Styling Web Pages with CSS

# 1 Introduction

Web pages are plain text documents formatted with HTML. Different HTML elements -- or tags -- format plain text to change their appearance. For instance, heading tags format their text to be much larger than their surrounding plain text. Paragraph tags add vertical spacing. List tags enumerate or bullet the line items they surround. Table tags can format data into rows and columns. Other than basic appearance and layout, HTML does not offer much else to make a Web page more appealing or distinguished from other Web pages.

To make Web pages stand out, we must use ***CSS*** (***Cascading Style Sheets***). CSS is a computer language that can describe the appearance of an existing HTML document. CSS can refer to different parts of a document and configure various appearance attributes such as foreground and background color, font, alignment, spacing, borders, paddings, etc. This assignment will give us a chance to learn about how to use CSS to style Web pages.

Assignments in this course contain three main sections: ***Labs***, ***Tuiter***, and ***Challenge***. The ***Lab*** section in this assignment will give you an opportunity to practice the concepts described in this assignment, i.e., HTML. Once you've had a chance to practice HTML, in the ***Tuiter*** section you'll be asked to apply what you've learned to build a Website called ***Tuiter*** inspired on a popular social network. The ***Challenge*** section explores additional, more challenging requirements that are optional for undergraduates, but required for graduate students.

Create a new branch called ***a3*** and do all your work there. When done, add, commit and push the branch to GitHub. Deploy the new branch to Netlify and confirm it's available in a new URL based on the branch name.

# 2 Learning objectives

* Styling Web content with Cascading Style Sheets (CSS)
* Laying out content with Flex
* Animating content with keyframes

# 3 Labs (50pts)

This section presents several CSS exercises to practice styling HTML documents. Use the same project you worked on last assignment. After you work through the exercises you will apply the skills to create ***Tuiter*** on your own. Using ***IntelliJ***, open the project you created in the previous assignment. From within IntelliJ, use ***File***, ***Open Project***, and navigate to the project directory, (***tuiter-react-web-app***), and click ***Open*** or ***OK***. From within IntelliJ, on the ***Project*** tab, open the ***tuiter-react-web-app*** directory, and then the ***public*** directory. Do all your work under the ***public*** directory of your project.

Under the ***public/labs*** directory, create a new directory called ***a3*** and create ***index.html*** under ***public/labs/a3***. For this assignment, do all your work in ***public/labs/a3/index.html***. To make it easy on the TAs, add a link to this new ***index.html*** file in ***public/index.html***.

## 3.1 Styling a document's look and feel (10pts)

### 3.1.1 Styling elements with the ***style*** attribute (3pts)

An HTML element's ***style*** attribute can configure the look and feel of the element by changing the values of its style properties as shown below

| <**element style="property1**: **value1**; **property2**: **value2"**>  element body  </**element**> | |
| --- | --- |

The value of the style attribute is a string formatted with a language called ***CSS*** or ***Cascading Style Sheets*** and is the focus of this assignment. To practice using the ***style*** attribute, copy and paste the example below into ***public/labs/a3/index.html***

| ***/public/labs/a3/index.html*** | |
| --- | --- |
| <**h1**>Cascading Style Sheet (CSS)</**h1**>  <**h2**>Style attribute and tag</**h2**>  <**h3**>Style attribute</**h3**>  <**p style="background-color**: **blue**; **color**: **white"**>  Style attribute allows configuring look  and feel right on the element. Although it's  very convenient it is considered bad practice  and you should avoid using the style attribute  </**p**> | **Cascading Style Sheet (CSS)**  **Style attribute and tag** |

In the exercise above we styled the paragraph element with its ***style*** attribute. We changed the color of its background by setting the ***background-color*** property to ***blue*** and also changing the foreground color to white by setting the ***color*** property to ***white***. There are 100s of style attributes of which we'll cover the most relevant.

### 3.1.2 Styling documents with the ***style*** tag (3pts)

| A slightly better way to configure the look and feel of a Web page is to declare ***CSS rules*** in the ***<style>*** tag in the ***<header>*** and then refer to the tag by their ***selector***, e.g., name, ID, or class. It's better because you can control the style of a whole page from a central place instead of dealing with individual elements. To practice using the ***style*** tag, copy and paste the highlighted style tag below at the end of the ***head*** tag as shown. |  | <**style**>  **selector1** {  **property1**: **value1**;  **property2**: **value2**;  }  </**style**> |
| --- | --- | --- |

| ***/public/labs/a3/index.html*** | |
| --- | --- |
| <**head**>  <**style**>  **p** {  **background-color**: **blue**;  **color**: **white**;  }  </**style**>  </**head**> | *<!-- new style tag*  *selector "p" refers to all paragraphs*  *sets all paragraph background color to blue*  *sets all paragraph foreground color to white*  *-->* |

Then copy the following paragraph at the end of the ***index.html*** document. The paragraph should render with a blue background and white text as shown.

| ***/public/labs/a3/index.html*** | |
| --- | --- |
| <**h3**>Style tag</**h3**>  <**p**>  A slightly better way to configure look and feel  is to declare CSS rules in the STYLE tag in the  header and then refer to the tag by their name,  ID, or class This paragraph's style is set by a  rule referring to the P tags  </**p**> |  |

### 3.1.3 Linking CSS styling documents with the ***link*** tag (4pts)

Although the style tag is slightly better than the style attributes, styling an entire website will entail editing many style tags in their HTML documents. Instead of changing styles inside many HTML documents, it is a ***best practice*** to do all styling configuration in separate CSS files and then link the files from the HTML document with the ***link*** tag. To practice using the ***link*** tag create a brand new file called ***index.css*** in the same directory of the ***index.html*** document, and copy the following content.

| ***/public/labs/a3/index.css*** | |
| --- | --- |
| **p** {  **background-color**: **blue**;  **color**: **white**;  } | */\*This is the same content currently in the style tag.\*/*  */\*We'll use this file from now on for this assignment\*/* |

Then, as shown below, comment out the style tag in ***index.html*** since we won't be using it anymore -- ever. Instead, use the ***link*** tag to load the ***index.css*** file created earlier.

| ***/public/labs/a3/index.html*** | |
| --- | --- |
| <**head**>  *<!--style>*  *p {*  *background-color: blue;*  *color: white;*  *}*  *</style-->*  <**link href="index.css"**  **rel="stylesheet"**/>  </**head**> | *<!--*  *Comment out, or remove, the style tag.*  *We moved this content to the index.css file*  *-->*  *<!-- Instead load the index.css*  *file created earlier -->* |

Finally copy the following content to the end of ***index.html*** which should render a blue and white paragraph as shown.

| ***/public/labs/a3/index.html*** | |
| --- | --- |
| <**h3**>Link tag</**h3**>  <**p**>  The best way to apply CSS rules is to declare  them in a separate CSS file and load it with  the LINK tag. Always use this method.  </**p**> |  |

## 3.2 Selecting content to style with selectors (10pts)

### 3.2.1 Selecting content with ***ID*** selectors (3pts)

The CSS rules in previous exercises styled all paragraphs at once by using the name of the tag ***p*** and then specifying the style property values. Instead of changing the look and feel of all the elements of the same name, e.g., ***p***, we can refer to a specific element by their ID, an attribute specifying a unique identifier. To practice using ID selectors, in ***index.css***, comment out the highlighted paragraph CSS rule as shown and add the two CSS rules referring to paragraphs with IDs ***id-selector-1*** and ***id-selector-2***.

| ***index.css*** | ***index.html*** | ***Browser*** |
| --- | --- | --- |
| */\*p {*  *background-color: blue;*  *color: white;}\*/*  **p#id-selector-1** {  **background-color**: **red**;  **color**: **white**;  }  **p#id-selector-2** {  **background-color**: **yellow**;  **color**: **black**;  } | <**h3**>ID selectors</**h3**>  <**p id="id-selector-1"**>  Instead of changing the look and  feel of all the elements of the  same name, e.g., P, we can refer  to a specific element by its ID  </**p**>  <**p id="id-selector-2"**>  Here's another paragraph using a  different ID and a different look  and feel  </**p**> |  |

### 3.2.2 Selecting content with ***class*** selectors (3pts)

Instead of using IDs to refer to specific elements, you can use an element's ***class*** attribute instead, or a combination of both. Class selectors can be used just like ID selectors if you keep them unique, but can also be used to apply the same style to several elements, even if they are different types of elements. We will be using class selectors exclusively from now on. To practice using class selectors, copy the CSS rule below into ***index.css***, and the HTML at the end of ***index.html***.

| ***index.css*** | ***index.html*** | **Browser** |
| --- | --- | --- |
| .**class-selector** {  **background-color**: **yellow**;  **color**: **blue**;  } | <**h3**>Class selectors</**h3**>  <**p class="class-selector"**>  Instead of using IDs to refer  to elements, you can use an  element's CLASS attribute</**p**>  <**h4 class="class-selector"**>  This heading has same style as  paragraph above</**h4**> |  |

The example above declares a selector that declares a style that transforms the background and foreground color. We can then use the selector to apply the transformation to several elements. The above example applies the style to two elements, the paragraph and the heading.

### 3.2.3 Selecting content based on the document structure (4pts)

Selectors can be combined to refer to elements in particular places in the document. A set of selectors separated by a space can refer to elements in a hierarchy. For instance: ***.selector1 .selector2 { … }*** refers to an element whose class is ***.selector2*** and is inside some ***descendant*** of another element whose class is ***.selector1***. If we use ">" instead to separate the classes, then we can establish a direct parent/child relationship. To practice selecting elements using a set of selectors, copy the following content in ***index.html***

| ***/public/labs/a3/index.html*** | |  |  |
| --- | --- | --- | --- |
| <**div class="selector-1"**>  <**h3**>Document structure selectors</**h3**>  <**div class="selector-2"**>  Selectors can be combined to refer elements in  particular places in the document  <**p class="selector-3"**>  This paragraph's red background is referenced as  <**br**/>.selector-2 .selector3<**br**/>  meaning the descendant of some ancestor.  <**br**/>  <**span class="selector-4"**>Whereas this span is a  direct child of its parent</**span**>  <**br**/>  You can combine these relationships to create  specific styles depending on the document  structure  </**p**>  </**div**>  </**div**> | | | *<!-- this is parent element with selector*  *.selector-1*  *.selector-2 is a direct child of*  *.selector-1*  *.selector-3 is a descendant of*  *.selector-1 and a direct child of*  *.selector-2*  *this is a descendant of .selector-1 and*  *.selector-2 and a direct child of*  *.selector-3*  *-->* |

Let's now style elements .selector-3 and .selector-4. Copy the CSS below into ***index.css***.

| ***/public/labs/a3/index.css*** | | |
| --- | --- | --- |
| .**selector-1** .**selector-3** {  **background-color**: **red**;  **color**: **white**;  }  .**selector-2** > .**selector-3** > .**selector-4** {  **background-color**: **yellow**;  **color**: **blue**;  } | */\* refers to .selector-3 as a descendant of .selector-1 \*/*  */\* refers to .selector-4 as a direct child of .selector-3 \*/*  */\* which is a direct child of .selector-2 \*/* | |

## 3.3 Styling color (5pts)

### 3.3.1 Styling the foreground color (2.5pts)

Foreground colors can be configured using the CSS ***color*** property as follows

| .**some-css-selector** {  **color**: **blue**;  } | */\* selects some DOM element \*/*  */\* sets color property to blue \*/* | |
| --- | --- | --- |

Colors can be defined as follows

* As strings, e.g., white, red, blue, etc
* As hexadecimals, e.g., #ABCDEF
* As RGB, e.g., rgb(12, 34, 56)

Here are a couple examples:

| .**the-sun** { **color**: **rgb**(255,255,0); } | .**the-sky** { **color**: **blue**; } | .**ketchup** { **color**: **#FF0000**; } |
| --- | --- | --- |
|  |  |  |

To practice working with foreground colors, copy the CSS rules below into ***index.css*** to declare several useful color classes. Copy the HTML code below into ***index.html***. Confirm it renders as shown.

| ***index.css*** | ***index.html*** | | |
| --- | --- | --- | --- |
| .**fg-color-black** { **color**: **black**; }  .**fg-color-white** { **color**: **white**; }  .**fg-color-blue** { **color**: **#7070ff**; }  .**fg-color-red** { **color**: **#ff7070**; }  .**fg-color-green** { **color**: **green**; } | <**h2**>Colors</**h2**>  <**h3 class="fg-color-blue"**>Foreground color</**h3**>  <**p class="fg-color-red"**>  The text in this paragraph is red but  <**span class="fg-color-green"**>this text is green</**span**>  </**p**> | | |
|  | | | |

### 3.3.2 Styling the background color (2.5pts)

To practice working with background colors, copy the CSS rules shown below into ***index.css*** and the HTML code into ***index.html***. Confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | |
| --- | --- | --- | --- |
| .**bg-color-yellow** {  **background-color**: **#ffff07**;  }  .**bg-color-blue** {  **background-color**: **#7070ff**;  }  .**bg-color-red** {  **background-color**: **#ff7070**;  }  .**bg-color-green** {  **background-color**: **green**;  } | <**h3 class="bg-color-blue fg-color-white"**>Background color</**h3**>  <**p class="bg-color-red fg-color-black"**>  This background of this paragraph is red but  <**span class="bg-color-green fg-color-white"**> the background  of this text is green and the foreground white</**span**>  </**p**> | | |
|  | | |

## 3.4 Styling the box model (10pts)

### 3.4.1 Styling borders (2.5pts)

Use CSS border properties to configure the look and feel of the border around content. Here's a sample of the properties that can be configured.

| .**some-selector** {  **border-width**: 10**px**;  **border-style**: **solid**|**dotted**|**dashed**|**double**;  **border-color**: **red** | **blue** ...;  } | */\* configure border with several properties\*/*  */\* border's width. Can also provide per border\*/*  */\* the style of the border\*/*  */\* the color of the border \*/* | | |
| --- | --- | --- | --- |

To practice styling borders, copy the CSS code below into ***index.css*** and the HTML code into ***index.html*** and confirm the browser renders as shown.

| ***/public/labs/a3/index.css*** | | | |
| --- | --- | --- | --- |
| .**border-fat** {  **border-width**: 20**px** 30**px** 20**px** 30**px**; }  .**border-thin** {  **border-width**: 4**px**; }  .**border-solid** {  **border-style**: **solid**; }  .**border-dashed** {  **border-style**: **dashed**; }  .**border-yellow** {  **border-color**: **#ffff07**; }  .**border-red** {  **border-color**: **#ff7070**; }  .**border-blue** {  **border-color**: **#7070ff**; } | <**h2**>Borders</**h2**>  <**p class="border-fat border-red border-solid"**>  Solid fat red border</**p**>  <**p class="border-thin border-blue border-dashed"**>  Dashed thin blue border</**p**> | | |
|  | | |

### 3.4.2 Stying margins and paddings (5pts)

You can also configure the spacing between elements. To practice with paddings, copy the CSS code below into ***index.css*** and the HTML into ***index.html***. Confirm browser renders as shown.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**padded-top-left** {  **padding-top**: 50**px**;  **padding-left**: 50**px**;  }  .**padded-bottom-right** {  **padding-bottom**: 50**px**;  **padding-right**: 50**px**;  }  .**padding-fat** {  **padding**: 50**px**;  } | <**h2**>Padding</**h2**>  <**div class="padded-top-left border-fat**  **border-red border-solid**  **bg-color-yellow"**>  Padded top left  </**div**>  <**div class="padded-bottom-right border-fat**  **border-blue border-solid**  **bg-color-yellow"**>  Padded bottom right  </**div**>  <**div class="padding-fat border-fat**  **border-yellow border-solid**  **bg-color-blue fg-color-white"**>  Padded all around  </**div**> | | |  |

To practice with margins, copy the CSS code below into ***index.css*** and the HTML into ***index.html***. Confirm browser renders as shown.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**margin-bottom** {  **margin-bottom**: 50**px**;  }  .**margin-right-left** {  **margin-left**: 50**px**;  **margin-right**: 50**px**;  }  .**margin-all-around** {  **margin**: 30**px**;  } | <**h2**>Margins</**h2**>  <**div class="margin-bottom**  **padded-top-left**  **border-fat border-red**  **border-solid**  **bg-color-yellow"**>  Margin bottom</**div**>  <**div class="margin-right-left**  **padded-bottom-right**  **border-fat border-blue**  **border-solid**  **bg-color-yellow"**>  Margin left right  </**div**>  <**div class="margin-all-around**  **padding-fat border-fat**  **border-yellow**  **border-solid**  **bg-color-blue**  **fg-color-white"**>  Margin all around  </**div**> | | |  |

### 3.4.3 Stying corners (2.5pts)

You can configure the corners of element borders to be rounded. Either all of them at once or specific corners. You can do this by configuring a border's radius. To practice rounding some corners, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**rounded-corners-bottom** {  **border-bottom-left-radius**: 40**px**;  **border-bottom-right-radius**: 40**px**;  }  .**rounded-corners-all-around** {  **border-radius**: 50**px**;  }  .**rounded-corners-inline** {  **border-radius**: 30**px** 0**px** 20**px** 50**px**;  } | <**h3**>Rounded corners</**h3**>  <**p class="rounded-corners-top border-thin border-blue border-solid padding-fat"**>  Rounded corners on the top  </**p**>  <**p class="rounded-corners-bottom border-thin border-blue border-solid padding-fat"**>  Rounded corners at the bottom  </**p**>  <**p class="rounded-corners-all-around border-thin border-blue border-solid padding-fat"**>  Rounded corners all around  </**p**>  <**p class="rounded-corners-inline border-thin border-blue border-solid padding-fat"**>  Different rounded corners  </**p**> | | |  |

## 3.5 Styling dimensions (3pts)

You can configure an element's dimensions with ***width*** and ***height*** properties. To practice setting element's dimensions, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**dimension-portrait** {  **width**: 75**px**;  **height**: 100**px**;  }  .**dimension-landscape** {  **width**: 100**px**;  **height**: 75**px**;  }  .**dimension-square** {  **width**: 75**px**;  **height**: 75**px**;  } | <**h2**>Dimension</**h2**>  <**div**>  <**div class="dimension-portrait bg-color-yellow"**>  Portrait  </**div**>  <**div class="dimension-landscape bg-color-blue**  **fg-color-white"**>  Landscape  </**div**>  <**div class="dimension-square bg-color-red"**>  Square  </**div**>  </**div**> | | |  |

## 3.6 Styling position (12pts)

You can configure an element's position with the ***position*** property. The property has many possible values, but we'll explore ***relative***, ***position***, and ***static***.

### 3.6.1 Styling relative position (2pts)

Setting ***position*** property to ***relative*** allows moving the element relative to its original position. To practice setting element's relative position, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**pos-relative-nudge-up-right** {  **position**: **relative**;  **bottom**: 30**px**;  **left**: 30**px**;  }  .**pos-relative-nudge-down-right** {  **position**: **relative**;  **top**: 20**px**;  **left**: 20**px**;  }  .**pos-relative** {  **position**: **relative**;  } | <**h2**>Position</**h2**>  <**h3**>Relative</**h3**>  <**div**>  <**div class="bg-color-yellow**  **dimension-portrait"**>  <**div class="pos-relative-nudge-down-right"**>  Portrait</**div**>  </**div**>  <**div class="pos-relative-nudge-up-right**  **bg-color-blue fg-color-white**  **dimension-landscape"**>  Landscape</**div**>  <**div class="bg-color-red dimension-square"**>  Square</**div**>  </**div**> | | |  |

### 3.6.2 Styling absolute position (2pts)

Setting ***position*** property to ***absolute*** allows moving the element relative to the position of its parent. To practice setting element's absolute position, copy the CSS and HTML below into ***index.css*** and ***index.html***. Notice several ***<br/>*** elements were added at the end of the example to make room for the next exercise.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**pos-absolute-10-10** {  **position**: **absolute**;  **top**: 10**px**;  **left**: 10**px**;  }  .**pos-absolute-50-50** {  **position**: **absolute**;  **top**: 50**px**;  **left**: 50**px**;  }  .**pos-absolute-120-20** {  **position**: **absolute**;  **top**: 20**px**;  **left**: 120**px**; } | <**h3**>Absolute position</**h3**>  <**div class="pos-relative"**>  <**div class="pos-absolute-10-10**  **bg-color-yellow dimension-portrait"**>  Portrait  </**div**>  <**div class="pos-absolute-50-50**  **bg-color-blue fg-color-white**  **dimension-landscape"**>  Landscape </**div**>  <**div class="pos-absolute-120-20**  **bg-color-red dimension-square"**>  Square </**div**>  </**div**><**br**/><**br**/><**br**/><**br**/><**br**/><**br**/><**br**/> | | |  |

### 3.6.3 Styling fixed position (1pt)

Setting ***position*** property to ***fixed*** allows setting the element relative to the window. That means that if you scroll the content of the page, the element will not scroll with it. To practice setting element's fixed position, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown. Your display may be different depending on the actual size of the screen and scrolling.

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**pos-fixed** {  **position**: **fixed**;  **right**: 0**px**;  **bottom**: 50%;  } | <**h3**>Fixed position</**h3**>  Checkout the blue square that says  "Fixed position" stuck all the way  on the right and half way down the  page. It doesn't scroll with the  rest of the page. Its position is  "Fixed".  <**div class="pos-fixed**  **dimension-square**  **bg-color-blue**  **fg-color-white"**>  Fixed position  </**div**> | | |  |

### 3.6.4 Styling z-index (1pt)

When the browser renders content declared in HTML documents, it calculates positions and dimensions so every element has a dedicated rectangle on the window. Typically elements don't fall on top of each other. When you start moving elements with ***position***, then overlapping elements are possible. By default elements are rendered in the order declared in HTML documents. Elements declared later render on top of elements declared earlier. The ***z-index*** CSS property overrides this behavior. Default value of ***z-index*** is ***auto***, which corresponds to 0. Increasing z-index can make elements render later, or on top of, others. To practice setting an element's ***z-index***, copy the CSS and HTML below into ***index.css*** and ***index.html***

| ***index.css*** | ***index.html*** | | | ***Browser*** |
| --- | --- | --- | --- | --- |
| .**zindex-bring-to-front** {  **z-index**: 10;  } | <**h2**>Z index</**h2**>  <**div class="pos-relative"**>  <**div class="pos-absolute-10-10**  **bg-color-yellow**  **dimension-portrait"**>  Portrait</**div**>  <**div class="zindex-bring-to-front**  **pos-absolute-50-50**  **bg-color-blue fg-color-white**  **dimension-landscape"**>  Landscape</**div**>  <**div class="pos-absolute-120-20**  **bg-color-red dimension-square"**>  Square</**div**>  </**div**><**br**/><**br**/><**br**/><**br**/><**br**/><**br**/><**br**/> | | | **Z index** |

### 3.6.5 Floating content (2pts)

HTML does not support laying out content horizontally. The CSS float property allows fixing that. To practice laying out content horizontally, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | | |
| --- | --- | --- | --- | --- |
| .**float-left** {  **float**: **left**;  }  .**float-right** {  **float**: **right**;  **height**: 100**px**;  }  .**float-done** {  **clear**: **both**;  } | <**h2**>Float</**h2**>  <**div**>  <**div class="float-left dimension-portrait bg-color-yellow"**>  Yellow</**div**>  <**div class="float-left dimension-portrait bg-color-blue fg-color-white"**>  Blue</**div**>  <**div class="float-left dimension-portrait bg-color-red"**>  Red</**div**>  <**img class="float-right" src="https://www.staradvertiser.com/wp-content/uploads/2021/08/web1\_Starship-gap2.jpg"**/>  <**div class="float-done"**></**div**>  </**div**> | | | |
|  | | | | |

### 3.6.6 Laying out content in a grid (4pts)

Using float we can implement a grid layout made up of rows and columns. To practice laying out content in a grid, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

| ***index.css*** | ***index.html*** | | | |
| --- | --- | --- | --- | --- |
| .**grid-row** {  **clear**: **both**;  }  .**grid-col-half-page** {  **width**: 50%;  **float**: **left**;  } | <**h2**>Grid layout</**h2**>  <**div class="grid-row"**>  <**div class="grid-col-half-page bg-color-yellow"**>  <**h3**>Left half</**h3**>  </**div**>  <**div class="grid-col-half-page bg-color-blue**  **fg-color-white"**>  <**h3**>Right half</**h3**>  </**div**>  </**div**> | | | |

| .**grid-col-third-page** {  **width**: 33%;  **float**: **left**;  }  .**grid-col-two-thirds-page** {  **width**: 67%;  **float**: **left**;  }  .**grid-col-left-sidebar** {  **width**: 20%;  **float**: **left**;  }  .**grid-col-main-content** {  **width**: 60%;  **float**: **left**;  }  .**grid-col-right-sidebar** {  **width**: 20%;  **float**: **left**;  } | <**div class="grid-row"**>  <**div class="grid-col-third-page bg-color-green**  **fg-color-white"**>  <**h3**>Left third</**h3**>  </**div**>  <**div class="grid-col-two-thirds-page bg-color-red**  **fg-color-white"**>  <**h3**>Right two thirds</**h3**>  </**div**>  </**div**>  <**div class="grid-row"**>  <**div class="grid-col-left-sidebar bg-color-yellow"**>  <**h3**>Side bar</**h3**>  <**p**>This is the left sidebar</**p**>  </**div**>  <**div class="grid-col-main-content bg-color-blue**  **fg-color-white"**>  <**h3**>Main content</**h3**>  <**p**>This is the main content. This is the main content.  This is the main content. </**p**>  </**div**>  <**div class="grid-col-right-sidebar bg-color-green**  **fg-color-white"**>  <**h3**>Side bar</**h3**>  <**p**>This is the right sidebar</**p**>  </**div**>  </**div**> | | | |
| --- | --- | --- | --- | --- |
|  | | | | |

## 3.7 Exercises (50pts)

1. Setup styling as described in [section 3.1](#_ba1ouok13e1f) (10pts)
2. Implement the selectors as described in [section 3.2](#_olc3meh7mecg) (10pts)
3. Implement the color styling as described in [section 3.3](#_slee3xd5i6bn) (5pts)
4. Style the box model as described in [section 3.4](#_czwkixyssu0) (10pts)
5. Style dimensions as described in [section 3.5](#_emq3l8l6uza6) (3pts)
6. Style positions as described in [section 3.6](#_5957uk3efxes) (12pts)

# 4 Tuiter (50pts)

For the tuiter section of this assignment you are going to continue implementing the ***Tuiter*** website started in previous assignments. You will create and style two new Web pages ***explore*** and ***bookmarks***. Continue your work in the directory ***public/tuiter***.

## 4.1 Styling the explore screen (25pts)

To practice what you've learned in the ***lab*** section, you'll apply those skills to create Tuiter's ***explore*** screen on your own. The explore screen lists random tuits based on your history and trends. In a previous assignment you created the ***navigation.html*** Webpage containing several links. One of those links referenced ***explore.html***. Create a new Web page in ***public/tuiter/explore/index.html*** and edit the link to refer to this new document. The requirements and wireframe for the ***explore*** screen are shown below under ***Requirements*** and ***Wireframe***. Do all your styling in ***public/tuiter/explore/index.css*** and link to it from ***index.html*** in the directory. Ignore icons, images, etc. that are not mentioned or described. You will not lose points for them. All sizes shown are approximate and suggestions on what the rough distance between elements should be. The intent is to make your webpage look as close as possible to the wireframe provided. If a size is not explicitly mentioned, you must still make an effort to make your content look as close to the wireframe as possible. Background color is black. Borders are rgb(47, 51, 54). All classes must be prefixed with "wd-". All content below is offset 16px from the left border. The letters A, B, C, etc. are used to reference different parts of the wireframe and don't need to be implemented, only the content they refer to. The wireframe shows at least one example of different types of posts labeled C, D, and E. For this assignment, create at least 2 examples of each. Grab dummy content from <https://www.lipsum.com/>, from real posts and articles, or create your own content.

| ***Requirements*** | ***Wireframe*** |
| --- | --- |
| 1. At the top there's an input field with rounded corners and a placeholder that reads "Search Tuiter". The color of the text is white and the background color is dark gray. 12px from the right border there's a link in the form of a gear that references "explore-settings.html" 2. Below the search input field there are a set of hyperlinks labeled "For you", "Trending", etc. The first link is highlighted with RGB color 29,161,242. The border below the highlighted link is the same color. All other links are color 110, 118, 125. The link fonts are 15px and padded all around by 16px. 3. Below the links there are suggested posts. The suggested posts have the topic, title and number of tuits. The topic, "Trending in Science", "Sports Tranding", "Trending in United States", have 13px font and color 110, 118, 125 4. Some suggested posts additionally have the name of the author of the post, "The New York Times" in white and an image with rounded |  |
| corners all the way to the right. The image is 100px wide and high. The image is 16px from the right border. | |

1. Some suggested posts don't have the name of the author of the post. Instead they have the topic of the post, e.g., "COVID-19" in light gray and an image with rounded corners all the way to the right. The image is 100px wide and high. The image is 16px from the right border.

## 4.2 Styling Bookmarks Screen (25pts)

Now let's style the ***bookmarks*** screen. Create another HTML document in ***public/tuiter/bookmarks/index.html*** and style it as shown below. Implement all your styling in ***public/tuiter/bookmarks/index.css***. If there is styling that is the same in multiple Web pages, implement the styling in ***public/tuiter/styles.css***. If a size is not explicitly mentioned you must still make an effort to make your content look as close as possible to the wireframe shown on the right. Background color is black. Borders are rgb(47, 51, 54). All classes must be prefixed with "wd-". The wireframe shows a single bookmarked post. Implement at least two of these posts for this assignment. Grab dummy content from <https://www.lipsum.com/>, from real posts and articles, or create your own content.

| 1. 16px from the left border and 12px from the top edge, there's a title at the top that says "Bookmarks" as shown in the wireframe. The color of the title is white and the size of the font is 20px. Below the title is the handle of the user "@WebDev" who bookmarked the posts listed in this screen. The color of the handle is light gray and the font size is 13px. There are 16px between the title and its closest left edge. 12px below the handle there's a light gray solid 1px border. 2. 16px from the left border and 12px below the title and handle's border there's a circular "avatar" image that represents the user who posted the content originally. The image has a diameter of 48px. 3. 12px to the right of the avatar image and 12px below the title and handle's border, we find the name of the author of the post "Robert Zubrin" followed by their handle "@robert\_zubrin". The font size of the author and their handle is 15px. The author color is white and the handle is light gray. There's a date after the handle. 4. Below the author's name we find the main content of the post in color white and font size 15px. The width of the content can not exceed 506px. The text is 16px away to the left of the right edge and 12px above the image E described below. |  |
| --- | --- |

1. 12px below the main content there is an image 504px by 264px in size. The edges of the image are rounded at the top corners. The image is 12px to the right of the avatar image.
2. 12px below the image there's a white bold title with 15px font size. Below the title there's text with the same size but light gray in color. The title and text are padded on all sides by 12px. The bottom of the text is rounded in the corners. The image E and text below it have a thin border of 1px in width and light gray color.
3. 12px below the text there are 4 icons with numbers next to them. Each pair of icons and numbers makes a hyperlink pointing to "#". The links are equally spaced out. There are 12px between the icons and the numbers. The heart icon and number next to it is the color red. All other icons and numbers are light gray.

## 4.3 Exercises

Complete the tuiter exercises described in section 4

1. Implement the explore screen as described in [section 4.1](#_cc8l5ok9a1wd) (25pts)
2. Implement the explore screen as described in [section 4.2](#_txmp5tugix7g) (25pts)

# 

# 5 Challenge (50pts required for graduate students)

The CSS attribute ***transform*** provides some powerful transformations such as ***rotate***. To practice with this attribute, copy the HTML and CSS below to rotate a paragraph as shown below. Keep doing your work in ***public/labs/a3/index.html*** and ***index.css***.

| ***public/labs/a3/index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h2**>Rotations</**h2**>  <**h3**>Rotating a paragraph</**h3**>  <**p class="rotate-paragraph"**>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</**p**> | .**rotate-paragraph** {  **width**: 250**px**;  **height**: 250**px**;  **transform**: **rotate**(90**deg**);  } |  |

## 5.1 Rotate a list (5pts)

Any element can be rotated. Here's an example that rotates an entire list of items. Copy and paste the code below. Also, create a different list of ordered items and rotate it counter clockwise.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Rotating a list</**h3**>  <**ul class="rotate-list"**>  <**li**>Item 1</**li**>  <**li**>Item 2</**li**>  <**li**>Item 3</**li**>  </**ul**> | .**rotate-list** {  **transform**: **rotate**(12**deg**);  **width**: 100**px**;  **background-color**: **red**;  **color**: **white**;  } |  |

## 5.2 Rotating a list item (5pts)

Instead of rotating an entire list, we could instead rotate individual line items of the list. Positive degrees rotate elements clockwise and negative degrees rotate them counterclockwise. Copy and paste the code below to rotate individual line items within a list.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Rotating list items</**h3**>  <**ul class="rotate-list-items"**>  <**li**>Item 1</**li**>  <**li**>Item 2</**li**>  <**li**>Item 3</**li**>  </**ul**> | .**rotate-list-items li** {  **width**: 100**px**;  **text-align**: **right**;  }  .**rotate-list-items li**:**nth-child**(1) {  **transform**: **rotate**(-10**deg**);  **background-color**: **yellow**;  }  .**rotate-list-items li**:**nth-child**(2) {  **transform**: **rotate**(2**deg**);  **background-color**: **blue**;  **color**: **white**;  }  .**rotate-list-items li**:**nth-child**(3) {  **transform**: **rotate**(10**deg**);  **background-color**: **red**;  } |  |

## 5.3 Rotating images (5pts)

Here's an example of applying rotation to an image element. Copy and paste the image shown below. The image shows the performance of the Dow Jones in the last 100 years. Apparently the stock market has had wild swings over the year, but maintained an average 10% growth over the long run. Create a similar example using an image of your own.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Rotating an image</**h3**>  <**img class="rotate-image" width="400px"**  **src="https://s14085.pcdn.co/wp-content/uploads/2019/10/dowjones\_chart\_100\_years\_20200327.png"**/> | .**rotate-image** {  **transform**: **rotate**(-10**deg**);  } |  |

## 5.4 Rotate a table (5pts)

We can also rotate an entire table as shown below. Copy and paste the code below. The table shows how money grows over time for various types of interest and monthly deposits for a 40 year period, an average working lifespan. Apparently putting away $500 a month away for 40 years at 10% growth (average stock market), will net you $2.7 million. Wow. I should have saved money. Oh well, too late for me, but as they say, the best time to start saving was 20 years ago, the next best time to start saving is today.

| ***public/labs/a3/index.html*** | | ***index.css*** |
| --- | --- | --- |
| <**h3**>Rotating a table</**h3**>  <**br**/><**br**/>  <**table border="1" class="rotate-table"**>  <**tr**><**td**>Return rate</**td**><**td**>Monthly contributions</**td**>  <**td**>Balance after 40 years</**td**></**tr**>  <**tr**><**td**>5</**td**><**td**>250</**td**><**td**>372,141.15</**td**></**tr**>  <**tr**><**td**>5</**td**><**td**>500</**td**><**td**>744,282.29</**td**></**tr**>  <**tr**><**td**>5</**td**><**td**>1000</**td**><**td**>1,488,564.58</**td**></**tr**>  <**tr**><**td**>7</**td**><**td**>250</**td**><**td**>621,379.12</**td**></**tr**>  <**tr**><**td**>7</**td**><**td**>500</**td**><**td**>1,242,758.23</**td**></**tr**>  <**tr**><**td**>7</**td**><**td**>1000</**td**><**td**>2,485,516.46</**td**></**tr**>  <**tr**><**td**>10</**td**><**td**>250</**td**><**td**>1,398,651.85</**td**></**tr**>  <**tr**><**td**>10</**td**><**td**>500</**td**><**td**>2,797,303.70</**td**></**tr**>  <**tr**><**td**>10</**td**><**td**>1000</**td**><**td**>5,594,607.40</**td**></**tr**>  <**tr**><**td**>13</**td**><**td**>250</**td**><**td**>3,251,260.21</**td**></**tr**>  <**tr**><**td**>13</**td**><**td**>500</**td**><**td**>6,502,520.42</**td**></**tr**>  <**tr**><**td**>13</**td**><**td**>1000</**td**><**td**>13,005,040.83</**td**></**tr**>  </**table**> | | .**rotate-table** {  **position**: **relative**;  **left**: 50**px**;  **transform**: **rotate**(15**deg**);}  .**rotate-table tr**:**nth-child**(1) {  **background-color**: **black**;  **color**: **white**;}  .**rotate-table tr**:**nth-child**(**n**+2) {  **background-color**: **yellow**;}  .**rotate-table tr**:**nth-child**(**n**+5) {  **background-color**: **blue**;  **color**: **white**;}  .**rotate-table tr**:**nth-child**(**n**+8) {  **background-color**: **red**;}  .**rotate-table tr**:**nth-child**(**n**+11) {  **background-color**: **green**;  **color**: **white**;  } |
|  | | |

## 5.5 Gradients (10pts)

Up to this point we've been painting backgrounds with solid colors. CSS allows applying different types of gradients when working with collors. Let's explore the simplest type of gradient -- linear gradients. To practice linear gradients copy the following code and verify the browser renders as shown. Create your own example based on the example provided using your own colors.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Linear gradients</**h3**>  <**p class="gradients-linear"**>  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.</**p**> | .**gradients-linear** {  **background**: **linear-gradient**(**yellow**, **red**);  **width**: 200**px**;  **height**: 200**px**;  } |  |

### 5.5.1 Radial gradients

To practice radial gradients copy the following code and verify the browser renders as shown. Create your own example based on the example provided using your own colors.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Radial gradients</**h3**>  <**p class="gradients-radial"**>  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.</**p**> | .**gradients-radial** {  **background**: **radial-gradient**(**yellow**, **green**);  **width**: 200**px**;  **height**: 200**px**;  } |  |

### 

### 5.5.2 Positioning radial gradients

Some cool effects can be achieved by moving the position of the center of radial gradients. To practice changing the position of a radial gradient, copy the following code and verify the browser renders as shown. Create your own example based on the example provided using different colors and other positions.

| ***index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h3**>Positioning radial gradients</**h3**>  <**p class="gradients-positioned"**>  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.</**p**> | .**gradients-positioned** {  **background**:  **radial-gradient**(  **at top left**, **blue**, **red**);  **width**: 200**px**;  **height**: 200**px**;  **color**: **white**;  } |  |

## 5.6 Animations (5pts)

### 5.6.1 Key frame animations

CSS can control some fairly complex animations by specifying several CSS transformations for several key frames. The browser can then interpolate between the different transformations along a specified unit of time. To practice animation, copy the following code and verify the browser renders as shown. Create your own example based on the example provided. Use your own paragraph, sizes, keyframes, colors, and font sizes. In your own examples, play around with the following attributes

* **animation-iteration-count: infinite;**
* **animation-direction: alternate;**

| ***index.html*** | ***index.css*** | |
| --- | --- | --- |
| <**h2**>Animations</**h2**>  <**h3**>Keyframe animations</**h3**>  <**p class="animation-keyframes"**>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.</**p**> | .**animation-keyframes** {  **position**: **relative**;  **top**: 0**px**;  **animation-duration**: 7**s**;  **animation-name**: **slidein**;  **width**: 200**px**;  **height**: 200**px**;  **background-color**: **red**;  **color**: **white**;  }  **@keyframes slidein** {  **from** {  **position**: **relative**;  **top**: 0**px**;  **font-size**: 100%;  **margin-left**: 100%;  **width**: 200**px**;  **height**: 200**px**;  **background-color**: **yellow**;  **color**: **black**;  } */\* continued on right \*/* | **50%** {  **position**: **relative**;  **top**: -200**px**;  **font-size**: 300%;  **margin-left**: 25%;  **width**: 400**px**;  **height**: 400**px**;  **background-color**: **blue**;  **color**: **white**;  }  **to** {  **position**: **relative**;  **top**: 0**px**;  **font-size**: 100%;  **margin-left**: 0%;  **width**: 200**px**;  **height**: 200**px**;  **background-color**: **red**;  **color**: **white**;  }  } |
|  | | |

## 5.7 Flex (10pts)

Flex is a set of styles that allows you to quickly layout a Web Page. Let's first define a simple set of block elements as shown below.

| ***index.css*** | ***index.html*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  }  .**flex-box** {  **border**: 5**px gray solid**;  **margin**: 10**px**;  **padding**: 10**px**;  }  .**flex-box-1** {  **background-color**: **yellow**;  }  .**flex-box-2** {  **background-color**: **blue**;  **color**: **white**;  }  .**flex-box-3** {  **background-color**: **red**;  **color**: **white**;  } | <**div class="flex-container"**>  <**div class="flex-box flex-box-1"**>Box 1</**div**>  <**div class="flex-box flex-box-2"**>  Box 2 Box 2<**br**/>Box 2 Box 2  </**div**>  <**div class="flex-box flex-box-3"**>  Box 3 Box 3 Box 3<**br**/>Box 3 Box 3 Box 3<**br**/>Box 3 Box 3 Box 3  </**div**>  </**div**> | |
|  | |

### 5.7.1 Aligning flex elements horizontally

Flex features are initialized by setting the value of the ***display*** property to ***flex*** as shown below. The flex property applies to the elements contained in the parent. Here, flex is applied to box 1, 2, and 3 contained in ***.flex-container***. By default flex elements are aligned to the left in their container..

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  } |  | |

Elements within a flex container can be ***justified*** in many ways such as at the ***start***, ***center***, and ***end***. Additionally you can add ***space-around*** or ***between*** the elements. Here are examples of using ***center***, ***end***, ***space-around*** and ***space-between*** justifications. In your HTML and CSS files, create independent examples that illustrate each of the following justification properties.

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **center**;  } |  | |

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **end**;  } |  | |

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **space-around**;  } |  | |

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **space-between**;  } |  | |

### 5.7.2 Aligning flex elements vertically

Flex elements can also be justified vertically with the ***align-items*** property. Setting the ***align-items*** property to ***flex-start*** aligns the elements at the top of their container. Setting it to ***center*** aligns the elements vertically in the middle of the container, and ***flex-end*** aligns the elements vertically at the bottom of the container. Create independent examples of all three ***align-items*** property values presented here.

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **space-evenly**;  **align-items**: **flex-start**;  } |  | |

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  **justify-content**: **space-evenly**;  **align-items**: **center**;  } |  | |

Flex elements can also be configured to stretch evenly horizontally and vertically to fill their container. Create an independent example that illustrates this feature as shown below.

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| .**flex-container** {  **border-width**: 5**px**;  **border-color**: **black**;  **border-style**: **solid**;  **display**: **flex**;  */\*justify-content: space-evenly;\*/*  */\*align-items: center;\*/*  }  .**flex-container** > .**flex-box** {  **flex-basis**: 100%;  } |  | |

## 5.8 Media queries (5pts)

Media queries allow browsers to adapt their rendering for specific media types such as printed media, screens, projectors, etc. Media output can be configured to use certain styles for different screen sizes. The examples below illustrate how we can apply a certain style for a specific window size range. In the example below we set the body's and paragraph's background color based on the size of the window

| ***public/labs/a3/index.html*** | ***index.css*** | ***Browser*** |
| --- | --- | --- |
| <**h2**>Media queries</**h2**>  <**p class="media-query"**>  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.  </**p**> | **@media** (**min-width**: 0**px**)  **and** (**max-width**: 600**px**) {  **body** {  **background-color**: **yellow**;  }  .**media-query** {  **background-color**: **red**;  **color**: **white**;  }  } |  |

If we add additional media rules we can trigger additional styles for different screen sizes. The example bellow toggles background and foreground colors based on the size of the screen

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| **@media** (**min-width**: 600**px**)  **and** (**max-width**: 1200**px**) {  **body** {  **background-color**: **blue**;  **color**: **white**;  }  .**media-query** {  **background-color**: **yellow**;  **color**: **black**;  }  } | ***Screen size 0 to 600px:***    ***Screen size 600 to 1200px:*** | |
| ***Screen size > 1200px (default)*** | | |

Adding a third media rule breaks up the styles applied based on 4 different screen size ranges: 0-600px, 600-1200px, 1200-1800px, and larger than 1800px.

| ***index.css*** | ***Browser*** | |
| --- | --- | --- |
| **@media** (**min-width**: 0**px**)  **and** (**max-width**: 600**px**) {  **body** {  **background-color**: **yellow**;  }  .**media-query** {  **background-color**: **red**;  **color**: **white**;  }  }  **@media** (**min-width**: 600**px**)  **and** (**max-width**: 1200**px**) {  **body** {  **background-color**: **blue**;  **color**: **white**;  }  .**media-query** {  **background-color**: **yellow**;  **color**: **black**;  }  }  **@media** (**min-width**: 1200**px**)  **and** (**max-width**: 1800**px**) {  **body** {  **background-color**: **red**;  **color**: **white**;  }  .**media-query** {  **background-color**: **blue**;  **color**: **white**;  }  } | ***Screen size 0 to 600px:***    ***Screen size 600 to 1200px:***    ***Screen size > 1200px (default)*** | |

Create examples of your own that define 3 different media rules for similar screen sizes. Define a couple of elements, e.g., heading, paragraph, divs, and change their style as the user resizes the screen

## 

## 5.9 Exercises

Complete the challenge exercises described in section 5 (50pts):

1. Implement the rotated list exercise as described in [section 5.1](#_b9z08qwdb1k3) (5pts)
2. Implement the rotated list items exercise as described in [section 5.2](#_1rab7knirjkv) (5pts)
3. Implement the rotated images exercise as described in [section 5.3](#_fprwh33qhc6k) (5pts)
4. Implement the rotated table exercise as described in [section 5.4](#_7av6d3g0ksy8) (5pts)
5. Implement the gradients exercise as described in [section 5.5](#_18od9phjuqze) (10pts)
6. Implement the animation exercise as described in [section 5.6](#_m6pts0flywda) (5pts)
7. Implement the flex exercise as described in [section 5.7](#_h8rf049lh444) (10pts)
8. Implement the media queries exercise as described in [section 5.8](#_la4hvractost) (5pts)

# 6 Delivery

1. In the same React.js application created in earlier assignments, ***tuiter-react-web-app***, complete all the exercises described in this document
2. In a branch called ***a3***, add, commit and push the source code of the React.js application ***tuiter-react-web-app*** to the same remote source repository in ***GitHub.com*** created in an earlier assignment. Here's an example of how to add, commit and push your code

| **$ git checkout -b a3**  **$ git add .**  **$ git commit -am "a3 css fall 2022"**  **$ git push** |
| --- |

1. Deploy the ***a3*** branch to the same ***Netlify*** created in an earlier assignment. Configure Netlify to deploy all branches to separate URLs. From your Netlify's dashboard go to ***Site settings > Build & deploy > Branches > Branch deploys*** and select ***All***. Now each time you commit to a branch, the application will be available at a URL that contains the name of the branch
2. As a deliverable in ***Canvas***, submit the following URLs
   1. The source repository in GitHub.com. It should look something like  
      **https://github.com/jannunzi/tuiter-react-web-app.git**
   2. The React.js application running on Netlify. It should look something like  
        
      [**https://a3-loving-torvalds-effde8.netlify.app/**](https://a3-loving-torvalds-effde8.netlify.app/)  
        
      Note that the URL contains the name of the branch