

HTML & CSS Notes

HTML Semantics

`<header>`

`<footer>`

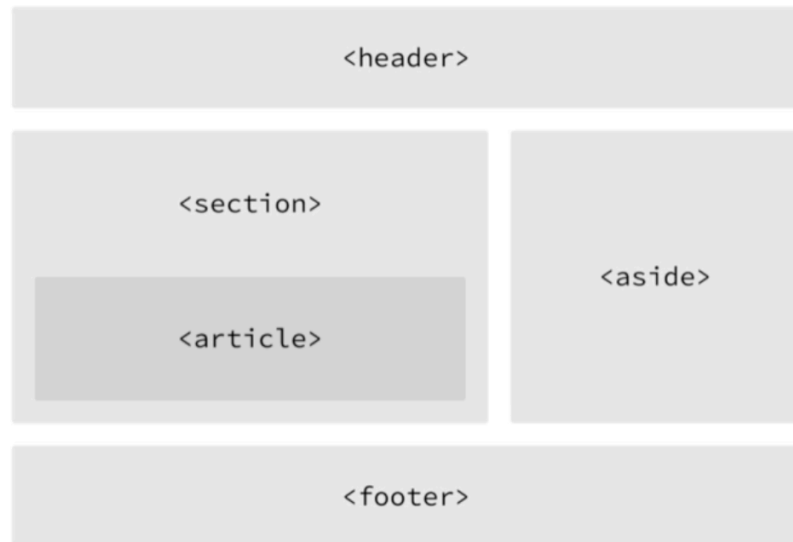
`<nav>`

`<main>`

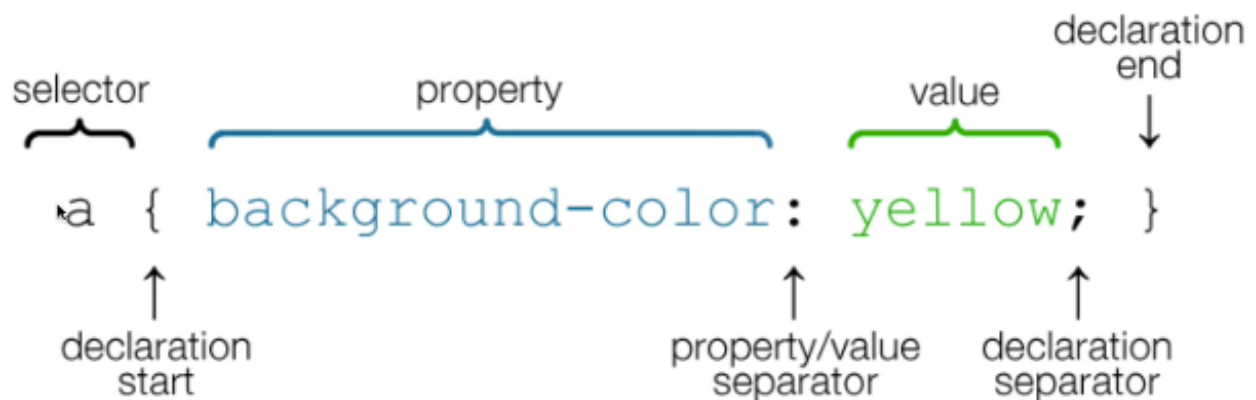
`<section>`

`<article>`

`<aside>`



CSS Structure

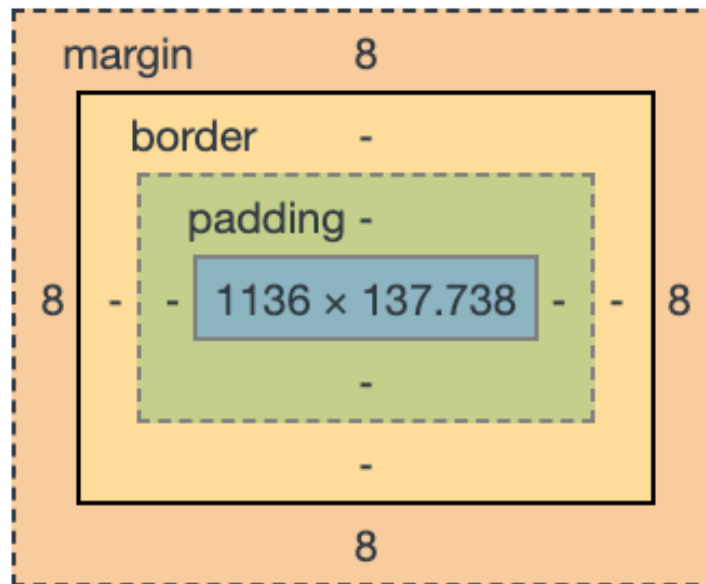


CSS Units: Absolute

cm	Centimeters
mm	Millimeters
in	Inches
<u>px</u>	Pixels (1px = 1/96th of 1in)
pt	Points (1pt = 1/72 of 1in)
pc	Picas picas (1pc = 12 pt)

CSS Units: Relative

%	To parent element
em	To font-size of parent element
<u>rem</u>	To font size of root element
vw	To 1% of viewport width
vh	To 1% of viewport height

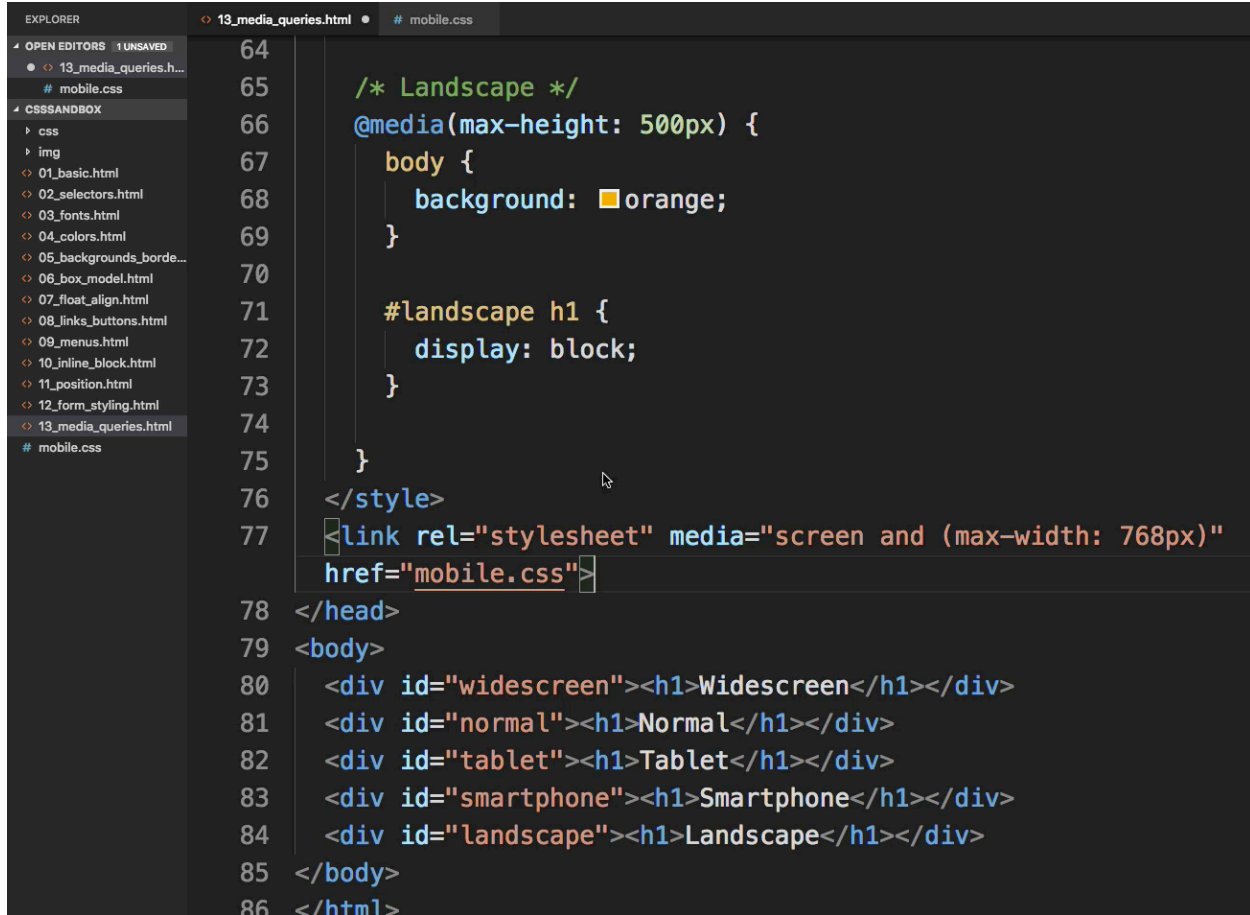


Position Values

Static	Not effected by tblr(top, bottom, left, right) properties/values
Relative	tblr values cause element to be moved from its normal position
Absolute	Positioned relative to its parent element that is positioned "relative"
Fixed	Positioned relative to the viewport
Sticky	Positioned based on scroll position

Notes from Hotel Project

1. Look for ideas into websites that has similar interests or features that you want to create. <https://themes.getbootstrap.com>
2. Free images <https://www.pexels.com/es-es/>
3. Create page structure:
 - Create a folder with: one folder for images "img" and other folder for "css"
 - Create files index.html | about.html | contact.html | style.css
4. Index.html
 - Create template with emmet !
 - Put title using keywords for search engines
 - Add meta tags for search engines one for description and another for keywords
 - Use link tag for css file
5. Start working with <body>
6. Insert the logo and use for link index.html to itself
7. Create navbar with and
8. Use .classes and #ids in your tags
9. Go to style.css file to start working with style. Working by component, that means create html structure and style it, before jumping into the next component.
10. Continue with showcase area
11. Procedure with Home info section.
12. Continue with features and footer
13. Use font awesome link to add icons.
14. About page
 - Copy index.html content and paste in about.html
 - Clear things that you are not going to use, leave navbar and footer
15. Contact page
16. Copy About page and add submit form
17. Make it responsive
18. Create new file mobile.css and add link tag to index.html, About and contact.
19. Add just the necessary changes to look good in small devices.



The screenshot shows a code editor with two files: 13_media_queries.html and # mobile.css. The HTML file contains a head section with a link to the CSS file and a body section with five divs representing different screen sizes: widescreen, normal, tablet, smartphone, and landscape. The CSS file contains a media query for landscape screens with a maximum height of 500px, setting the background to orange and the display of the #landscape h1 to block.

```
64
65 /* Landscape */
66 @media(max-height: 500px) {
67     body {
68         background: orange;
69     }
70
71     #landscape h1 {
72         display: block;
73     }
74 }
75
76 </style>
77 <link rel="stylesheet" media="screen and (max-width: 768px)"
78 href="mobile.css">
79
80 </head>
81 <body>
82     <div id="widescreen"><h1>Widescreen</h1></div>
83     <div id="normal"><h1>Normal</h1></div>
84     <div id="tablet"><h1>Tablet</h1></div>
85     <div id="smartphone"><h1>Smartphone</h1></div>
86     <div id="landscape"><h1>Landscape</h1></div>
87 </body>
88 </html>
```

Units

Its better to use Rem units, makes it more accessible

Another huge reason to use rem units its for accessibility and browser settings apply changes to rem units not to em.

Pixels area a fixed units

Vh - Viewport height

Vw - Viewport width

Values between 0 and 100

Useful to create **landing pages**

Because It doesn't matter how tall or short the browser is, as we scroll adjust content.

Flexbox

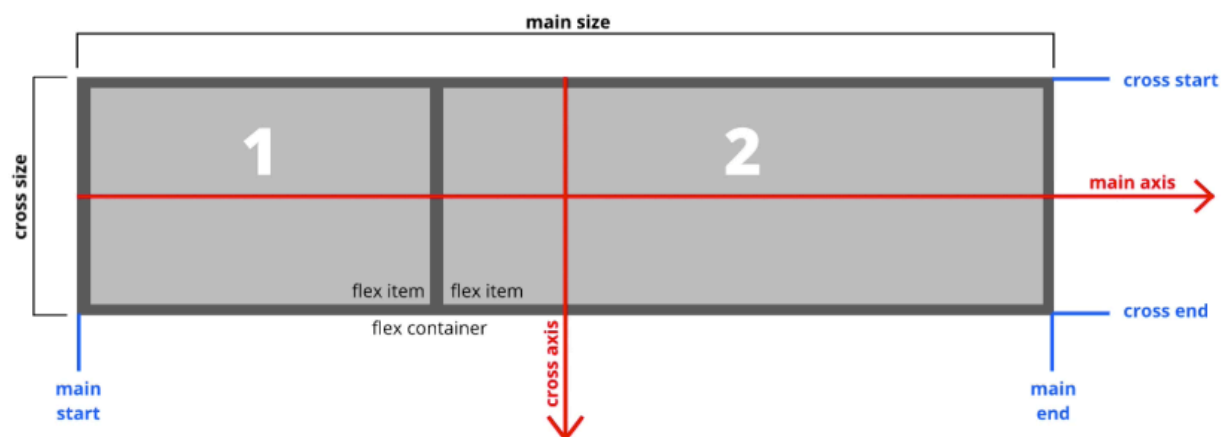
Row - Direction- X - Main axis Y - Cross axis (default)

Column - Direction Y - Main axis - X -Cross axis

Flex Properties

► **display: flex;** */ Creates a “flex container” */

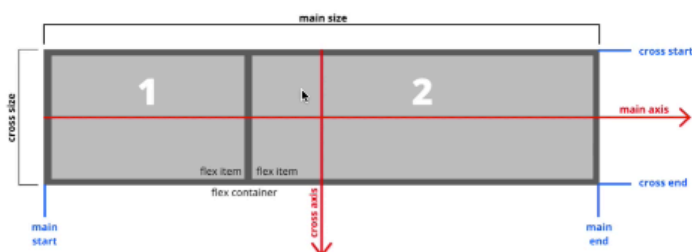
► All direct child elements are “flex items”



► justify-content: Align along the main axis (horizontal)

► align-items: Align items along the cross axis (Vertical)

► align-content: Align when extra space in cross axis



GRID

Similar to flex box, more powerful and a bit more difficult

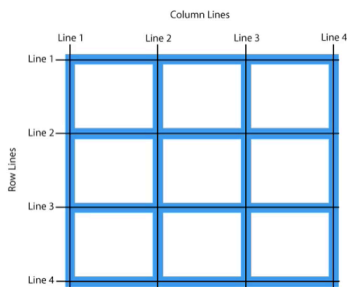
Two dimensional layouts

New unite “fr” fraction

► **display: grid;** */ Creates a “grid” */

► All direct child elements are “grid items”

► “grid-template-columns” defines width and number of cols



It is not one or the other

Grid for outer elements and grid-like layouts like boxes

For simple alignments or simple elements (Inner elements, menu items, etc) use flex

```
grid-row: span 8;
```

```
grid-column: 2 / 4;
```

```
height: 100px;
```

```
grid-row: span 9;
```

```
grid-row: span 9;
```

SASS

Sintactically Awesome StyleSheets

CSS Preprocessor / Precompiler

Enhances the functionality of CSS

Other preprocessors include Less and Stylus

How Does SASS Work?

Uses **.scss** or .sass file extensions

The browser does NOT read Sass, it must be compiled

Sass files are compiled to normal CSS files

There are many different types of Sass compilers (cli & gui)

Why is it worth using it ?

Variables

Nesting

Partials / Imports

Functions & Mixins

Conditionals

Inheritance

Operators & Calculators

Color Functions

.scss is usually preferred over **.sass** as it uses the same syntax as regular css

SASS	SCSS	CSS
<pre>\$color: red \$color2: lime a color: \$color &:hover color: \$color2</pre>	<pre>\$color: #f00; \$color2: #0f0; a { color: \$color; &:hover { color: \$color2; } }</pre>	<pre>a { color: red; } a:hover { color: lime; }</pre>

Web Hosting Basics

Shared Hosting

One account of many on a server (same operating system environment)

Cheapest option around \$3 to \$15 per month.

Used for small websites, low traffic.

Examples: Inmotion Hosting, Hostgator, Bluehost

VPS Hosting (Virtual Private Server)

Physical machine but virtualized environment (server)

Less people on the same server

Can create multiple shared accounts

More access and privileges

Everything shared hosting offers

\$20 to \$100 per month

Used to create multiple websites with separate control panels

Examples: Inmotion Hosting, Hostgator, Bluehost

20. Dedicated Server

Physical machine sitting in a data center you rent out.

Full access and privileges

Harder to manage but very powerful

\$100 to \$400 per month

For large size apps that are going to have a lot of users

Examples: Inmotion Hosting, Hostgator, Bluehost

21. Reseller Hosting

Very similar to VPS

Create and manage multiple shared accounts

Can sell shared accounts to your own customers

Usually comes with reseller software (WHCMS)

Examples: Inmotion Hosting, Hostgator,

Cloud Hosting

Most for web apps like e-commerce or social network.

Multiple servers work together

Extremely powerful, very scalable and great for large apps

Not for beginners/Harder to manage

Pay as you go

Examples: **Digital Ocean**, Linode, Vultr

You need to know linux terminal commands, engine x servers, firewalls, etc

Static Hosting

Does not come with bells and whistles of managed hosting

Upload static site via Git

Great for front-end apps and static websites, including sites that want to have backend services.

Free but pay for extra features

Examples: Netlify, Github Pages

Know at least the basics of Git (version control system).

What's Next

- ▶ Build your own projects based on what you have learned
- ▶ Learn JavaScript
- ▶ May want to learn Bootstrap or another CSS framework
- ▶ Code everyday
- ▶ Stay driven

✓ Learn a front-end framework

- React
- Angular
- Vue.js

✓ Learn Node.js or another server side language

- Express
- AdonisJS
- Loopback
- Swagger