Notes written by: Abdulrahman AlQallaf

Last modification date: 27-Jul-2016

Remarks:

- These notes are primarily written after completing the “Introduction to CSS3” course offered the University of Michigan on Coursera. I have incorporated information from other sources in some sections to help me better understand the material. These notes are based on my understanding and they are intended to be used as a summary for me to go back to from time to time. Anyone reading this should not rely on my summary and should always revert back to the original sources.
- This document does not contain all the information necessary to review the course, refer back to the code written and the additional resources provided while taking this course.

Week 1

- CSS rules are applied in following order / priority (low to high)
    - Browser default
    - External style sheet
    - Internal style
    - Inline style
    - The most recent rule has precedence

⇨ Note: to override later rules, use "!important"

- CSS syntax uses the following format

⇨ Note: for the advanced selector, use (http://learn.shayhowe.com/advanced-html-css/complex-selectors/) as a reference.

- You can specify a color in CSS by using
    - Color name
    - Hex value
    - Rgb (red, green, blue)
    - Rgba (red, green, blue, alpha – used for opacity)

⇨ Note: you can use inspect element in chrome to select the color and see the changes live

⇨ Note: to check if you have a good color contrast in your website, use a contrast checker such as (http://wave.webaim.org/)

- Fonts
  - Font families are styles of text (e.g., verdana, Helvetica, …)
  - To let the browser fall back to a different font family if one is not supported, do the following:

```
<style>
    font-family: verdana, arial;
</style>
```

  - To expand beyond web-safe fonts, use @font-face:

```
<style type="text/css">
    @font-face {
        font-family: mySpecialFont;
        src: url('colleen.ttf'); /* TTF is a file extension for a font file developed by Apple. TTF stands for True Type Font. */
    }

    h1 {
        font-family: mySpecialFont;
    }
</style>
```

  - Font style
    - Normal
    - Italic
    - Oblique
  - Font variant
    - Normal
    - Small caps
  - Text transform
    - Capitalize
    - Uppercase
    - Lowercase
  - Font size
    - xx-small, x-small, small, smaller
    - medium
    - larger, x-large, xx-large, larger
    - use pixel
  - Text align
    - left
    - right
    - center
    - justify
  - Adjust the space between lines
    - Use “line-height”

- Every element is a box
- Elements can have the following display values
    - block (e.g., div)
    - inline (e.g., span)
    - inline-block (same as inline, but accepts height and width properties)
    - none (removed from page)
- Visibility values:
    - visible
    - hidden
    - collapse
- Note on visibility vs. display

> `display:none` means that the tag in question will not appear on the page at all (although you can still interact with it through the dom). There will be no space allocated for it between the other tags.
>
> `visibility:hidden` means that unlike `display:none`, the tag is not visible, but space is allocated for it on the page. The tag is rendered, it just isn't seen on the page.

- Floating elements
    - Float property: elements are aware of one another and will not overlap (if the position property wasn't changed)
        - Left
        - Right
    - Clear property: used to keep floating elements from overlapping
        - Left
        - Right
        - Both
- Element overflow can happen when the content is bigger than the container they are in. You can set the following for the overflow property:
    - visible
    - hidden
    - scroll
    - auto (add scrollbar as needed)

Week 2

- Border:
  - Any element can a have a border around it
  - Properties:
    - style (i.e., border-style)
      - none
      - dotted
      - dashed
      - solid
      - double
      - groove
      - ridge
      - inset
      - outset
      - hidden
    - width
      - pixel
      - thin
      - medium
      - large
    - color
      - name
      - rgb
      - hex
      - transparent

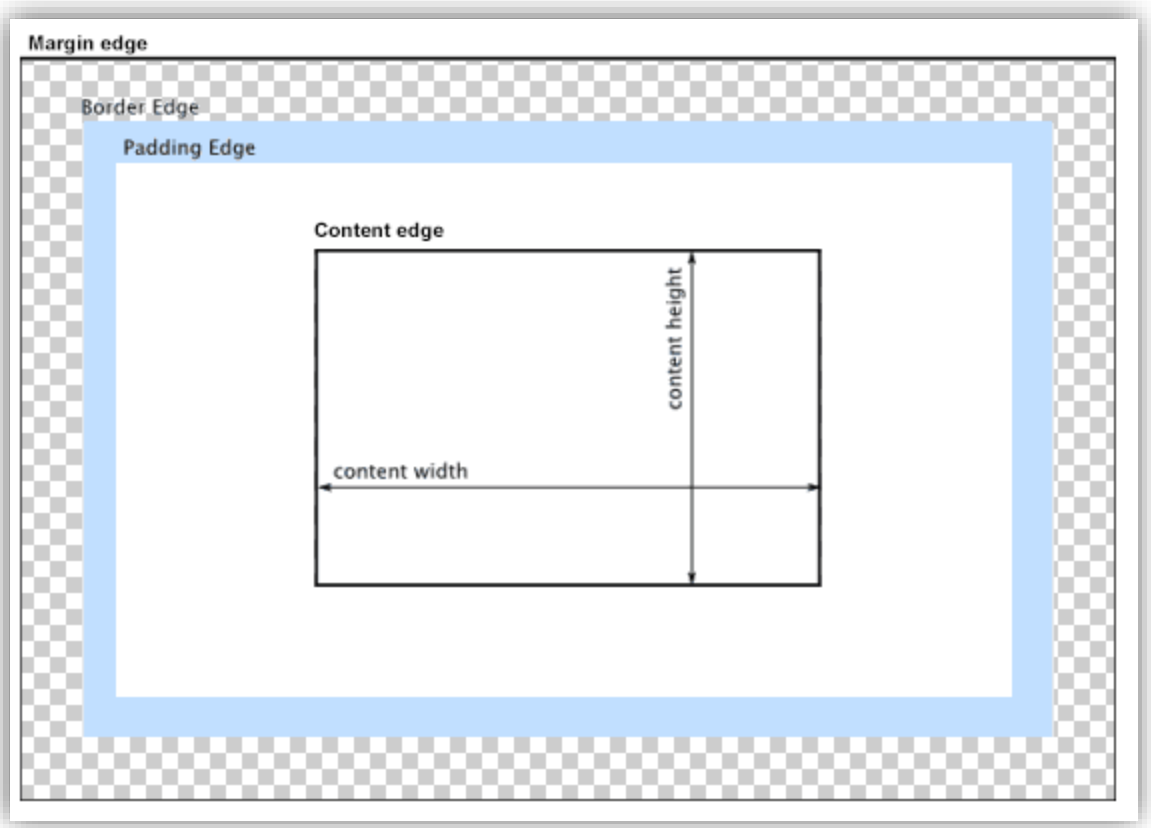
- To style lists, you can do the following

```
<style>
    ol {
        list-style-type: upper-alpha;
        list-style-image: url("lightining.png"); /* here we are using an image instead of the default */
        font
    }
</style>
```

- Anchor links
  - Can take all of the usual styles as well as "text-decoration"
  - Anchor link states:
    - a:link --> normal, un-visited link
    - a:visited --> has been visited
    - a:hover --> activated by mouse
    - a:focus --> activated with the keyboard
    - a:active --> is being clicked

⇨ Note: don't make a <a> look like a button, use a <buton> instead. Be semantically correct.

- Padding == inside space
- Margin == outside space
- CSS box model



- The ENTIRE width / height of an element includes the following:
  - content size
  - specified width / height
  - padding
  - margin
  - border
- If you use "`box-sizing: border-box;`" then The width and height properties (and min/max properties) includes content, padding and border, but not the margin.
- If you use "auto", content will be centered. However,
  - The element must be a block
  - The element must not float
  - The element must not have a fixed or an absolute position
- Measurements can be:
  - Absolute, such as: px, mm, cm , pt
  - Fluid, such as: %, vw, vh

- Browsers are different in how they display elements, not all browsers support all CSS3 properties. The easiest way to eliminate browser differences is by using a default / reset style sheet.

⇨ Note: remember, you should put your stylesheet after the default stylesheet.

- Browser prefixes
    - Provide a quick fix for handling unsupported CSS3 properties.
    - The following are the browser prefixes
        - -webkit-
        - -moz-
        - -ms-
        - -o-
    - Sites like (http://caniuse.com/) will tell you when you need to use prefixes

- Advice:
    - Always design the layout first, then code
    - Use borders to help you debug
- Your site should follow the POUR guidelines:
    - Perceivable
    - Operable
    - Understandable
    - Robust

Week 3

- A CSS pseudo-class is a keyword added to selectors that specifies a special state of the element to be selected. For example :hover will apply a style when the user hovers over the element specified by the selector.
- Just like pseudo-classes, pseudo-elements are added to selectors but instead of describing a special state, they allow you to style certain parts of a document. For example, the ::first-line pseudo-element targets only the first line of an element specified by the selector.
- Types of pseudo-classes:
    - link (e.g., visited, ...)
    - user action (e.g., click, hover, ...)
    - forms (interfaces)
    - structural / positional (e.g., first-child, nth-child, ...)
    - textual (e.g., first-letter, first-line, ...)
    - positional / generated (e.g., before, after, ...)
    - fragments (e.g., selection, ...)

- Transitions: CSS Transitions allows property changes in CSS values to occur smoothly over a specified duration

⇨ A transition applies a property (background-color, width, height, etc.) over time.

- The properties that you can control:
    - transition-property --> what is it that you want to change? (size, color, position, ...)
    - transition-duration --> how long should each transition last?
    - transition-timing --> should it be a smooth transition (linear)? or different?
    - transition-delay --> how long should the wait be before the transition begins?
- To set up a transition
    - define your element
    - choose the elements for transition
    - define the new values (you must combine this step with a pseudo-class)
- Example

```
<style>
    div {
        color: #000000;
        background: #2db34a;
        line-height: 200px;
        text-align: center;
        width: 250px;
        height: 200px;
        border-radius: 6px;
        transition-property: color, width, background, border-radius;
        transition-duration: 0.5s;
        transition-timing-function: linear;
        transition-delay: 0.5s;
    }

    div:hover {
        color: #ffffff;
        width: 350px;
        background: #2d31b3;
        border-radius: 50%;
    }
</style>
```

- transforms: CSS transforms allows elements styled with CSS to be transformed in two-dimensional or three-dimensional space

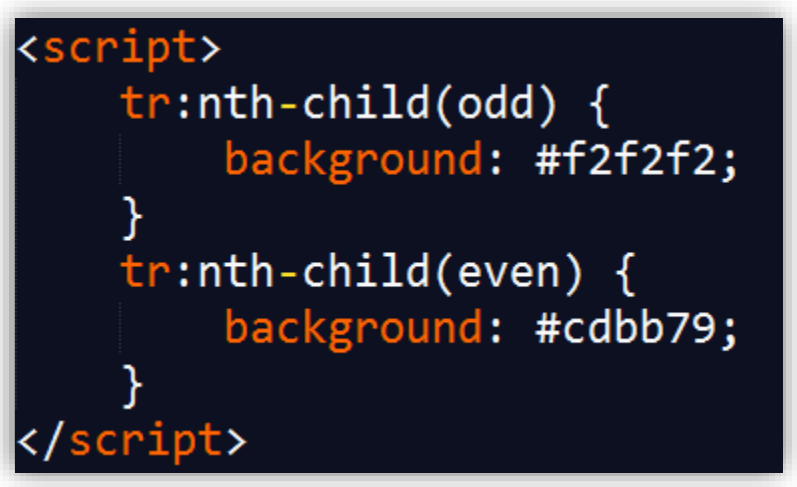
⇨ A transformation rotates/scales/skews the element over the X,Y, or Z axis's.

- 2D transform options:
  - translate
    - translate(x, y)
    - move x pixels to the left / right and y pixels up / down
  - rotate
    - rotate(deg)
    - rotate / spin the element a certain number of degrees
  - scale
    - scale(width, height)
    - change the width and height of the element
  - skew
    - skew(x-angle, y-angle)
    - rotate the element a certain number of degrees along the x and y axis
  - matrix
    - combines all of the 2D transform methods into one
- 3D rotate:
  - rotate3d(x, y, z)

- position properties:
  - static
    - default value for elements
    - not affected by top, bottom, left, and right
  - relative
    - positioned relative to itself
    - this preserves the original position of the element, and other elements are not allowed to move into this space
  - absolute
    - element is removed from the document flow and positioned relative to its nearest ancestor
    - the original space and position of the absolutely positioned element will not be preserved (i.e., other elements behave as if the element does not exist)
  - fixed
    - positioned relative to the browser window (will not move even if the window is scrolled)
- position is modified by the following properties:
  - top
  - right
  - bottom
  - left
  - z-index --> dictates stacking / layering order

Week 4

- To alternate colors of table rows between even and odd, use the following:



```
<script>
    tr:nth-child(odd) {
        background: #f2f2f2;
    }
    tr:nth-child(even) {
        background: #cdbb79;
    }
</script>
```

- Example of a calendar: http://codepen.io/ColleenEMc/pen/pjdpVW
- Example of a vertical navigation: http://codepen.io/ColleenEMc/pen/qOVpYg?editors=0100
- Example of a horizontal navigation: http://codepen.io/ColleenEMc/pen/dYZJeE
- General guidelines
    - use headings properly
    - give meaningful description for links
        - don't use 'here', 'click here'
        - don't use URL as link descriptor (unless very short and intuitive)