|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Colors** |  |

|  |
| --- |
| **Your Tasks** |
| * Indicate the three ways colors can be described in CSS * Distinguish between foreground and background * Interpret colors expressed in RGB * Interpret colors expressed in hexadecimal * Have Ms. Pluska check off the above tasks * Apply the *z-index* property * Apply the *display: inline-block* property * Apply the *float* property * Receive credit for the group portion of this lab |

* **Indicate the three ways colors can be described in CSS**

CSS supports a wide variety of colors. These include named colors, like blue, black, and LimeGreen, along with colors described by a numeric value. Using a numeric system allows us to take advantage of the whole spectrum of colors that browsers support. In this lesson, we’re going to explore all the color options CSS offers.

Colors in CSS can be described in three different ways:

* Named colors — English words that describe colors, also called keyword colors
* RGB — numeric values that describe a mix of red, green, and blue
* HSL — numeric values that describe a mix of hue, saturation, and lightness
* **Distinguish between foreground and background**

Before discussing the specifics of color, it’s important to make two distinctions about color. Color can affect the following design aspects:

1. The foreground color
2. The background color

*Foreground* color is the color that an element appears in. For example, when a heading is styled to appear green, the *foreground* color of the heading has been styled.

Conversely, when a heading is styled so that its *background* appears yellow, the *background* color of the heading has been styled

In CSS, these two design aspects can be styled with the following two properties:

1. color - this property styles an element’s foreground color.
2. background-color - this property styles an element’s background color.

In the example above, the text of the heading will appear in red, and the background of the heading will appear blue.

|  |  |
| --- | --- |
|  | |
| **Index.html** | **Styles.css** |
| <div class="one">BOX 1</div> | .one {  color: blue;  background-color: yellow;  height:2em;  } |

|  |  |
| --- | --- |
| Write css rules to style the boxes as shown | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">One</div>  <div class = "two">Two</div> |  |

* **Interpret colors expressed in RGB**

There is another syntax for representing RGB values that uses decimal numbers. It looks as follows,

|  |
| --- |
| h1 {  color: rgb(23, 45, 23);  } |

Here, each of the three values represents a color component, and each can have a decimal number value from 0 to 255. The first number represents the amount of red, the second is green, and the third is blue.

Recall, the number of places required to represent a given number in binary can be determined as follows

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary – base 2** | | | | | | | | | |
|  | 27 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 |
| Max value | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| Places | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

Based on the table above, the base 10 numbers can be represented as follows in binary

|  |  |
| --- | --- |
| **Base 10** | **binary** |
| 128 | 10000000 |
| 64 | 1000000 |
| 32 | 100000 |
| 16 | 10000 |
| 8 | 1000 |
| 4 | 100 |
| 2 | 10 |
| 1 | 1 |

|  |
| --- |
| How many bits (or places) are required to represent the number 255 in binary?  How many bytes are required (1 byte = 8 bits)?  How many bits are required to represent an RGB color?  How many bytes are required to represent an RGB color? |
|  |

|  |  |
| --- | --- |
| **Color** | **RGB** |
| DarkSeaGreen  Sienna  SaddleBrown3  Brown  Black  White  Aqua | 143,188,143  160, 81, 45  139, 69, 19  150, 75, 0  000, 000, 000  255, 255, 255  000, 255, 255 |

|  |  |
| --- | --- |
| Write css rules to style the boxes as shown | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">The background is sienna and the foreground is Aqua</div>  <div class = "two">The background is darkseagreen and the foreground is white</div> |  |

* **Interpret colors expressed in hexadecimal**

Another syntax that we can use to specify colors is called hexadecimal. Colors specified using this system are called hex colors. A hex color begins with a hash character (#) which is followed by three or six characters. The characters represent values for red, blue and green.

|  |  |
| --- | --- |
| **Color** | **Hexadecimal value** |
| DarkSeaGreen  Sienna  SaddleBrown3  Brown  Black  White  Aqua | #8FBC8F  #A0522D  #8B4513  #A52A2A  #000000 or #000  #FFFFFF or #FFF  #00FFFF or #0FF |

In the table above, you may notice that there are both letters and numbers in the values. This is because the hexadecimal number system has 16 digits (0-15) instead of 10 (0-9) like you are used to. To represent 10-15, we use A-F. This is illustrated below,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| F | E | D | C | B | A | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

The number of places required to represent a given number in hexadecimal can be determined as follows

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 163 | 162 | 161 | 160 |
| Max value | 4096 | 256 | 16 | 1 |
| Places | 4 | 3 | 2 | 1 |

Based on the table above, the base 10 numbers can be represented as follows in hexadecimal

|  |  |
| --- | --- |
| **Base 10** | **hexadecimal** |
| 4096 | 1000 |
| 256 | 100 |
| 16 | 10 |
| 1 | 1 |

|  |  |
| --- | --- |
| Convert the following hexadecimal numbers to decimal | |
| **Hexadecimal** | **Decimal** |
| A1 |  |
| B2 |  |
| FF |  |
| 1A |  |

|  |
| --- |
| How many places are required to represent the number 255 in hexadecimal? |
|  |

The following example illustrates how to convert a decimal number into hexadecimal,

number = 3741

|  |  |  |  |
| --- | --- | --- | --- |
| **Base divisor** | **Number divided** | **Remainder** | **Hexadecimal value** |
| 16 | 3741 ÷ 16 = 233 | 13 | D |
| 16 | 233 ÷ 16 = 14 | 9 | 9 |
| 16 | 14 ÷ 16 = 0 | 14 | E |

Now list hexadecimal remainders from top to bottom: E9D

|  |  |
| --- | --- |
| Convert the following decimal numbers to hexadecimal. To do this following these steps: | |
| **Decimal** | **Hexadecimal** |
| 255 |  |
| 64 |  |
| 32 |  |
| 128 |  |

The RGB values associated with a color expressed in hexadecimal can be interpreted as follows,

|  |  |  |
| --- | --- | --- |
| **Color** | **Hexadecimal** | **Interpretation** |
| Sienna | #A0522D | A0 = Red  52 = Green  2D = Blue |

Notice that the last two digits of the hexadecimal number represent Red, the middle two represent Green, and the last two represent Blue. Also, notice that because the maximum value for either R, G, B cannot exceed a byte of memory. Put another way,

|  |  |  |
| --- | --- | --- |
| Max R, G, B values in decimal, binary, and hexadecimal | | |
| **Decimal** | **Binary** | **Hexadecimal** |
| 255 | 11111111 | FF |

Now let’s return to our hexadecimal colors from before,

|  |  |
| --- | --- |
| **Color** | **Hexadecimal value** |
| DarkSeaGreen  Sienna  SaddleBrown3  Brown  Black  White  Aqua | #8FBC8F  #A0522D  #8B4513  #A52A2A  #000000 or #000  #FFFFFF or #FFF  #00FFFF or #0FF |

Notice that Black, White, and Aqua are all represented with both three characters and six characters. This can be done with hex colors whose number pairs are the same characters. In the example above, Aqua can be represented as #0FF because both of the first two characters are 0 and the second and third pairs of characters are both Fs. Keep in mind that all three character hex colors can be represented with six characters (by repeating each character twice).

You can include hex colors in your css rules just as you would include named colors:

|  |
| --- |
| background-color: #9932cc; |

|  |  |
| --- | --- |
| Write css rules to style the boxes as shown | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">The background is sienna and the foreground is Aqua</div>  <div class = "two">The background is darkseagreen and the foreground is white</div> |  |

* **Have Ms. Pluska check off the above tasks**



Before you continue have Ms. Pluska check off the above tasks

Do not continue until you have Ms. Pluska’s (or her designated TA’s) signature \_\_\_\_\_\_\_\_\_\_\_\_

In the example below, we set the *.box-top* position to relative and the *z-index* to 2. We changed position to relative, because the *z-index* property does not work on static elements. The *z-index* of 2 moves the *.box-top* element forward, because it is greater than the .box-bottom *z-index*, 1.

|  |  |
| --- | --- |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = “box-top”></div>  <div class = “box-bottom”></div> | .box-top {  background-color: Aquamarine;  position: relative;  z-index: 2;  height:100px;  width:100px;  }  .box-bottom {  background-color: DeepSkyBlue;  position: absolute;  top: 20px;  left: 50px;  z-index: 1;  height:100px;  width:100px;  } |

|  |  |
| --- | --- |
| Write css rules to style the boxes as shown | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">One</div>  <div class = "two">Two</div>  <div class = "three">Three</div> |  |

* **Apply the *display: inline-block* property**

The default display for html elements is vertically. The display: inline-block property forces elements to appear next to each other. Consider the following example,

|  |  |
| --- | --- |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">One</div>  <div class = "two">Two</div>  <div class = "three">Three</div> | .one {  background-color: purple;  width:100px;  height:100px;  display: inline-block;  }  .two {  background-color: yellow;  width:100px;  height:100px;  display: inline-block;  }  .three {  background-color: gray;  width:100px;  height:100px;  display: inline-block;  } |

|  |  |
| --- | --- |
| Write a css rule to style the navigation bar shown. Use the *display: inline-block* property to align the elements horizontally | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <ul class="nav">  <li>Projects</li>  <li>Proudest Moments</li>  <li>Experience</li>  <li>Education</li>  <li>Contact</li>  </ul> |  |

* **Apply the *float* property**

So far, you’ve learned how to specify the exact position of an element using offset properties. If you’re simply interested in moving an element as far left or as far right as possible on the page, you can use the *float* property.

The *float* property can be set to one of two values:

1.  *left* - this value will move, or float, elements as far left as possible.

2. *right* - this value will move elements as far right as possible.

Floated elements must have a width specified, as in the example above. Otherwise, the element will assume the full width of its containing element, and changing the float value will not yield any visible results.

|  |  |
| --- | --- |
|  | |
| **Index.html** | **Styles.css** |
| <div class = "one">One</div>  <div class = "two">Two</div>  <div class = "three">Three</div> | .one {  background-color: purple;  width:100px;  height:100px;  float: right;  }  .two {  background-color: yellow;  width:100px;  height:100px;  float: right;  }  .three {  background-color: gray;  width:100px;  height:100px;  float: right;  } |

|  |  |
| --- | --- |
| Write a css rule to style the navigation bar shown. Use the *float* property to align the elements horizontally. Use the *list-style-type* property to remove the default bullets | |
| **Output** | |
|  | |
| **Index.html** | **Styles.css** |
| <ul class="nav">  <li>Projects</li>  <li>Proudest Moments</li>  <li>Experience</li>  <li>Education</li>  <li>Contact</li>  </ul> |  |

* **Receive Credit for the group portion of this lab**



* Indicate the names of all group members.
* Have Ms. Pluska check your Box Model tasks
* Submit your lab to the needs to be graded folder to receive credit for the group portion of this lab.
* Do not submit your lab until you have Ms. Pluska’s (or her designated TA’s) signature

\_\_\_\_\_\_\_