Reminders

- Attendance is 15% of grade
- Discuss Registration
- Quiz 3 Due Thursday
- Lab 1 Due next Tuesday
- T.A. Hours
  - Monday and Wednesday 4-5 pm
- Zoom Classes Week 6, September 26 and 28th
- Job Fair Q & A

ITMD 441/541
Web Application Foundations
Week 4

FALL 2023 – SEPTEMBER 11, 2023

This week’s Agenda

- CSS3 Techniques
  - CSS Layout
  - Responsive Design
  - CSS Transitions, Transforms, Animations
- Assignments

CSS3 Techniques

CSS Layout

CSS Layout

- In the beginning there was very little control over layout
- Most pages content was just displayed in the order of the HTML
- HTML only had some attributes that applied styling
- For more complex layouts nested table elements were used
- With the creation of CSS, the separation of HTML and style was possible
- The CSS Zen Garden project showed the power of CSS

CSS Layout

- CSS Page Layout Techniques
  - Normal Flow
  - Display Property
  - Flexbox
  - Grid
  - Floats
  - Positioning
  - Table Layout
  - Multiple-column layout
  - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Introduction

CSS Layout

- Normal Document Flow
  - HTML is displayed in exact order of the document
  - Block direction runs according to language
- Display Property
  - Main property to do page layout in CSS
  - Changes way an element displays
  - Elements have a default value, mostly block or inline
  - Can be used to change the default presentation like have a block act like an inline or an inline act as block
  - Two main modern display layout values are flexbox and grid

CSS Flexbox

- Flexible Box Layout CSS module
- Designed for **one-dimensional layout**, single axis, rows or columns
- Set `display: flex` on the parent container and all the direct children elements become flex items
- By default, it will layout the flex items horizontally next to each other and the same height with no wrapping
- Other properties to adjust this behavior
- There are also properties to control how how the items grow or shrink and the order they will display
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox
- https://css-tricks.com/snippets/css/a-guide-to-flexbox/

CSS Grid

- CSS Grid Layout Module
- Designed for **two-dimensional layout** , multi-axis, rows and columns
- Set `display: grid` on the parent container and all the direct children elements become grid items
- By default, it doesn't look any different, unlike flexbox, you get a one column grid
- Row and column tracks on the parent are created with the `grid-template-rows` and `grid-template-columns` properties
- They can be defined with normal units or with the fractional unit `fr`
- Gaps between tracks created with `column-gap` and `row-gap` properties
- Gaps need to be length or percentage, not `fr`

CSS Grid

- There is a repeat function that will allow you to define repetitive tracks in your grid-template rows or columns
- Tracks you create are the explicit grid and there is also implicit grid
- Implicit tracks are created with auto size by default, but they can also have default sizes with `grid-auto-rows` or `grid-auto-columns`
- There is s minmax function that you can use to set the min and max size of a track
- Grid item placement can be controlled and allow for placement at any grid tracks with the `grid-column` and `grid-row` properties
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Grids
- https://learncssgrid.com/
- https://css-tricks.com/snippets/css/complete-guide-grid/

CSS Floats

- Removes an element from the normal document flow and moves it to the left or right
- Causes the following elements to float or wrap around it
- The `clear` property is used to stop a following element from floating or wrapping around the element
- Originally created to float an image in some text but was used to float anything
- These techniques were used to create multi-column layouts in days before flexbox or grid
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Floats
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Legacy_Layout_Methods

CSS Positioning

- We talked about this last week
- Uses the `position` property with one of the following values
- Static – default normal flow
- Relative – puts the element in normal flow but allows you to move it after
- Absolute – takes element out of the flow and positions it to the first parent that has a positioning context
- Fixed – positions element based on browser viewport
- Sticky – newer combination of relative and fixed
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Positioning

CSS Table Layout

- CSS properties that define the way tables lay out in a page
- These properties can be applied to other elements that are not tables to get them to lay out like tables
- Now this should be considered a legacy method and not used unless you can not use flexbox or grid

CSS Multi-column

- Provides a way to lay out content in columns like a newspaper
- Not as useful for web content because you would have to scroll up and down
- Can be useful for things like long lists
- Make a block element into multiple columns with either the `column-count` or `column-width` properties
- There is also a `column-gap` property to add space between columns
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Multiple-column_Layout

Responsive Design

Responsive Design

- Basic HTML is responsive by default
- Text reflows depending on the screen size
- For example, a paragraph element (block element) will take the full width of its container and the text will reflow to fit.
- Where this becomes an issue is when you are tying to go beyond basic HTML flow and do a stylized layout
- Historically pages would use either a fluid or fixed layout
  - Fluid layout allows elements to fill their container or have set widths based on percentages
  - Fixed layouts would have elements set to specific pixel dimensions
    - Often pages would be design to look good on a specific screen size
    - For a long time that was a 960px width because it fit on 1024x768 monitors

Responsive Design

- The next era of layouts came as phones started being able to show web pages
- Basic feature phones where very limited so people started designing separate mobile websites usually on a different domain
- As phones got more powerful and could display a full featured web page (starting really with iPhone) this became limiting
- Along came the concept of responsive design in 2010
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Responsive_Design

Responsive Design

- Was made primarily of three techniques together
  - Fluid Grids/layouts
  - Fluid Images
    - Setting the `max-width` property to 100%
  - Media Queries
    - Ability to load different CSS based on media rules like screen size
    - Media Queries Level 3 specification in 2009
- Need to add the viewport meta tag to tell mobile browser to not scale the pages to fix 980px and just make the viewport the device width

Media Queries

- They can be used to target rulesets to specific screen sizes, system settings, or media types like print
- Two ways to use them
  - `@media` rule in a stylesheet that surrounds the rulesets that you want to apply to that media condition
    - `@media type and (feature)`
  - Using the media attribute on a link tag to only load a stylesheet if that media condition applies
    - `<link rel="stylesheet" href="specific.css" media="type and (feature)">`
- https://css-tricks.com/a-complete-guide-to-css-media-queries/

Responsive Images

- Basic idea here is images will never be bigger than their container and are allowed to scale down
- At a most basic level you set images max width with one of the following properties
  - `max-width: 100%;`
  - `max-inline-size: 100%;` and `block-size: auto;`
- You can deliver different size images based on media conditions with images and the `srcset` and `sizes` attributes
- You can do art direction and serve different images altogether with the `picture` element and `source` elements
- https://developer.mozilla.org/en-US/docs/Learn/HTML/Multimedia_and_embedding/Responsive_images

CSS Transitions, Transforms, Animations

CSS Transitions

- Transitions are a way to control the speed of a CSS property change
- You can set things like duration and delay of property changes
- Uses the `transition` shorthand property or the individual properties of `transition-property`, `transition-duration`, `transition-timing-function`, `transition-delay`
- The transitions will apply when an elements CSS state changes, typically things like hover or a class is added or removed for example
- Not all properties can be transitioned, only animatable ones
- CSS Transforms are common since they are hardware excellerated
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions/Using_CSS_transitions

CSS Transforms

- Transforms are used to change the shape and position of elements
- Transforms do not affect the normal document flow
- Transforms include rotation, skewing, scaling, and translation
- Can be in 3D space in addition to the normal 2D plane
- Only elements with `display: block` can be transformed
- Two main transform properties
- `transform`
  - Space separated list of transforms to apply
- `transform-origin`
  - Position of the origin for the transform, by default it is the center of the element

CSS Transforms

- The transforms are a series of functions
  - Scale
    - Scales an element along one or more of its axis's
  - rotate
    - Rotates an element around its origin
  - Translate
    - Moves an element along one or more of its axis’s without affecting the document flow
  - Skew
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transforms/Using_CSS_transforms

CSS Animations

- CSS Animations allow you to animate changes to CSS styles from one set of rules to another
- Can be more complex than a simple transition, instead of going from one point to another there can be in-between keyframes
- Same animatable properties as transitions
- Transitions trigger on simple state changes, there is much more control over duration, iteration, playback control with animations
- There is a shorthand property `animation`, but you can use all the individual properties instead
- The `@keyframes` at rule defines the steps of the animation sequence
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations

Assignments

Reading/Assignments

- Quiz 3 – Week 3 Content – Due end of day September 14
- Lab 1 – Due Tuesday, September 19
- Reading:
  - MDN Into to CSS Layout:
    - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Introduction
  - Learn CSS – 8, 9, 10, 25, 19 (transform section), 21 - at minimum
    - https://web.dev/learn/css/
  - Learn Responsive Design – 0, 1, 2, 4, 5, 7, 8, 14 - at minimum
    - https://web.dev/learn/design/
- Quiz 4 – Week 4 Content – Will be assigned this weekend no later than Saturday morning.