Web Development Course

What really happens when a website is rendered on browser

DNS Server

Frontend

Backend

Database

HTML & CSS

- HTML – Content

- CSS – Appearance of that Content

Common HTML Terms

- Elements
- Tags
- Attributes

HTML Document Structure

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is a web page.</p>
  </body>
</html>
```
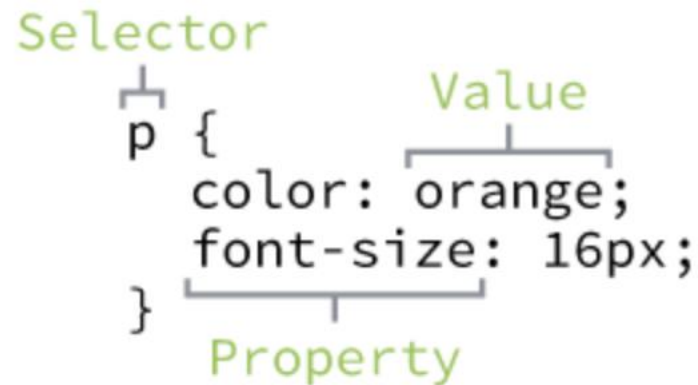
Required HTML Elements

All HTML elements have to follow a required structure, that includes the following Elements:

- <!DOCTYPE html> - informs the version of html being used
- <html> - signifies beginning of the document
- <head> - includes metadata, content is not displayed
- <body> - all visible content fall within this elment

Common CSS Terms

- Selectors
- Properties
- Values

```
Selector
  ┌┐            Value
  p {       ┌─────────┐
    color: orange;
    font-size: 16px;
  }     └──────────┘
          Property
```

CSS Selectors

- Type Selectors

```
div { ... }
```

- Class Selectors

```
.awesome { ... }
```

- Id Selectors
- Advanced Selectors

Referencing CSS

```html
<head>
  <link rel="stylesheet" href="main.css">
</head>
```

Using CSS Resets

https://meyerweb.com/eric/tools/css/reset/

Blocks Vs Inline Elements

- Block-level elements begin on a new line, stacking one on top of the other, and occupy any available width. Block-level elements may be nested inside one another and may wrap inline-level elements. We'll most commonly see block-level elements used for larger pieces of content, such as paragraphs.

- Inline-level elements do not begin on a new line. They fall into the normal flow of a document, lining up one after the other, and only maintain the width of their content. Inline-level elements may be nested inside one another; however, they cannot wrap block-level elements. We'll usually see inline-level elements with smaller pieces of content, such as a few words.

Most Crucial HTML Elements

- Divs
- Spans
- Headings
- Paragraphs
- Bold Text
- Italicize Text

Structure of Webpage

<header>

<section>

<article>

<aside>

<footer>

HTML Elements Cont…

- Hyperlinks
- Relative and Absolute Paths
- Linking to an Email Address
- Opening Link in a New Window
- Linking to Parts of the same page

CSS Cascading

- Specificity
- Combining Selectors
- Specificity Within Combined Selectors

Common CSS Property Values

- Colors
  - Keyword Colors
  - Hexadecimal colors
  - RGB and RGBa colors
- Lengths
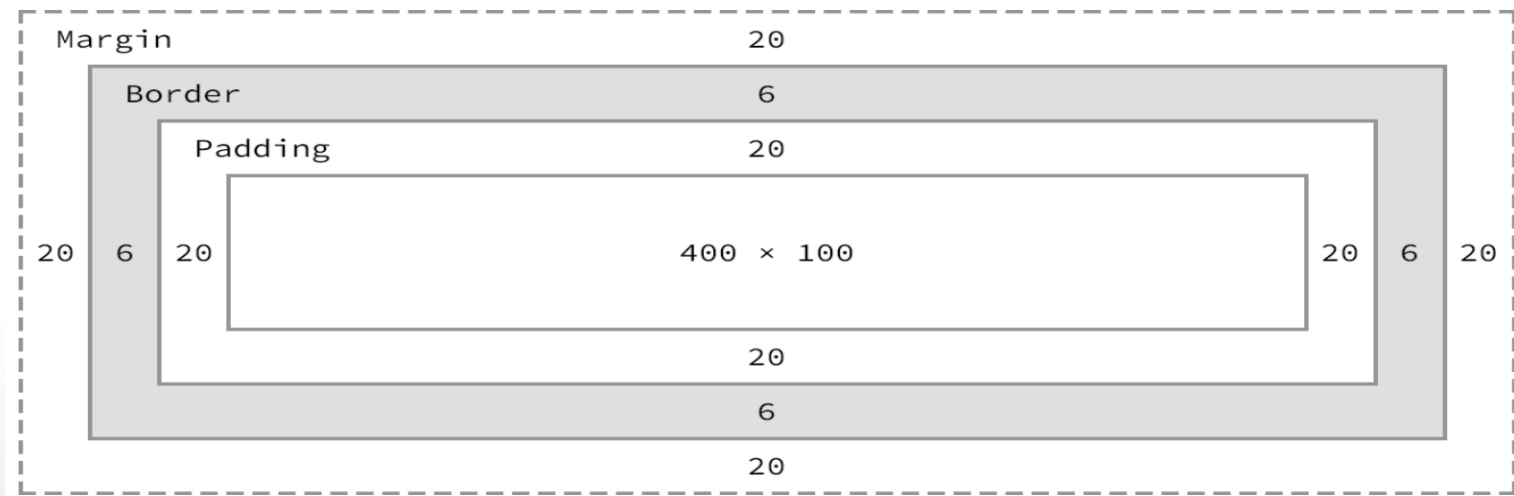 - Absolute lengths
  - Relative lengths

Display Property

- Block
- Inline
- Inline-block
- None

The Box Model

- Every element on a page is a rectangular box.
- How do you calculate the height and width of this rectangular box.
- Width, height, padding, border, margin
- Total width : margin-right + border-right + padding-right + width + padding-left + border-left + margin-left

Width & Height

- Inline level elements will not honor height and width properties. They will always take the height and width which is required for the content within them to be visible.

- Block level and inline-block level elements will honor the height and width properties.

Margin

- Margin for an element always falls outside of the border.
- Inline level elements don't honor vertical margins i.e. top and bottom margins. Whereas that is not the case with block level elements or inline-block level elements.
- Color : Transparent property, background color of the parent element.

Padding

- Padding it falls inside of the element's border.
- Vertical and horizontal paddings both gets applied even for inline level elements. Unlike margin property.
- Color : transparent property, background color of the element to which it is applied to.

Border

- Width, style and color.
- Border-radius
- Box-sizing – content-box, padding-box, border-box

Positioning Content

Vertical Positioning

Positioning With Float

- Width of element defaults to the width of content within it. Fix it by adding a fixed width property.

- Floated elements often touch each other. Fix it by using the margin property.

- Display value may change to block for non block level elements.

Positioning Content


```
<header>

<section>          <aside>
float: left;       float: right;

<footer>
```

Floats for Positioning

Problems with using Floats

<header>

<section>
float: left;

<footer>

<aside>
float: right;

Element after the cleared elements may acquire the remaining space called "gutter"

Clearing and Containing Floats

- Proper styles are not rendered properly on elements around floated elements. These can be either elements sitting next to the floated element or parent element of a floated element.

- Unwanted elements consume any space available around floated elements.

- Apply clear property to elements appearing after the floated element, not before, to return the page to its normal flow.

- Contain float by applying clearfix styling to parent element of the floated element.

Clearfix Code

```css
.group:before,
.group:after {
  content: "";
  display: table;
}

.group:after {
  clear: both;
}

.group {
  clear: both;
  *zoom: 1;
}
```

Clearfix for containing Floats

<header>

<section="group">

<section>
float: left;

<aside>
float: right;

<footer>
clear: both;

Positioning with Inline-Block

`<header>`

`<section>` display: inline-block;

`<section>` display: inline-block;

`<section>` display: inline-block;

`<footer>`

Remember to take care of whitespaces between inline block elements

Tip

Use Inline-Block to create reusable grid layout for the page.

Use floats to wrap content around an element

Typography – Font based Props

- Color
- Font-family
- Font-size
- Font-style
- Font-weight
- Line-height
- Font

Typography – Text based Props

- Text-align
- Text-transform
- Letter-spacing
- Text-Decoration

Background

- Background-color
- Background-image – background-repeat
- Background-position – by default background images are positioned at left top corner of an element.
- Background

Forms

- Input element

```
<input type="date" name="birthday">
<input type="time" name="game-time">
<input type="email" name="email-address">
<input type="url" name="website">
<input type="number" name="cost">
<input type="tel" name="phone-number">
```

- Textarea

- Radio Buttons

```html
<input type="radio" name="day" value="Friday" checked> Friday
<input type="radio" name="day" value="Saturday"> Saturday
<input type="radio" name="day" value="Sunday"> Sunday
```

- Checkboxes
- Drop-down lists

```html
<select name="day" multiple>
  <option value="Friday" selected>Friday</option>
  <option value="Saturday">Saturday</option>
  <option value="Sunday">Sunday</option>
</select>
```

Forms continued…

- Submit Input
- Submit Button
- Label
- Fieldset
- Placeholder
- Required