* CSS variables
  + Can be used to change many CSS style properties at once by changing only one value
  + Create a variable by writing -- before name, then colon, and then value, such as:
    - --penguin-skin: gray;
  + Assign a variable to a CCS property
    - background: var(--penguin-skin);
  + Attach fall back value to the variable (which would be used if the variable doesn't work):
    - background: var(--penguin-skin, black);
  + Variables can be changed when the condition of a media query is met (by changing the width of the browser window)
* :root element can be thought of as a container for your entire HTML document
  + If variables are defined in :root, they will be available throughout the whole web page
  + Variable defined in :root can be overwritten by setting the variable again in another element
* text-align
  + Possible values: left, center, right, justify
* width
  + Relative length units (such as em), absolute length units (such as px), or as a percentage of its containing parent element
* height
  + Relative length units (such as em), absolute length units (such as px), or as a percentage of its containing parent element
* strong tag - to bold things
  + Applies CSS property: font-weight: bold;
* u tag to underline
  + Applies CSS property: text-decoration: underline;
* em tag used to emphasize text
  + Applies CSS property: font-style: italic;
* s tag to strickthrough text
  + Applies CSS property: text-decoration: line-through;
* <hr> tag to add line - self-closing tag
* background-color: rgba(45, 45, 45, 0.1);
* box-shadowproperty applies one or more shadows to an element.
  + Takes values for offset-x(how far to push the shadow horizontally from the element), offset-y(how far to push the shadow vertically from the element), blur-radius, spread-radiusand a color value, in that order. The blur-radiusand spread-radiusvalues are optional
  + Example: box-shadow: 0 10px 20px rgba(0,0,0,0.19), 0 6px 6px rgba(0,0,0,0.23);
* opacity
  + 1 - opaque; 0 - transparent
* text-transform
  + lowercase - "transform me"
  + uppercase - "TRANSFORM ME"
  + capitalize - "Transform Me"
  + initial - Use the default value
  + inherit - Use the text-transform value from the parent element
  + none - Default: Use the original text
* font-size - can be used to set size in px
* line-height - change vertical space of each line
* pseudo-class is a keyword that can be added to selectors, in order to select a specific state of the element
  + a:hover {

   color: red;

}

* CSS box model
  + Box-level items - own line
  + Inline items
* Position: relative - allows you to specify how CSS should move it relative to its current position in the normal flow of the page
  + Used with:
    - Offset direction: left, right, top or bottom
      * Moves the element away from the referenced side (effectively, the opposite direction).
    - Pixels, percentages, or ems
  + Changing the relative position of the element does not change the normal flow - other elements still appear as it was in it's default position
  + Example: moves the p elements 10 px from the bottom
    - p {

  position: relative;

  bottom: 10px;

}

* Position: absolute
  + Locks element relative to the parent positioned container (which is done through position: relative)
    - Locks to closest relative
  + Removes element from normal flow - other elements ignore it
* position: fixed
  + used with the CSS offset properties and also removes the element from the normal flow of the document
* float: left, right
  + Floating element removed from normal flow; pushed to left or right of parent container
* z-index
  + Can be used to change whether the element appears behind or on top of other elements
  + Must be an integer
  + Higher values of z-index move the element on top of other elements with lower values
* margin: auto
  + Using margin to center horizontally on the page
  + Can be applied to div and also to img (if we also change the display to block)
* Colors
  + Complementary colors - opposite each other on the color wheel
    - red (#FF0000) and cyan (#00FFFF)
    - green (#00FF00) and magenta (#FF00FF)
    - blue (#0000FF) and yellow (#FFFF00)
* RGB additive color model
  + Primacy colors: Red (R), Green (G), Blue (B)
  + Secondary colors - created by mixing primary colors
    - Cyan - G and B (00FFFF)
    - Magenta - R and B
    - Yellow - R and G
  + Tertiary colors
    - orange = #FF7D00
* Other colors
  + Raspberry FF007D
  + Teal 09A7A1
* hsl() - alternative way of picking color by stating it's properties
  + Hue - color - between 0 and 360
  + Saturation - amount of gray in a color
    - In percentage - 100% being fully saturated
  + Lightness - amount of white or black in a color
    - 0% black
    - 100% white
    - 50% normal color
  + Examples with fully saturated, normal color:
    - red: hsl(0, 100%, 50%)
    - yellow: hsl(60, 100%, 50%)
    - green: hsl(120, 100%, 50%)
    - cyan: hsl(180, 100%, 50%)
    - blue: hsl(240, 100%, 50%)
    - magenta: hsl(300, 100%, 50%)
    - Can be used to adjust the tone of a color
* linear-gradient() used to create a gradient
  + background: linear-gradient(gradient\_direction, color 1, color 2, color 3, ...); [have to specify at least two colors, have option to add more colors]
  + gradient-direction specified in deg (90deg specifies a vertical gradient; 45deg is angled as a backslash)
  + Example: background: linear-gradient(90deg, red, yellow, rgb(204, 204, 255));
* repeating-linear-gradient() - repeats the gradient
  + Example:

background: repeating-linear-gradient(

90deg,

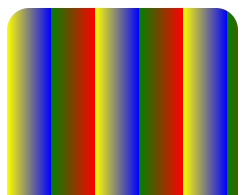
yellow 0px,

blue 40px,

green 40px,

red 80px

);



* + Example 2:
  + background: repeating-linear-gradient(

45deg,

yellow 0px,

yellow 40px,

black 40px,

black 80px

);



* background property supports url() function - can link to an image or gradient
  + Example:

background: url("<https://i.imgur.com/MJAkxbh.png>");

* border-radius: 50% can be used to create circles (if the width and height of the object are the same size)
* left: xx and right: xx can be set to percentages too (not just pixels)
  + Example: left: 65%;
* transform property with scale() function can be used to change the size of an element

p {

  transform:scale(2);

}

* Other functions of the transform property: move, rotate, skew
* Transform property can be used with pseudo-classes, such as :hover to add interactivity
  + Example: to scale the paragraph elements to 2.1 times their original size when a user hovers over them:

p:hover {

              transform: scale(2.1);

          }