CMSC335

Web Application Development with JavaScript

CSSII

Department of Computer Science

University of MD, College Park

Slides material developed by Ilchul Yoon, Nelson Padua-Perez

CSS Templates

- CSS (theme) templates
  - Primarily for static HTML sites
  - Goes beyond CSS. A template distribution includes
    - HTML files
    - CSS files
    - Images, fonts, icons
- https://www.w3schools.com/w3css/w3css_templates.asp
- https://www.free-css.com/
- https://styleshout.com/free-templates/
- https://html5up.net/
- https://freebiesbug.com/

Custom Properties (variables)

- We can define CSS variables
  - To define a variable, use two dashes followed by the name (use dashes to separate words of a name with multiple words)
  - Use --ANAME: property value
    - » E.g., --my-favorite-color: red;
- To refer to the variable's value use var
  - E.g., var(--my-favorite-color);
- **calc()** – function for expression evaluation
  - font-size: calc(var(--my-size) * 2); /* no units specified, implied */
  - You may need to specify units
    - » font-size: calc(var(--my-size) + 2em);
- **Example:** CustomProperties.html
  - :root { } can be replaced with :html { }

Importing CSS into CSS

- **We can use @import to import a CSS file**
  - Use url() to specify the file e.g., @import url(ImportingThree.css);
- **@import** must be defined at the top of the stylesheet before any other rule (except @charset and @layer) and style declarations (otherwise it will be ignored)
- **Example:** Import.html

Using Google Fonts

- Google supports a set of nice fonts anyone can link in HTML docs
- https://fonts.google.com/
- How to use
  - Click on a font of interest
  - Select the “Get font” button at the top right (this will add it to your bag)
  - You can add several fonts to the bag (top right)
  - If you click on the bag icon you will see a “Get embed code” and “Download all” options
    - » The “Get embed code” provides <link> and @import options
- Use the fonts with the “font-family” CSS property
- **Example:** GoogleFont.html

Descendant and Child Selectors

- **Descendant selector**
  - Override the type, class, and id selector styles
  - Typically with two elements where the second is a descendant
  - **Example: #header h2** {font-weight: normal;}
  - **Example:** DescendantSelector.html
- **Child selector**
  - A child selector matches when an element is the child of some element. A child selector is made up of two or more selectors separated by "**>**"
  - **Example:** body > p { line-height: 1.3 }
    - » Sets the style of all p elements that are children of body
  - **Examples:** ChildSelector.html

Attribute Selectors

- Match elements with certain attributes defined in the source document
- **Syntax**
  - **[att]** Matches when the element sets the "att" attribute, whatever the value of the attribute
  - **[att="val"]** Matches when the element's "att" attribute value is exactly "val"
- **Examples:**
  - **h1[title] { color: blue; }** → Matches all **h1** elements that specify the "title" attribute, whatever its value
  - **span[class="example"] { color: blue; }** →Matches all **span** elements whose "class" attribute has exactly the value "example“
  - **input[type = "submit"] { color:blue; }** → An input element with a type attribute that has the value submit
- **Example:** AttributeSelector.html

Pseudo-element and Universal Selectors

- **Pseudo-element**
  - Keyword added to a selector lets you style a specific element part. Pseudo-element employs a double-colon (:☺ so it can be distinguished from pseudo-class (in the past, a single colon was used)
  - **Examples:** ::**first-letter**, ::**first-line**
- **Universal selector**
  - Applies to all elements in context
  - **Example: *** {font-family: arial, Helvetica; }
- **Example:** PseudoElementsUniv.html, PseudoElementsUniv.css

Display Property

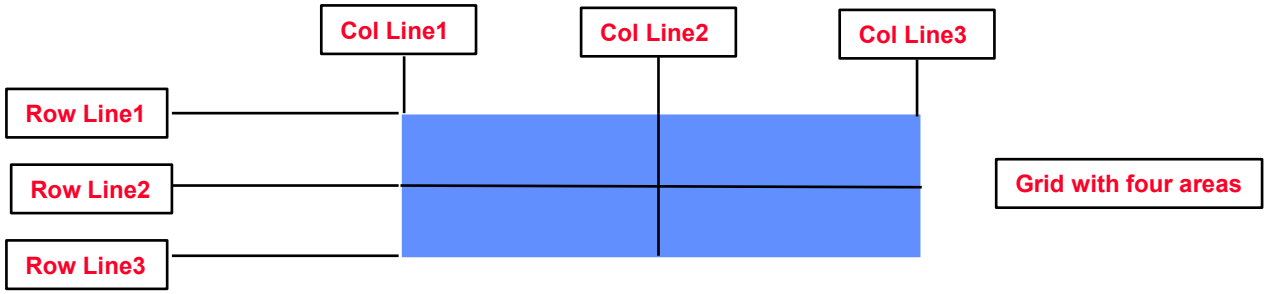
- display property
  - Defines the type of rendering box (e.g., block, inline) of an element
  - **Values**
    - » **inline** - causes a block-level element to act like an inline one
    - » **block** - causes an inline element to act like a block-level one
    - » **inline-block** - causes a block-level element to flow like an inline one while retaining other features of a block-level element
    - » **none** - hides an element from the page
    - » **flex** - displays element as a block-level flex container
    - » **grid** - displays element as a block-level grid container
  - **Reference:** https://www.w3schools.com/cssref/pr_class_display.php
- **Example:** InlineBlockNoneDisplayProperty.html

flex Display Property

- **flex** display property - sets a container (e.g., div) to be a flexbox (flexible box layout) element. Elements in the container are organized in a row or column
- Terminology
  - **main axis** - defined by the flex-direction property
  - **cross axis** - perpendicular to main axis
- **flex-direction** property values → row, row-reverse, column, column-reverse
- **Example**: FlexDisplayProperty.html

grid Display Property

- **grid** display property - sets a container (e.g., div) to use a grid
  - grid - a collection of horizontal and vertical lines
    - » Horizontal lines are called **rows**
    - » Vertical lines are called **columns**
    - » Space between rows and columns is called **gap**



grid Display Property

- Using the grid property, we can create layouts easily (in the past, using float and positioning)
- Unlike the **flex** property, adding the **grid** property will not make the elements look any different, as you will only get a one-column grid
- Several alternatives to specify the rows and columns properties
  - Positioning with **grid-template areas**
  - **Line-based** placement
- **Example**: Grid.html
  - **grid-template-columns** property - defines the number and width of columns
  - **grid-template-columns** property - defines the height of each row
  - Use case - defining calculator
- **Example**: GridTemplateAreas.html

float Property

- **CSS normal document flow/normal position** - placing of elements one after another based on the document structure and whether the element is an inline or block element
- **float**
  - Places an element on the left or right side of the container, enabling text and inline elements to wrap around it
  - **Values**
    - » none, left, right
- **Example:** FloatI.html
- **Example:** FloatII.html
  - Creating a layout using floats