

## 6.1 Positioning elements

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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

### PARTICIPATION ACTIVITY

#### 6.1.1: Relative positioning.



#### Animation captions:

1. The "Content" <div> displays in the default location.
2. Adding relative positioning to #content does not change the "Content" <div> 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

#### 6.1.2: Relative and static positioning.



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



```

```

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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

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



```

```

- ☐ 30 pixels higher
- ☐ 30 pixels lower
- ☐ No change

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

```

```

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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019



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 web page. 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

#### 6.1.3: Fixed positioning.



#### Animation captions:

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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019



#### PARTICIPATION ACTIVITY

#### 6.1.4: Fixed and absolute 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

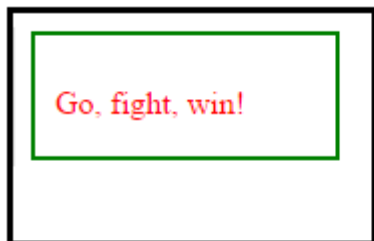
Absolute positioning is similar to fixed positioning except the position is based on the nearest positioned ancestor element that uses fixed, absolute, or relative positioning. If no positioned ancestor element exists, the element is positioned relative to the document body. An absolute positioned element scrolls with the document.

Figure 6.1.1: Cheer is absolute positioned inside a positioned ancestor (left) and relative to the document body (right).

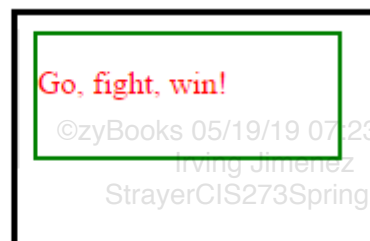
```
#container {  
  border: solid 2px green;  
  position: relative;  
  height: 60px;  
  width: 150px;  
}  
#cheer {  
  color: red;  
  position: absolute;  
  left: 10px;  
  top: 25px;  
}
```

```
#container {  
  border: solid 2px green;  
  /* No positioning */  
  height: 60px;  
  width: 150px;  
}  
#cheer {  
  color: red;  
  position: absolute;  
  left: 10px;  
  top: 25px;  
}
```

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



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



©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

#### PARTICIPATION ACTIVITY

#### 6.1.5: Absolute positioning.

Refer to the CSS below.

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

- 1) The `<span>` 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 `<span>` will not scroll with the page contents.

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019



```
<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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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 6.1.2: 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;
}
```

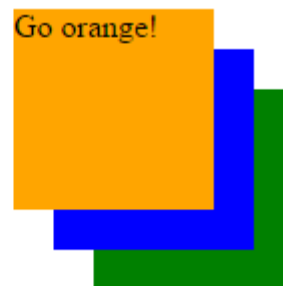
©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

```
,
#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

#### 6.1.6: 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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

**PARTICIPATION  
ACTIVITY**

## 6.1.7: Positioning practice.

The web page below displays the iconic "I ♥ NY" logo. Use the **position** and **z-index** properties to make the web page render like the expected web page.

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

HTML

CSS

```
1 <span class="first words">I</span> <span class="heart">&hearts;</span> <span cl
2 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>
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Example 1

Example 2

Example 3

Example 4

**Example 5****PARTICIPATION  
ACTIVITY**

## 6.2.1: 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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019



- 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:

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

- `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)

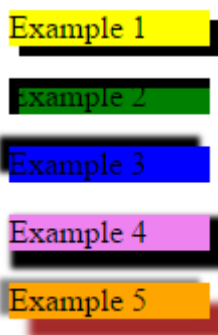
- **color** - Optional shadow color (default is usually the current CSS color)

Figure 6.2.2: 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;  
}
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

```
<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

#### 6.2.2: 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.  
☐

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

True

☐ False

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

☐ True

☐ False

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

**PARTICIPATION  
ACTIVITY**

6.2.3: Shadow practice.

The web page 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

```
1 <div class="card">
2   <p class="question">Q: Who invented the WWW?</p>
3   <p class="answer">A: Tim Berners-Lee</p>
4 </div>
5
6 <div class="card">
7   <p class="question">Q: When was the first website published?</p>
8   <p class="answer">A: 1991</p>
9 </div>
10
11 <div class="card">
12   <p class="question">Q: What web browser did most people use in the early 2000s?</p>
13   <p class="answer">A: Internet Explorer</p>
14 </div>
15
```

Render web page

Reset code

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019



## Your web page

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## 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, forth 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***, and ***border-bottom-right-radius***.

### PARTICIPATION ACTIVITY

#### 6.2.4: Rounded corners.



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



A



B



C



D

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

D

A

C

B

```
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-height** - Image section height
- **border-image-width** - Image section width
- **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

### PARTICIPATION ACTIVITY

6.2.5: Try different border-image values.



The borderv1.png image is used to display a border image around the `<div>` in the web page below. The blue circles and green diamonds in the image are about 30 x 30 pixels.



©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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

1. Change the image width/height from 31 to 15 and render the page. Observe how roughly half the circle and half the diamond is used to render the border.

2. Change the 15 to 60 and render the page. Observe how a 60 x 60 pixel section (2/3 of the image) is used to render the border corners. Since `border1.png` is only 90 x 90 pixels, an unused 60 x 60 pixel section does not exist, so the border sides are empty.
3. Change the `border` size from 20px to 30px. Render the web page and observe how the border size increased.
4. Change the image width/height back to 31 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. Change "repeat" to "stretch". Render the page and observe how the green diamonds stretch to fill the border.
6. Finally, add the number 15 after 31. The 31 is now the height only, and 15 is the width. Render the page and observe how only half the green diamond is used to form the left and right borders.

HTML CSS

```
1 <div id="example">Example using a border image.</div>
2
```

Render web page

Reset code

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## Your web page

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

### PARTICIPATION ACTIVITY

#### 6.2.6: Border images.



Refer to the CSS below.

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

- 1) 50 x 50 pixel sections of some-border.png are used to create the border image.

☐ True

☐ False
- 2) If some-border.png is 50 x 50 pixels, then the border will have empty sides.

☐ True

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

☐ True

☐ False
- 4) The width and height of a border image are specified by **border-**

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

**image-width** and **border-image-height**.

- ☐ True
- ☐ False

## CSS3 browser support

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

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.

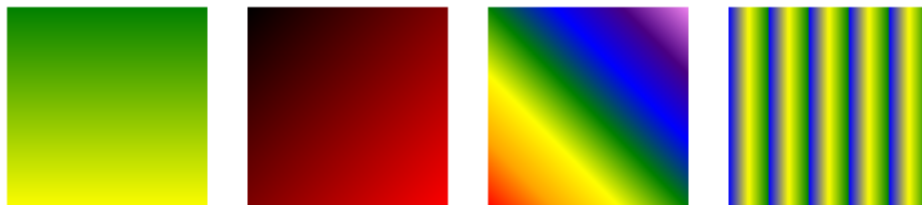
The **repeating-linear-gradient()** function repeats a linear gradient where the color values are supplied an optional percent. The percentage value after the last color is the percent of the gradient's total length the repeating gradient should occupy. Ex:

**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 6.2.3: 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

#### 6.2.7: Linear gradients.

- 1) What direction creates the gradient below?



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

**Check**[Show answer](#)

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

- 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](#)

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## 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 6.2.4: 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%);
}
```



©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

#### PARTICIPATION ACTIVITY

#### 6.2.8: 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?



©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019





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

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.

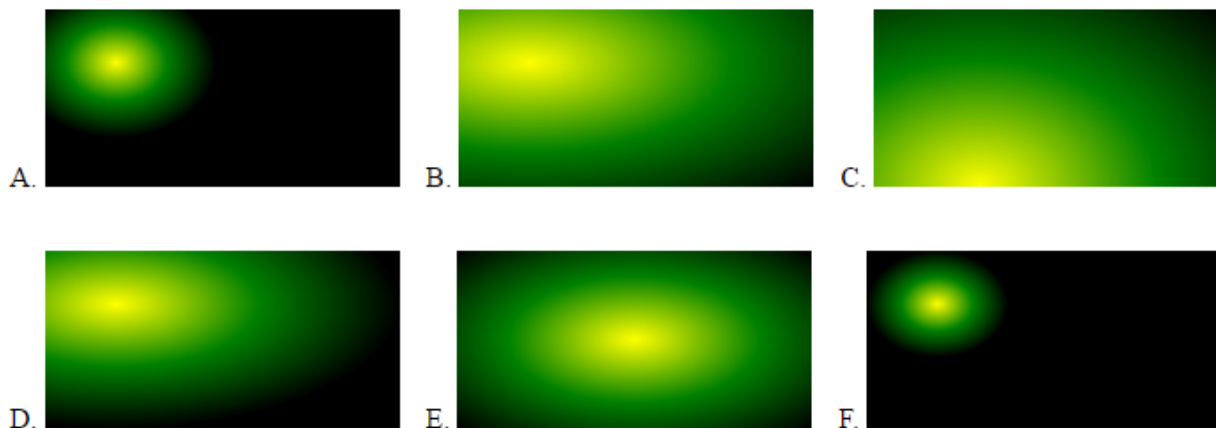
By default, a radial gradient's shape reaches to the farthest corner of the containing rectangle. An extent keyword describes the size of the radial gradient's shape:

- **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

#### 6.2.9: Positioned radial gradients.

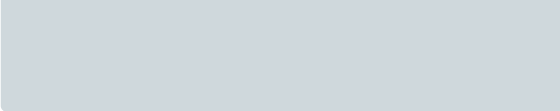
Match the background with the radial gradient CSS that produced the background.



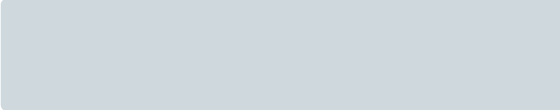
©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

`radial-gradient(at 60px 100px,  
yellow, green, black)`


`radial-gradient(yellow, green,  
black)`



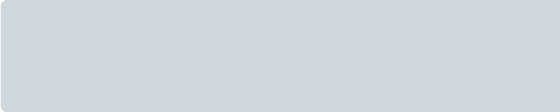
radial-gradient(closest-corner at 20% 30%, yellow, green, black)



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**

## 6.2.10: Gradient practice.



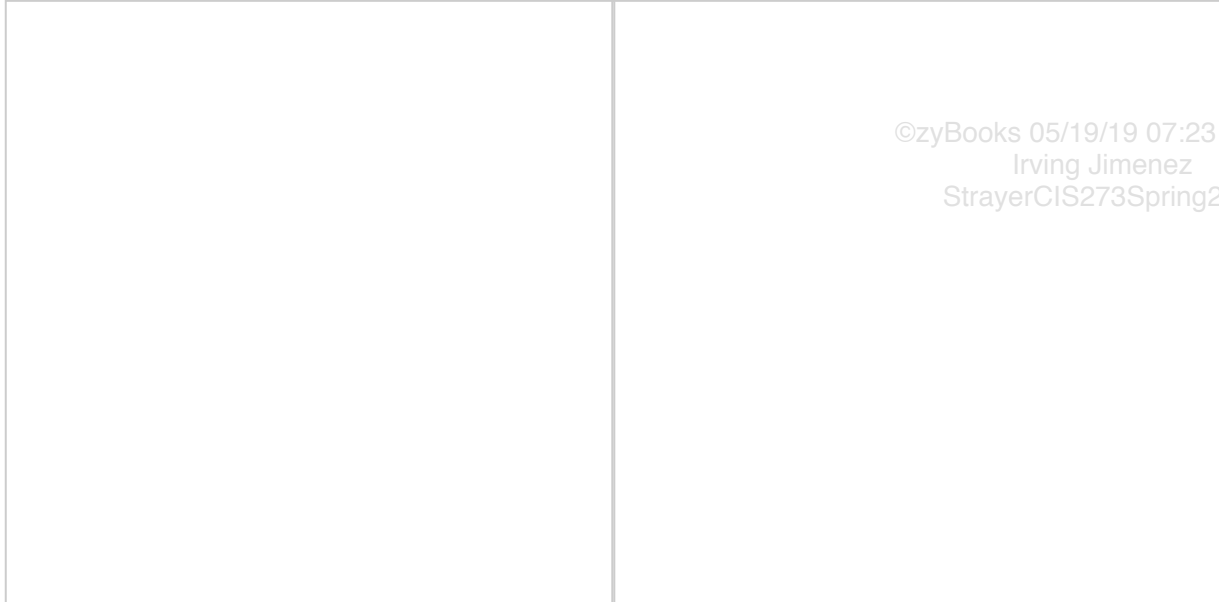
The web page below displays an advertisement with a background produced by the CSS function `repeating-radial-gradient()`. Make the following modifications to the HTML and CSS so the rendered web page resembles the expected web page:

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 web page. 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="advertisement1">Vote this Tuesday!</div>
2
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

[Render web page](#)[Reset code](#)**Your web page****Expected web page****CHALLENGE  
ACTIVITY****6.2.1: Special effects.**[Start](#)

For the `<p>` tag, set the text-shadow to be green with an offset-x of 3px and offset-y of 12px. [SHOW EXPECTED](#)

CSS

HTML

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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

1

2

3

4

Check

Next

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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## 6.3 Styling forms

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 6.3.1: HTML form without CSS styling and an improved form with styling.

Name   
Email   
Service

Name   
Email   
Service

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

### PARTICIPATION ACTIVITY

6.3.1: Create a styled form.



The web page 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;  
}
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Render the web page and observe the labels are equal length and right aligned.

2. Add a selector that gives the text inputs and drop-down menu the same consistent width with some padding to increase the size of the inputs and with a margin to leave some space between the inputs. Also, change the border color and radius to make the inputs look nicer. Finally, use `box-sizing: border-box` to make the browser include the content, padding, and border in the width and height to make the text inputs the same size as the drop-down menu.

```
input[type=text], select {  
  width: 350px;  
  padding: 10px;  
  margin: 6px 0;  
  border: 1px solid #aaa;  
  border-radius: 4px;  
  box-sizing: border-box;  
}
```

Render the web page and observe the inputs are equal size and are spaced out.

3. Add styling to the submit button changing the color to blue and changing the appearance to look 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 web page 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;  
}
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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

HTML

CSS

```
1 <form>
2   <div>
3     <label for="name">Name</label>
4     <input type="text" id="name" name="fullname">
5   </div>
6   <div>
7     <label for="email">Email</label>
8     <input type="text" id="email" name="email">
9   </div>
10  <div>
11    <label for="service">Service</label>
12    <select id="service" name="service">
13      <option>Basic</option>
14      <option>Prime</option>
15      <option>Deluxe</option>
16    </select>
17  </div>
18
19  <input type="submit" value="Register">
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Render web page

Reset code

### Your web page

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

#### PARTICIPATION ACTIVITY

#### 6.3.2: 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`

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

3) In the example above, what happens if `box-sizing: border-box` is not applied to the text inputs and drop-down menu?



- ☐ The text inputs will not be visible.
- ☐ The drop-down menu will not be as tall as the text inputs.
- ☐ The text inputs will be wider than the drop-down menu.

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.

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## Proper use of field labels

*Form field labels should be placed uniformly in the same location on a website's*

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.



An input field can be further improved. Changing an input's border color or background color diverts 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

#### 6.3.3: Augmenting an input.



The web page 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 HTML and CSS:

1. Create a CSS class that sets a text input's border to red to indicate an error:

```
input.error {
  border: 2px solid red;
}
```

Add the class to the email input field in the HTML and render the web page. The email input field 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 a light blue:

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

Render the web page and observe that the name and email inputs become light blue when the inputs have the focus. You may also notice that your 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: 8px 8px;  
  background-repeat: no-repeat;  
  padding-left: 40px;  
}
```

Render the web page and observe the search icon in the search input.

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

HTML

CSS

```
1 <input type="search" placeholder="Search">  
2  
3 <form>  
4   <div>  
5     <label for="name">Name</label>  
6     <input type="text" id="name" name="fullname">  
7   </div>  
8   <div>  
9     <label for="email">Email</label>  
10    <input type="text" id="email" name="email" value="joe@gmail">  
11  </div>  
12  <div>  
13    <label for="service">Service</label>  
14    <select id="service" name="service">  
15      <option>Basic</option>  
16      <option>Prime</option>  
17      <option>Deluxe</option>  
18    </select>  
19  </div>
```

Render web page

Reset code

Your web page

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

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) In the example above, removing the **background-repeat** property causes the background image to display repeatedly on the search input.

- ☐ True  
☐ False

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Radio buttons and checkboxes use the styling properties supplied by the browser and may differ between browsers. Styling radio buttons and checkboxes requires:

1. Making the radio button or checkbox's label clickable
2. Hiding the default radio button or checkbox
3. Displaying a custom radio button or checkbox before each label that changes appearance when checked or focused

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

#### 6.3.5: Styling radio buttons.

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

#### Animation content:

undefined

#### Animation captions:

1. 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 all radio buttons to have absolute positioning with 1x1 size and clipping the 1x1 area makes the radio buttons invisible, 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 is displayed around the content before the label.

#### PARTICIPATION ACTIVITY

#### 6.3.6: Style the checkboxes.



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

1. Modify the CSS that selects the radio button's label and adds a pointer cursor to also select the checkboxes' labels and add a pointer cursor to the checkboxes' labels:

```
input[type=radio] + label, input[type=checkbox] + label {
    cursor: pointer;
    margin-right: 20px;
    font-size: 16pt;
}
```

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

2. Modify the CSS that selects and hides the radio buttons to also select and hide the checkboxes:

```
input[type=radio], input[type=checkbox] {
    position: absolute;
    height: 1px;
    width: 1px;
    clip: rect(0 0 1 1);
}
```

Render the web page and verify the default checkboxes are no longer visible.

3. Modify the CSS that adds the radio button before the radio button labels to add the same radio button in front of the checkbox labels:

```
input[type=radio] + label::before, input[type=checkbox] + label::before {
  content: "\00a0"; /* Non-breaking space */
  etc...
}
```

Render the web page and verify the checkboxes are all circular like the radio buttons.

4. Modify the CSS so only the radio buttons appear as circles, but the checkboxes do not:

```
input[type=radio] + label::before {
  border-radius: 10px;
}

input[type=radio] + label::before, input[type=checkbox] + label::before {
  content: "\00a0";
  /* No border-radius property */
  etc...
}
```

Render the web page and verify the checkboxes are now square.

5. Add CSS 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 web page and verify that clicking on a checkbox displays a checkmark in the box.

6. Add CSS to display a gray border around the checkbox that has the focus:

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

Render the web page and verify that the last checkbox clicked on has the gray border around the checkbox.

HTML CSS

```
1 <p>Size:</p>
2 <div>
3   <input id="small" type="radio" name="size" value="small">
4   <label for="small">Small</label>
5 </div>
6 <div>
7   <input id="medium" type="radio" name="size" value="medium">
8   <label for="medium">Medium</label>
9 </div>
10 <div>
11   <input id="large" type="radio" name="size" value="large">
12   <label for="large">Large</label>
13 </div>
14
15 <p>Flavors:</p>
16 <div>
17   <input id="raspberry" type="checkbox" name="flavor" value="chocolate">
18   <label for="raspberry">Raspberry</label>
19 </div>
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Render web page

Reset code

### Your web page

#### PARTICIPATION ACTIVITY

#### 6.3.7: Styling radio buttons and checkboxes.



©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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>
```

- ☐ Before<span>Test</span>
- ☐ <span>BeforeTest</span>
- ☐ <span>Test</span>Before

#### CHALLENGE ACTIVITY

#### 6.3.1: Styling forms.

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

Start

For the `<label>` tag, set display to inline-block, use a width of 40px, align the text to the left, and add a margin on the right of 4px. **SHOW EXPECTED**

CSS

HTML

```
1 label {  
2  
3 /* Your solution goes here */  
4  
5 }  
6 form {  
7   font: 10pt Arial;  
8   background-color: #eee;  
9   padding: 10px;  
10 }
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

1

2

3

4

Check

Next

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

## 6.4 Sass

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

### 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.

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

**PARTICIPATION  
ACTIVITY**

## 6.4.1: Compiling SCSS into 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**

## 6.4.2: 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 web page's background color to blue.

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

- ☐ True  
☐ False

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019





- 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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## 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 6.4.1: Selector nesting.

// SCSS	/* Resulting CSS */
<pre>.notes {   font-size: smaller;    strong {     color: green;   } }</pre>	<pre>.notes {   font-size: smaller; }  .notes strong {   color: green; }</pre>

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

Figure 6.4.2: Referencing the parent with **&**.

// SCSS	/* Resulting CSS */
<pre>a {   text-decoration: none;   &amp;:hover {     color: blue;   } }</pre>	<pre>a {   text-decoration: none; }  a:hover {   color: blue; }</pre>

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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 6.4.3: Property nesting.

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

/* Resulting CSS */
p {
  font-family: Arial;
  font-size: 12pt;
  font-weight: bold;
}
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

**PARTICIPATION  
ACTIVITY**

6.4.3: 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;

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

```

    }
    ○ section article:hover
    {
        color: blue;
    }

```

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

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

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

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

©zyBooks 05/19/19 07:23 473675  
 Irving Jimenez  
 StrayerCIS273Spring2019

## 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)`

©zyBooks 05/19/19 07:23 473675  
 Irving Jimenez  
 StrayerCIS273Spring2019

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`.

## ACTIVITY

## 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.

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

PARTICIPATION  
ACTIVITY

## 6.4.5: 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](#)

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## Functions

SassScript includes a large number of utility functions.

Table 6.4.1: Some SassScript functions

Table 6.4.1. Some SassScript functions.

Function	Description	Example
<code>lighten(color, amount)</code>	Returns a <b>color</b> lightened by an <b>amount</b> between 0% and 100%	<pre>/* Returns #d00 */ \$color: lighten(#a00, 10%);</pre>
<code>invert(color)</code>	Returns the inverse (negative) of a <b>color</b>	<pre>/* Returns #5ff */ \$color: invert(#a00);</pre>
<code>to-upper-case(string)</code>	Returns <b>string</b> using all uppercase characters.	<pre>/* Returns "BEHOLD!" */ \$message: to- upper- case("Behold!");</pre>
<code>round(number)</code>	Returns a <b>number</b> rounded to the nearest whole number	<pre>/* Returns 21px */ \$width: round(20.5px);</pre>
<code>random(limit)</code>	Returns a random integer between 1 and <b>limit</b> (inclusive)	<pre>/* Returns a number between 1 and 5 */ \$width: random(5) * 20px;</pre>

PARTICIPATION  
ACTIVITY

## 6.4.6: 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

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

## 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

6.4.7: Including mixins.

### 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

6.4.8: 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;  
}
```

©zyBooks 05/19/19 07:23 473675  
Irving Jimenez  
StrayerCIS273Spring2019

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

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

©zyBooks 05/19/19 07:23 473675

Irving Jimenez

StrayerCIS273Spring2019

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

```
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 r
}
```

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