.

Home page.

The index.html file is the home page, and the external stylesheet styles.css is used by all the website's pages to present a consistent look. Some special characteristics of the HTML and CSS below include:

- A subtle gray text shadow on the "Restaurant Reviews" header, produced with the CSS `text-shadow` property.
- The use of an unordered list to create the navigation links. The bullets are removed from the list with the CSS `list-style-type:none`, and the `<li>` elements are displayed side-by-side by changing the `<li>` elements to inline elements with `display:inline`.
- Featured restaurant sections with a rounded border, produced using the CSS `border-radius` property.
- A flexbox to put the stars and description to the right of the restaurant photo.

Images from Wikimedia.org

index.html | styles.css

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Restaurant Reviews</title>
      <link rel="stylesheet" href="styles.css">
   </head>
   <body>
      <header>
         <h1>Restaurant Reviews</h1>
         <nav>
            <ul>
               <li><a href="index.html">Home</a></li>
               <li><a href="addreview.html">Add Review</a></li>
               <li><a href="about.html">About</a></li>
            </ul>
```

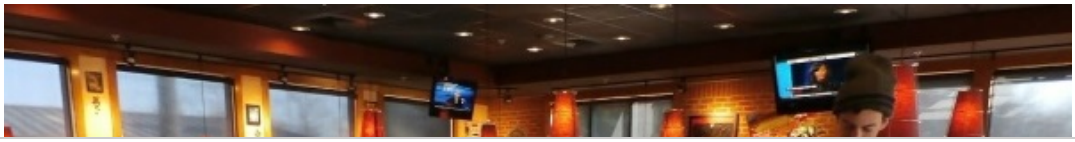
Render webpage Reset code

Your webpage

Restaurant Reviews

Home Add Review About

Main Street Cafe



PARTICIPATION ACTIVITY 5.8.1: Home page.

1) Increasing the first two numbers in the `text-shadow` property value for the header moves the text shadow further away from the text.

   - True
   - False

2) The link text to Main Street Cafe and Greek House appears red when the mouse hovers on top of the links.

   - True
   - False

3) The navigation link text appears red when the mouse hovers on top of the links.

   - True
   - False

4) Adding `border-radius:10px;` to the `nav li` rule adds rounded corners to the red rectangle that appears around navigation links.

- True
- False

5) The flex container has 3 flex items.

- True
- False

6) Changing the `flex-direction` property to `row-reverse` puts the restaurant photo on the right side of the stars and description.

- True
- False

Show an advertisement

The home page may be modified to display an advertisement in the upper-right corner by using fixed positioning. To draw the user's attention to the advertisement, some CSS animations can drop the advertisement onto the webpage after a short delay. The `animation-*` properties in the figure below use the keyframe `lowerAd` to lower the advertisement onto the screen by changing the advertisement's `top` property from -80px to 40px over a 1 second duration. The `animation-fill-mode: forwards` declaration makes the advertisement retain the `top:40px` value from the animation.

Figure 5.8.1: HTML and CSS for the advertisement.

```
<div id="advertisement">
   <a href="about.html">Add your advertisement here!
</a>
</div>
```

```
@keyframes lowerAd {
   from { top: -80px; }
   to { top: 40px; }
}

#advertisement {
   animation-name: lowerAd;
   animation-duration: 1s;
   animation-delay: 2s;
   animation-fill-mode: forwards;
   position: fixed;
   right: 30px;
   top: -80px;
   width: 100px;
   background: linear-gradient(lightgreen, white);
   box-shadow: 5px 5px;
   padding: 10px;
   border-radius: 4px;
}
```

The advertisement can expand in size when the mouse is hovering over the advertisement by using the CSS `transform` property to scale the advertisement larger. The CSS `transition` property animates the expanding advertisement over 200 ms.

Figure 5.8.2: CSS for expanding advertisement.

```
#advertisement {
   animation-name: lowerAd;
   animation-duration: 1s;
   animation-delay: 2s;
   animation-fill-mode: forwards;
   position: fixed;
   right: 30px;
   top: -80px;
   width: 100px;
   background: linear-gradient(lightgreen,
white);
   box-shadow: 5px 5px;
   padding: 10px;
   border-radius: 4px;
   transition: all 200ms ease-in-out;
}

#advertisement:hover {
   transform: scale(1.2, 1.2);
}
```

Animated advertisement.

After the page is rendered, the advertisement falls onto the webpage after a two second delay. Press the "Render webpage" button to reset the animation. Place the mouse on top of the advertisement to see the advertisement grow.

index.html styles.css

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Restaurant Reviews</title>
      <link rel="stylesheet" href="styles.css">
   </head>
   <body>
      <header>
         <h1>Restaurant Reviews</h1>
         <nav>
            <ul>
               <li><a href="index.html">Home</a></li>
               <li><a href="addreview.html">Add Review</a></li>
               <li><a href="about.html">About</a></li>
            </ul>
```

Render webpage Reset code

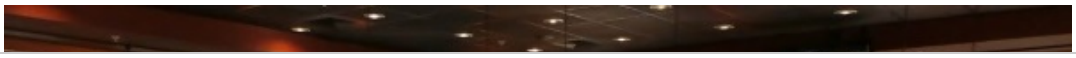
Your webpage

Restaurant Reviews

Home Add Review About

You've come to the right place to find a great restaurant near you!

Main Street Cafe



PARTICIPATION ACTIVITY 5.8.2: Advertisement.

1) Which CSS property should be modified to make the advertisement drop immediately when the page is rendered?

- `animation-name`
- `animation-duration`
- `animation-delay`

2) What CSS property is needed to make the advertisement slide in from the right?

```
@keyframes lowerAd {
   from { _____: -100px; top:
40px; }
   to { _____: 30px; top:
40px; }
}
```

- `left`
- `right`
- `bottom`

3) What CSS makes the advertisement scale and tilt right when the mouse hovers over the ad?

```
#advertisement:hover {
   transform: scale(1.2, 1.2)
____;
}
```

- `translate(-5px, 5px)`
- `scaleZ(-1.5)`
- `rotate(5deg)`

Styling the detail pages

The detail page for each restaurant displays additional restaurant details and user reviews. Two CSS rules are added to styles.css to place the overall rating stars on top of the restaurant photo:

- `.restaurant-photo` - Adds relative positioning to the section that displays the restaurant photo and overall rating stars
- `.stars` - Positions overall rating stars on top of the restaurant photo with absolute positioning

Figure 5.8.3: CSS for styling the reviews and overall rating stars.

```
.restaurant-photo {
   position:
relative;
}

.stars {
   position:
absolute;
   top: 220px;
   left: 10px;
   width: 180px;
}
```

The user reviews are layed out with a grid layout. The `.review` rule creates the grid container with 2 columns and lays out the grid using named grid items. The grid items are assigned names in the classes `.review-name`, `.review-date`, `.review-rating`, and `.review-details`.

Figure 5.8.4: Orange grid lines show the grid layout for each review.

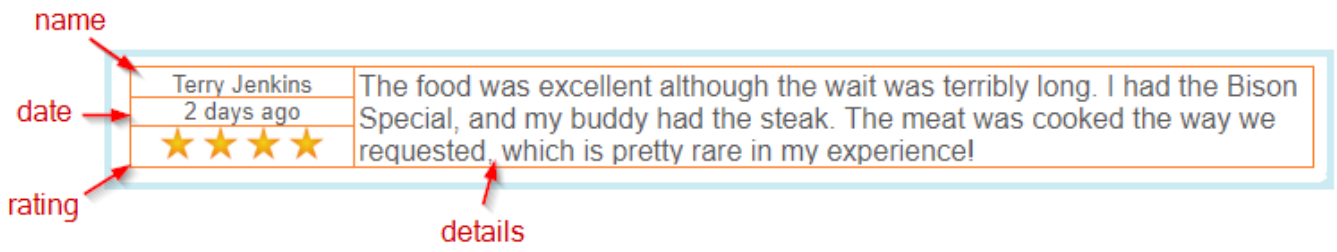


Figure 5.8.5: CSS for the user rating grid layout.

```
.review {
   display: grid;
   grid-template-columns: 130px
auto;
   grid-template-areas:
      "name details"
      "date details"
      "rating details";
   border-radius: 5px;
   background: white;
   padding: 8px 8px;
   margin: 10px 0;
   font-size: smaller;
}

.review-name {
   grid-area: name;
   text-align: center;
}

.review-date {
   grid-area: date;
   text-align: center;
}

.review-rating {
   grid-area: rating;
   text-align: center;
}

.review-rating > img {
   width: 20px;
}

.review-details {
   grid-area: details;
   font-size: larger;
}
```

Restaurant review details.

mainstreetcafe.html | styles.css

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Restaurant Reviews: Main Street Cafe</title>
      <link rel="stylesheet" href="styles.css">
   </head>
   <body>
      <header>
         <h1>Restaurant Reviews</h1>
         <nav>
            <ul>
               <li><a href="index.html">Home</a></li>
               <li><a href="addreview.html">Add Review</a></li>
               <li><a href="about.html">About</a></li>
            </ul>
```

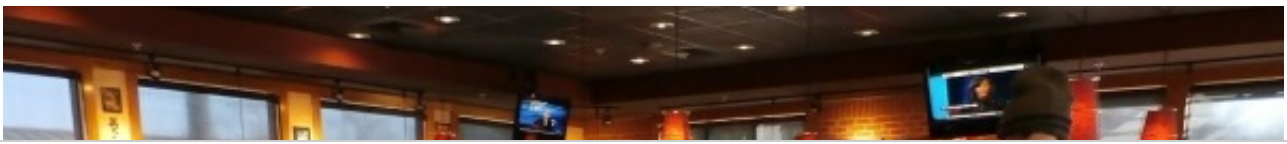
Render webpage Reset code

Your webpage

Restaurant Reviews

Home Add Review About

Main Street Cafe



PARTICIPATION ACTIVITY 5.8.3: Detail page.

1) The reviews would have a blue background if `background:white` was removed from the `.review` rule.
   - True
   - False

2) The `top` and `left` values in the `.stars` rule position the stars relative to the browser's top and left edges.
   - True
   - False

3) Changing the `.stars` rules declaration from `position:absolute` to `position:fixed` changes nothing about the rendered webpage.
   - True
   - False

4) The star images only appear on top of the restaurant image because the star images are defined later in the HTML.
   - True
   - False

5) The grid layout used to lay out the user reviews has 2 total grid items.
   - True
   - False

6) Which CSS property must be modified to change the number of rows the review detail spans?

- grid-template-columns
- grid-template-areas

Styling the form

The form on the addreview.html page is improved by fixing the width of the form labels and adding padding, sizing, and better borders to the form widgets.

PARTICIPATION ACTIVITY 5.8.4: Styling the form.

Match the letter to the missing CSS.

```
label {
   width: 100px;
   display: __A__;
   text-align: right;
   margin-right: 8px;
}

input, select, textarea{
   width: 350px;
   padding: 10px;
   border: 1px solid #aaa;
   border-radius: 4px;
   box-sizing: __B__;
}

input[__C__], input[type=date] {
   width: auto;
}

button {
   width: 100px;
   background-color: var(--accent-color);
   color: white;
   padding: 15px;
   margin: 10px 0;
   border: none;
   border-radius: 4px;
   cursor: __D__;
}

button:hover {
   background-color: var(--accent-dark-color);
}
```

If unable to drag and drop, refresh the page.

D A B C

---

	pointer
	border-box
	inline-block
	type=number

Reset

Styling the form.

addreview.html | styles.css

```
1  <!DOCTYPE html>
2  <html lang="en">
3     <head>
4        <meta charset="UTF-8">
5        <title>Restaurant Reviews - Add Review</title>
6        <link rel="stylesheet" href="styles.css">
7     </head>
8     <body>
9        <header>
10          <h1>Restaurant Reviews</h1>
11          <nav>
12             <ul>
13                <li><a href="index.html">Home</a></li>
14                <li><a href="addreview.html">Add Review</a></li>
15                <li><a href="about.html">About</a></li>
16             </ul>
```

Render webpage Reset code

Your webpage

5.9 LAB: CSS practice

Download the ZIP file below containing an HTML file. Modify the HTML file to produce a webpage for the following poem/song, entitled "Yesterday":

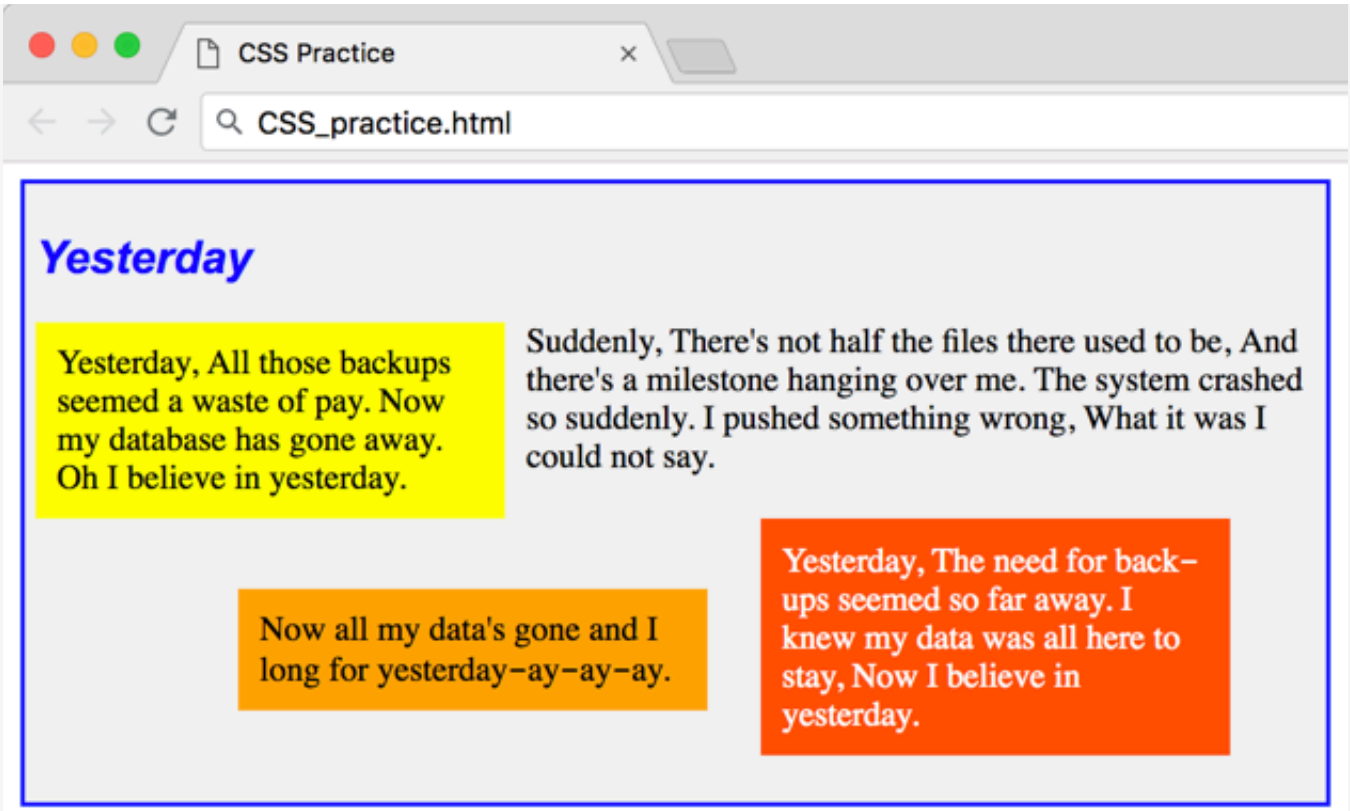
```
Yesterday, All those backups seemed a waste of pay. Now my database
has gone away. Oh I believe in yesterday.

Suddenly, There's not half the files there used to be, And there's a
milestone hanging over me. The system crashed so suddenly. I pushed
something wrong, What it was I could not say.

Now all my data's gone and I long for yesterday-ay-ay-ay.

Yesterday, The need for back-ups seemed so far away. I knew my data
was all here to stay, Now I believe in yesterday.
```

Use CSS to format the webpage so it looks exactly like the screenshot below:



Requirements

1. Use either an embedded stylesheet or inline styles for all formatting.
2. Use a for each of the 4 verse containers and the outer container.
3. The title “Yesterday” uses an <h2> tag with a 16pt Arial font that is blue and italicized. Do not use an <i> tag.
4. The entire song is embedded in a container that is 600 x 280 pixels, has a two pixel solid blue border, and a five pixel padding. The background color is #F0F0F0.
5. The outer 's first child is the <h2>. The 4 child s follow, declared in the order listed below.
6. The yellow section:
    - has a background color of "yellow"
    - floats to the left
    - has a 200 pixel width
    - has a 10 pixel padding
    - has a right margin of 10 pixels, with all other margins set to 0
7. The section with no background color has no CSS styling.
8. The orange section:

- has a background color of "orange"
- is positioned at (110px, 200px) with absolute positioning
- has a 200 pixel width
- has a 10 pixel padding

9. The orange-red section:
   - has a background color of "rgb(255, 80, 0)"
   - has white text
   - floats to the right
   - has a 200 pixel width
   - has a 10 pixel padding
   - has a right margin of 40 pixels, a top margin of 20 pixels, and all other margins set to 0
10. Do not use
 anywhere in your HTML.

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.9.1: LAB: CSS practice | 10 / 10

Submission Instructions

Downloadable files

`index.html` **Download**

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

index.html

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work [What is this?](#)

```
4/13 S10 min:3
```

Latest submission - 11:31 AM PDT Submission passed Total score: 10

on 04/13/24 all tests / 10

Only show failing tests **Download this submission**

1:Unit test 1 / 1

Main has proper styles and no
 tags are used (1 point)

- ✓ At least 1 exists on the page
- ✓ Outer container has correct background color
- ✓ Outer container has solid border
- ✓ Outer container has border width of 2 pixels
- ✓ Outer container has blue border
- ✓ Outer container has solid border
- ✓ Outer container has border width of 2 pixels
- ✓ Outer container has blue border
- ✓ Outer container has solid border
- ✓ Outer container has border width of 2 pixels
- ✓ Outer container has blue border
- ✓ Outer container has solid border
- ✓ Outer container has border width of 2 pixels
- ✓ Outer container has blue border
- ✓ Outer container has width of 600 pixels
- ✓ Outer container has height of 280 pixels
- ✓ Outer container has left padding of 5 pixels

- ✓ Outer container has top padding of 5 pixels
- ✓ Outer container has right padding of 5 pixels
- ✓ Outer container has bottom padding of 5 pixels
- ✓ No
 tags exist on the page

2:Unit test 1 / 1

Main contains <h2> with proper styles (1 point)

- ✓ At least 1 exists on the page
- ✓ Main contains a single <h2>
- ✓ <h2> has correct font color
- ✓ <h2> has italic font style

3:Unit test 2 / 2

Yellow has proper styles (2 points)

- ✓ At least 1 exists on the page
- ✓ Yellow container contains expected verse text
- ✓ Yellow container has correct background color
- ✓ Yellow container has 10 pixel left padding
- ✓ Yellow container has 10 pixel top padding
- ✓ Yellow container has 10 pixel right padding
- ✓ Yellow container has 10 pixel bottom padding
- ✓ Yellow container floats to the left
- ✓ Yellow container has 200 pixel width

- ✓ Yellow container has 10 pixel right margin
- ✓ Yellow container has 0 pixel left margin
- ✓ Yellow container has 0 pixel top margin
- ✓ Yellow container has 0 pixel bottom margin

4:Unit test 2 / 2

No-color has proper styles (2 points)

- ✓ At least 1 exists on the page
- ✓ No-color container contains expected verse text
- ✓ No-color container has correct background color
- ✓ No-color container has default float style

5:Unit test 2 / 2

Orange has proper styles (2 points)

- ✓ At least 1 exists on the page
- ✓ Orange container contains expected verse text
- ✓ Orange container has correct background color
- ✓ Orange container uses absolute positioning
- ✓ Orange container's left position is 110 pixels
- ✓ Orange container's top position is 200 pixels
- ✓ Orange container has default float (none)
- ✓ Orange container has 200 pixel width
- ✓ Orange container has 10 pixel left padding

- ✓ Orange container has 10 pixel top padding
- ✓ Orange container has 10 pixel right padding
- ✓ Orange container has 10 pixel bottom padding

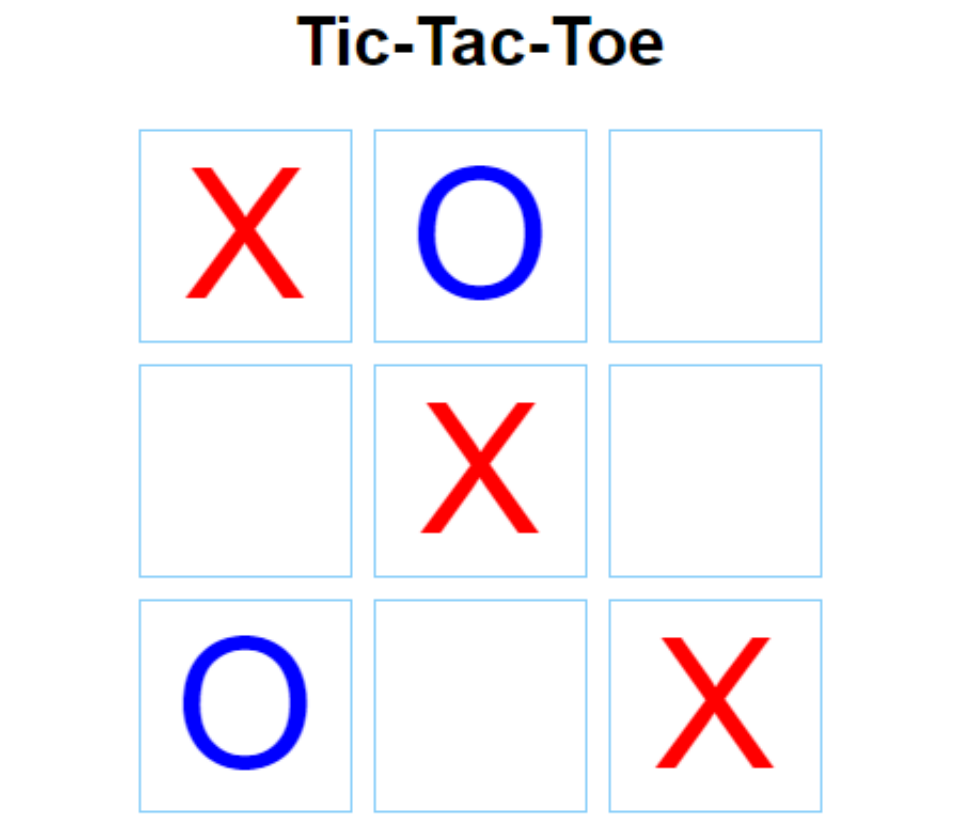
6:Unit test 2 / 2

Orange-red has proper styles (2 points)

- ✓ At least 1 exists on the page
- ✓ Orange-red container contains expected verse text
- ✓ Orange-red container has correct background color
- ✓ Orange-red container has correct text color
- ✓ Orange-red container floats right
- ✓ Orange-red container has 200 pixel width
- ✓ Orange-red container has 10 pixel left padding
- ✓ Orange-red container has 10 pixel top padding
- ✓ Orange-red container has 10 pixel right padding
- ✓ Orange-red container has 10 pixel bottom padding
- ✓ Orange-red container has 40 pixel right margin
- ✓ Orange-red container has 0 pixel left margin
- ✓ Orange-red container has 20 pixel top margin
- ✓ Orange-red container has 0 pixel bottom margin

5.10 LAB: Tic-tac-toe game with grid layout

Download the ZIP file below containing an HTML and CSS file. Modify styles.css so the webpage displays a tic-tac-toe board as shown below.



Add the following CSS to the `#board` selector in styles.css:

- A `display` property that makes the board's `<div>` element a grid layout container.
- A `grid-template-columns` property to create three columns that are all 100px wide.
- A `grid-template-rows` property to create three rows that are all 100px tall.
- A `grid-gap` property to create a 10px gap between each of the nine grid cells.
- A `justify-content` property that centers the grid horizontally.

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.10.1: LAB: Tic-tac-toe game with grid layout | 10 / 10

Submission Instructions

Downloadable files

`tic-tac-toe.html` and `styles.css` **Download**

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

styles.css

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work What is this?

```
4/13 S10 min:2
```

Latest submission - 11:59 AM PDT on 04/13/24 Submission passed all tests ✓ Total score: 10 / 10

☐ Only show failing tests **Download this submission**

1:Unit test ^ 2 / 2

Test display property

- ✓ #board selector exists
- ✓ display property of rule

2:Unit test ^ 2 / 2

Test grid-template-columns property

- ✓ #board selector exists
- ✓ grid-template-columns property of rule

3:Unit test 2 / 2

Test grid-template-rows property

- ✓ #board selector exists
- ✓ grid-template-rows property of rule

4:Unit test 2 / 2

Test grid-gap (column-gap and row-gap) property

- ✓ #board selector exists
- ✓ column-gap property of rule
- ✓ row-gap property of rule

5:Unit test 2 / 2

Test justify-content property

- ✓ #board selector exists
- ✓ justify-content property of rule

5.11 LAB: Position the playing cards

Download the ZIP file below containing an HTML and CSS file. Modify styles.css file so the webpage displays two playing cards on a green background.

Add the following CSS to styles.css:

- Add a `linear-gradient` background to the `.table` rule that goes from forestgreen to darkgreen.
- Add a `border-radius` property of 5px and a black `box-shadow` property with 3px x and y offsets to the `.card` rule.
- Add absolute positioning to the `.cardTop` rule, and position the card 150px from the left and 100px from the top of the container edge.
- Add absolute positioning to the `.cardBottom` rule, and position the card 240px from the left and 120px from the top of the container edge.
- Add a `z-index` property to the `.cardTop` and `.cardBottom` rules to make the top card appear on top.
- Add absolute positioning to the `.topLeft` selector to place the symbol 2px from the left and top edges of the card.
- Add absolute positioning to the `.bottomRight` selector to place the symbol 2px from the right and bottom edges of the card.

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.11.1: LAB: Position the playing cards | 10 / 10

Submission Instructions

Downloadable files

`cards.html` and `styles.css` **Download**

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

styles.css

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work [What is this?](#)

```
4/13 S9,10 min:3
```

Latest submission - 6:28 PM PDT on 04/13/24 — Submission passed all tests ✓ — Total score: 10 / 10

☐ Only show failing tests **Download this submission**

1:Unit test ^ 1 / 1

Test linear-gradient property

- ✓ .table selector exists
- ✓ background property of rule

2:Unit test ^ 1 / 1

Test border-radius property

- ✓ .card selector exists
- ✓ border-radius property of rule

3:Unit test 1 / 1

Test box-shadow property

- ✓ .card selector exists
- ✓ box-shadow property of rule is "3px 3px black"

4:Unit test 2 / 2

Test .cardTop rule

- ✓ .cardTop selector exists
- ✓ position property of rule
- ✓ left property of rule
- ✓ top property of rule

5:Unit test 2 / 2

Test .cardBottom rule

- ✓ .cardBottom selector exists
- ✓ position property of rule
- ✓ left property of rule
- ✓ top property of rule

6:Unit test 1 / 1

Test z-index property

- ✓ z-index in .cardTop > z-index in .cardBottom

7:Unit test 1 / 1

Test .topLeft rule

- ✓ .topLeft selector exists
- ✓ position property of rule
- ✓ left property of rule
- ✓ top property of rule

8:Unit test 1 / 1

Test .bottomRight rule

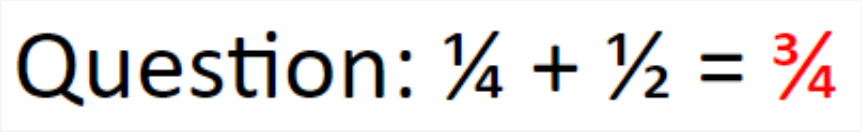
- ✓ .bottomRight selector exists
- ✓ position property of rule
- ✓ right property of rule
- ✓ bottom property of rule

Previous submissions

6:27 PM on 4/13/24	9 / 10	View

5.12 LAB: Animating the answer

Download the ZIP file below containing an HTML file. Modify the HTML file's embedded stylesheet to produce an animation, resulting in the webpage below.



Make the following modifications to the embedded stylesheet:

- Add a keyframe called `moveFraction` that does the following:

    - At 0% sets the font color to yellow and uses `transform` property to translate to position (-355px, 60px) so the fraction is off the left side of the screen.
    - At 50% translates to (0px, 60px) so the fraction is just below and right of the question.
    - At 100% sets font color to red and translates to (0px, 0px) so the fraction is located next to the = sign.
- In the `#answer` rule start the moveFraction animation 1 second after the CSS loads, and make the animation duration 2 seconds.
- In the `#answer` rule use `animation-fill-mode: forwards` to keep the fraction from resetting back to the original location after the animation completes.
- Add a `transition` property to the `#question` rule to perform a `transform` over 0.6 seconds with the `ease-in-out` timing function.
- Add a `transform` property to the `#question:hover` rule to scale by 0.9.

Verify that when the page loads, the 3/4 fraction moves from off the screen to the right, just under the question, then up next to the question. When the mouse hovers over the question, the text should shrink some until the mouse is moved off the text.

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.12.1: LAB: Animating the answer | 10 / 10

Submission Instructions

Downloadable files

`question.html` **Download**

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

question.html

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work [What is this?]

```
4/13 S6,10 min:4
```

Latest submission - 6:43 PM PDT on 04/13/24 — Submission passed all tests ✓ — Total score: 10 / 10

☐ Only show failing tests **Download this submission**

1:Unit test 1 / 1

Test moveFraction keyframes exists

- ✓ moveFraction keyframes selector exists
- ✓ moveFraction is a keyframes selector

2:Unit test 2 / 2

Test moveFraction 0% rule

- ✓ moveFraction keyframes selector exists
- ✓ moveFraction has at least 1 rule
- ✓ First rule is 0%
- ✓ color property of rule 0%
- ✓ transform property of rule 0%

3:Unit test 1 / 1

Test moveFraction 50% rule

- ✓ moveFraction keyframes selector exists
- ✓ moveFraction has at least 2 rules
- ✓ Second rule is 50%
- ✓ transform property of rule 50%

4:Unit test 2 / 2

Test moveFraction 100% rule

- ✓ moveFraction keyframes selector exists
- ✓ moveFraction has 3 rules
- ✓ Second rule is 100%
- ✓ color property of rule 100%
- ✓ transform property of rule 100%

5:Unit test 2 / 2

Test #question rule

- ✓ #question selector exists
- ✓ transition property of rule

6:Unit test 2 / 2

Test #question:hover selector

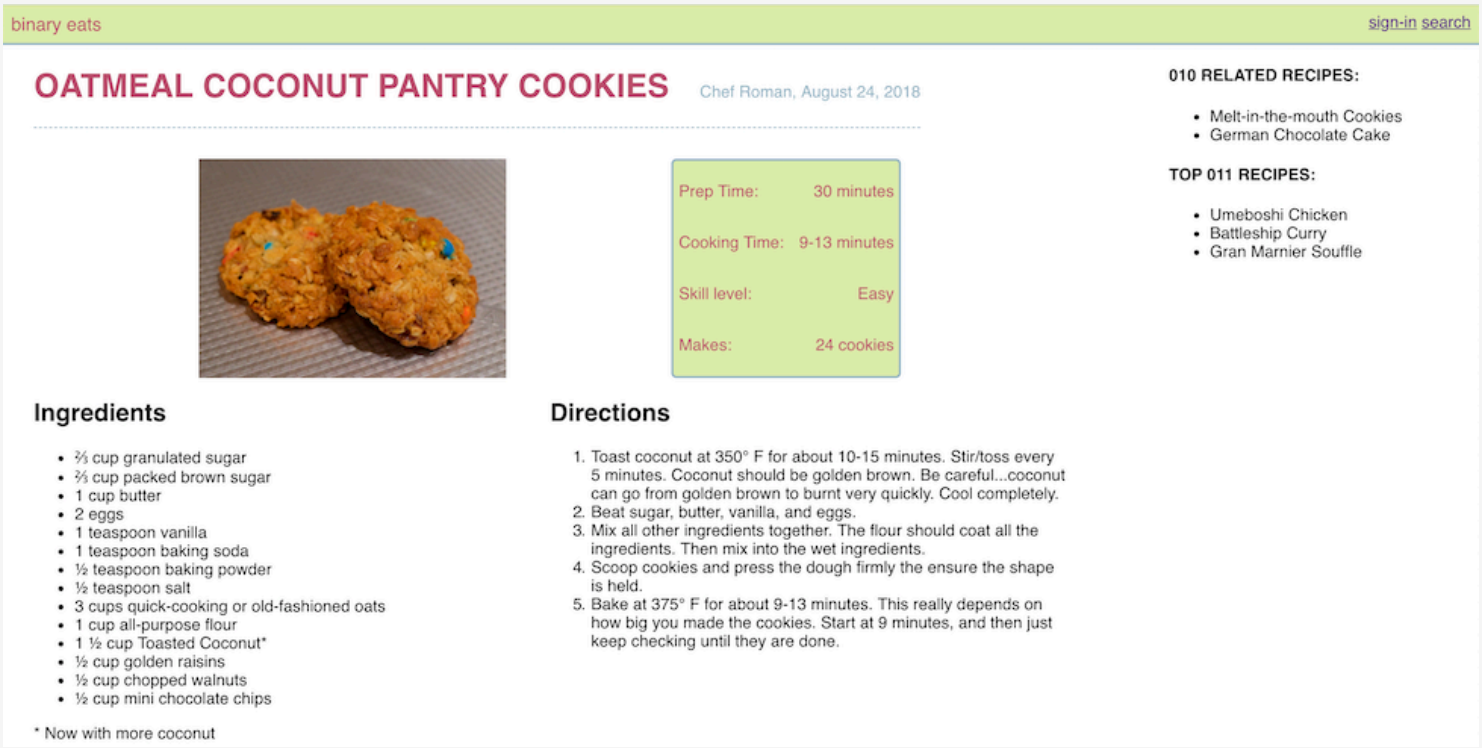
- ✓ #question:hover selector exists
- ✓ transform property of rule

Previous submissions

6:40 PM on 4/13/24	6 / 10	View

5.13 LAB: Recipe with flexbox

Download the ZIP file below containing HTML, CSS, and image files. Modify styles.css so the HTML produces the webpage below.



Add the following style rules to the stylesheet:

- `<header>` tag: set display to be flex and justify-content to be space-between
- `content` class: set display to be flex and flex-wrap to be wrap
- `main-content` class: set display to be flex, flex-wrap to be wrap, flex-grow to be 1, and flex-basis to be 70%
- `title` class: set display to be flex and align-items to be baseline
- `summary` class: set display to be flex, justify-content to be space-evenly, flex-grow to be 1, and flex-basis to be 100%
- `<div>` tag child of `<div>` tag child of `summary` class: set display to be flex and justify-content to be space-between
- `ingredients` and `directions` classes: set flex-grow to be 1 and flex-basis to be 45%
- `related-content` class: set flex-grow to be 1 and flex-basis to be 20%

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.13.1: LAB: Recipe with flexbox 10 / 10

Submission Instructions

Downloadable files

`oatmeal_cookies.html` , `styles.css` , and `oatmealCookie_small.jpg`

Download

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

styles.css

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work What is this?

```
4/13 S10 min:3
```

Latest submission - 11:47 AM PDT on 04/13/24 Submission passed all tests ✓ Total score: 10 / 10

☐ Only show failing tests Download this submission

1:QUnit test ^ 1 / 1

Testing flexbox styles of <header> tag

- ✓ display of <header> tag
- ✓ justify-content of <header> tag

2:QUnit test 1 / 1

Testing flexbox styles of content class

- ✓ display of content class
- ✓ flex-wrap of content class

3:QUnit test 2 / 2

Testing flexbox styles of main-content class

- ✓ display of main-content class
- ✓ flex-wrap of main-content class
- ✓ flex-grow of main-content class
- ✓ flex-basis of main-content class

4:QUnit test 1 / 1

Testing flexbox styles of title class

- ✓ display of title class
- ✓ align-items of title class

5:QUnit test 2 / 2

Testing flexbox styles of summary class

- ✓ display of summary class
- ✓ justify-content of summary class
- ✓ flex-grow of summary class
- ✓ flex-basis of summary class

6:QUnit test 1 / 1

Testing flexbox styles of tag child of tag child of summary class

- ✓ display of summary class > div > div
- ✓ justify-content of summary class > div > div

7:QUnit test 1 / 1

Testing flexbox styles of ingredients and directions classes

- ✓ flex-grow of ingredients class
- ✓ flex-grow of directions class
- ✓ flex-basis of ingredients class
- ✓ flex-basis of directions class

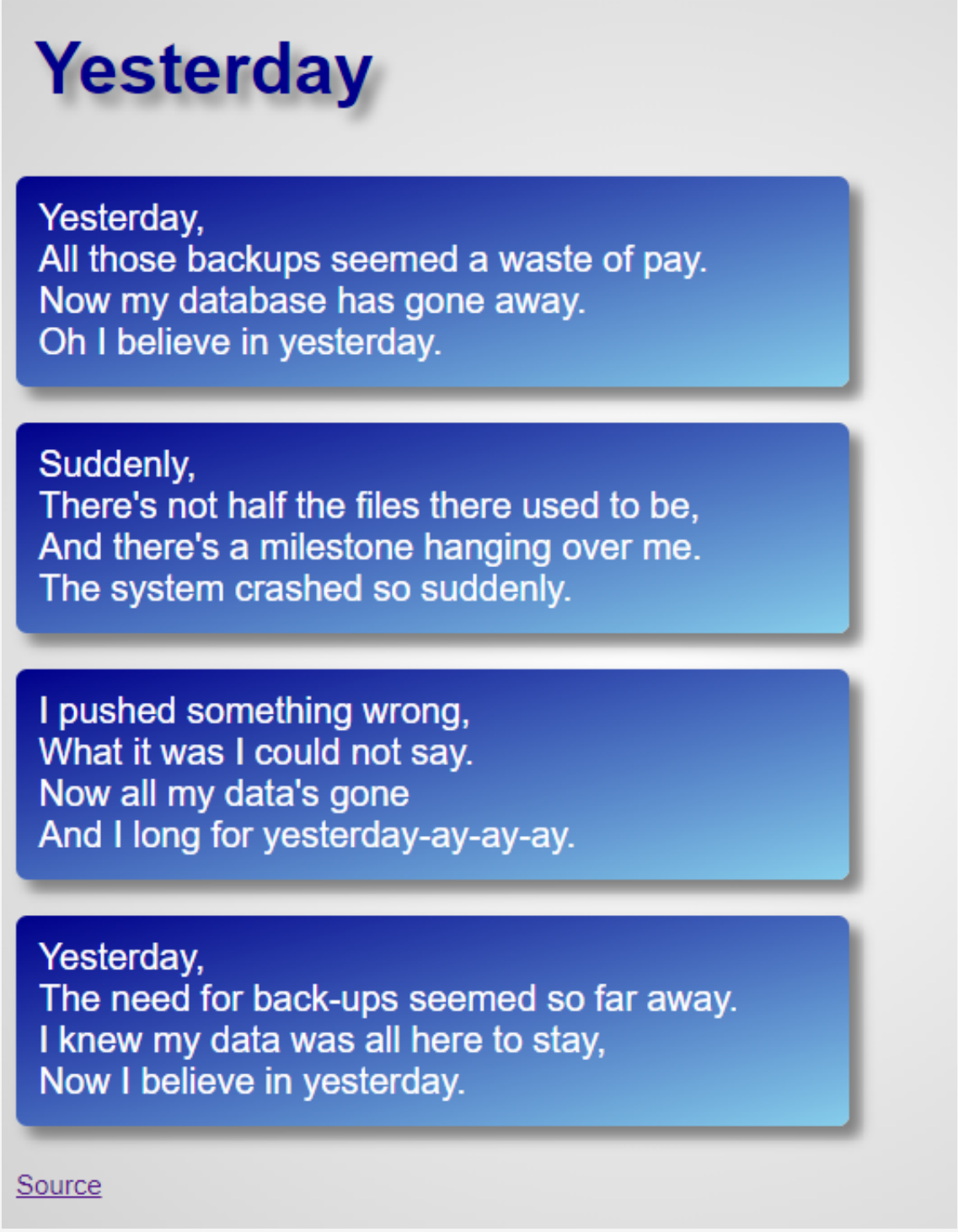
8:QUnit test 1 / 1

Testing flexbox styles of related-content class

- ✓ flex-grow of content class
- ✓ flex-basis of content class

5.14 LAB: Style the song

Download the ZIP file below containing an HTML and CSS file. The HTML file contains a song based on the Beatles' *Yesterday*. Modify the CSS file to produce the following webpage:



Requirements

1. The body element should have:
    - Arial font-family
    - A radial-gradient background that goes from white to light gray ("lightgray")

2. The h1 element should have:
    - Dark blue ("darkblue") font color
    - 8 pixel padding
    - Gray text-shadow with 5 pixel x and y offsets and a 6 pixel blur radius.

3. The p elements with lyrics should have:
    - White font color
    - 350 pixel width
    - 16 pixel font-size
    - 10 pixel padding
    - A linear-gradient background that goes from dark blue ("darkblue") to sky blue ("skyblue") and runs from the top-left corner to the bottom-right corner
    - Gray box-shadow with 5 pixel x and y offsets and a 6 pixel blur radius
    - 5 pixel border-radius

.

550544.4142762.qx3zqy7

LAB ACTIVITY | 5.14.1: LAB: Style the song | 10 / 10

Submission Instructions

Downloadable files

`song.html` and `styles.css` **Download**

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

styles.css

Drag file here
or
Choose on hard drive.

Submit for grading

Coding trail of your work [What is this?]

```
4/13 S5,10 min:4
```

Latest submission - 7:05 PM PDT on 04/13/24 — Submission passed all tests ✓ — Total score: 10 / 10

☐ Only show failing tests **Download this submission**

1:Unit test 2 / 2

<body> has proper styles

- ✓ Font family uses correct font
- ✓ radial-gradient() function is called
- ✓ First radial gradient color is white
- ✓ Second radial gradient color is light gray

2:Unit test 2 / 2

<h1> has proper styles

- ✓ Font color is dark blue
- ✓ Padding is 8px
- ✓ text-shadow property exists
- ✓ Shadow color is gray
- ✓ offset-x, offset-y, and blur-radius are specified
- ✓ offset-x is 5px
- ✓ offset-y is 5px
- ✓ blur-radius is 6px

3:Unit test 3 / 3

<p> has proper styles (part 1)

- ✓ Paragraph 1 font color is white
- ✓ Paragraph 1 width is 350px
- ✓ Paragraph 1 font size is 16px
- ✓ Paragraph 1 linear-gradient() function is called
- ✓ Paragraph 1 linear-gradient specifies three values
- ✓ Paragraph 1 linear-gradient direction goes towards bottom-right corner
- ✓ Paragraph 1 linear-gradient top-left is dark blue
- ✓ Paragraph 1 linear-gradient bottom-right is sky blue
- ✓ Paragraph 2 font color is white
- ✓ Paragraph 2 width is 350px
- ✓ Paragraph 2 font size is 16px
- ✓ Paragraph 2 linear-gradient() function is called
- ✓ Paragraph 2 linear-gradient specifies three values
- ✓ Paragraph 2 linear-gradient direction goes towards bottom-right corner
- ✓ Paragraph 2 linear-gradient top-left is dark blue
- ✓ Paragraph 2 linear-gradient bottom-right is sky blue
- ✓ Paragraph 3 font color is white
- ✓ Paragraph 3 width is 350px
- ✓ Paragraph 3 font size is 16px

- ✓ Paragraph 3 linear-gradient() function is called
- ✓ Paragraph 3 linear-gradient specifies three values
- ✓ Paragraph 3 linear-gradient direction goes towards bottom-right corner
- ✓ Paragraph 3 linear-gradient top-left is dark blue
- ✓ Paragraph 3 linear-gradient bottom-right is sky blue
- ✓ Paragraph 4 font color is white
- ✓ Paragraph 4 width is 350px
- ✓ Paragraph 4 font size is 16px
- ✓ Paragraph 4 linear-gradient() function is called
- ✓ Paragraph 4 linear-gradient specifies three values
- ✓ Paragraph 4 linear-gradient direction goes towards bottom-right corner
- ✓ Paragraph 4 linear-gradient top-left is dark blue
- ✓ Paragraph 4 linear-gradient bottom-right is sky blue

4:Unit test 3 / 3

<p> has proper styles (part 2)

- ✓ Paragraph 1 padding is 10px
- ✓ Paragraph 1 box-shadow property exists
- ✓ Paragraph 1 box-shadow color is gray
- ✓ Paragraph 1 box-shadow specifies offset-x, offset-y, and spread-radius
- ✓ Paragraph 1 box-shadow offset-x is 5px
- ✓ Paragraph 1 box-shadow offset-y is 5px

- ✓ Paragraph 1 box-shadow spread-radius is 6px
- ✓ Paragraph 1 border-radius is 5px
- ✓ Paragraph 2 padding is 10px
- ✓ Paragraph 2 box-shadow property exists
- ✓ Paragraph 2 box-shadow color is gray
- ✓ Paragraph 2 box-shadow specifies offset-x, offset-y, and spread-radius
- ✓ Paragraph 2 box-shadow offset-x is 5px
- ✓ Paragraph 2 box-shadow offset-y is 5px
- ✓ Paragraph 2 box-shadow spread-radius is 6px
- ✓ Paragraph 2 border-radius is 5px
- ✓ Paragraph 3 padding is 10px
- ✓ Paragraph 3 box-shadow property exists
- ✓ Paragraph 3 box-shadow color is gray
- ✓ Paragraph 3 box-shadow specifies offset-x, offset-y, and spread-radius
- ✓ Paragraph 3 box-shadow offset-x is 5px
- ✓ Paragraph 3 box-shadow offset-y is 5px
- ✓ Paragraph 3 box-shadow spread-radius is 6px
- ✓ Paragraph 3 border-radius is 5px
- ✓ Paragraph 4 padding is 10px
- ✓ Paragraph 4 box-shadow property exists
- ✓ Paragraph 4 box-shadow color is gray

- ✓ Paragraph 4 box-shadow specifies offset-x, offset-y, and spread-radius
- ✓ Paragraph 4 box-shadow offset-x is 5px
- ✓ Paragraph 4 box-shadow offset-y is 5px
- ✓ Paragraph 4 box-shadow spread-radius is 6px
- ✓ Paragraph 4 border-radius is 5px

Previous submissions

7:04 PM on 4/13/24	5 / 10	View