html	provide additional information about our element, src and alt are attributes for img element
css	consists of a DOCTYPE declaration that specifies the version of HTML being used followed by a tree (hierarchy) of elements that define the structure and content of the document.
javascript	root element in an HTML document which always includes a head and a body element. head element: contains information about the page such as its title body element: includes the structure and content of the page.
URL	attribute that implements the grouping of elements to provide similar styling or behaviour
relative/absolute URL	attribute that implements a unique id for the element
DOM	identifies potential errors in code

attributes	hypertext-markup language, language that defines the structure for webpages
HTML document	cascading style sheets, styling language used to define styling for webpages
html element	language used to add functionality to webpages
class	uniform resource locator, the address of a given unique resource on the Web. there are relative urls and absolute urls
id	specifies the target resource relative to the current resource/ specifies the location of a resource irrespective of the current resource. It can start with a / to indicate the root of the website or a protocol to represent a resource on a different website.
validating HTML/CSS pages	The document object model is an API for valid HTML documents. It models the heirarchy of HTML objects in a tree data structure that can be traversed and manipulated to provide functionality for webpage

viewport	are only respected by block elements, span and inline-elements do not listen to these properties, you can change display: inline-block, to have an inline element that respects width and height properties, by default, height of block level elements is zero and width 100%
entities	we can provide different styles for different devices depending on features such as screen size, orientation, etc. By using media queries and relative measurement units we can build responsive web sites that adjust smoothly to various screen sizes.
hyperlink and link	contains multiple fonts as fallbacks in case of error with first font
semantic HTML elements	simulating a slow connection
block-level elements	permanent storage browsers use to store assets for faster performance. It ’ s essentially somewhere on the disk. The cache can always be cleared.
inline elements	the use of type in a design. It deals with fonts, size, color, layout, and other factors that affect the design of type on a page.

width and height properties

the viewable area of the webpage

media queries

special html characters denoted with prepended ampersand and end with a semicolon, ex of common html entity is copyright symbol, nonbreaking space entity, html characters

font stacks

often used interchangeably but hyperlink denotes element with link and link is actual url referenced

network throttling

html that implements structure which provides meaning and description of data to search engines, screen readers, and other software

caching

always start on new line and take up entire horizontal available space.

typography

don't start on new line and take up as much width as necassary.

<div></div> / <span></span>	navigation menus either in header or footer usually
<article></article>	there can only be one main
<section></section>	links, url can be relative:denotes path from current directory or absolute path:starts with / (root) or protocol,
<header></header>	since heading sizes can always be modified, only use headings according to hierarchy of your pages, only use one h1, h2s for main sections, h3 for subsections and so on
<footer></footer>	embedded stylesheets (style tag in head), external stylesheets, and inline styles
<aside></aside>	link tag for external stylesheets, rel attribute specifies what kind of resource we are linking to, href attribute is for absolute or relative url

<nav></nav>	block-level element / inline element, both are generic containers
<main></main>	an independent, self-contained content, ex forum post, comments, review
<a href=""></a>	sections are to group content
<h1></h1> ... <h6></h6>	headers are used for introductory content whether it be for page or article
3 WAYS TO PROVIDE CSS	footers used in same way as header but for closing content
<link rel="stylesheet" href="">	used for content that is not directly related to main content

NORMALIZING CSS	determine weight of selector: selectors are given more weight if they are more specific, unless !important keyword used because it has most weight, however try to avoid using if possible first id selector, then class and attribute selectors, then element selectors
each css rule contains...	<!-- some elements inherit styles from their parents --> <!-- if you don't wish to inherit style reset inherited property with value of initial keyword or if there is property that is not inherited property that you wish to inherit list the property and set value to inherit keyword -->
css trick for making circle image	the values of border are: thickness, style, color
Relational Selectors	use box-shadow property to apply shadows to elements, or text-shadow to apply shadows to text
Pseudoclass Selectors	When rendering an HTML document, the browser puts each element inside a box. The box contains four areas: the content area, the padding area, the border area and the margin area. We use use margins to add space between elements and padding to add space in between border and content.
Pseudo-element Selectors	whenever top and bottom margins are combined into a single margin. The size of the margin is equal to the largest of the two margins.

Selector specificities	providing common stylesheet that renders your html elements the same in different browsers since some of their settings differ
INHERITANCE	each css rule contains a selector (select elements by their type, class, ID, and attributes (which aren't that common)) and one or more declarations(property and value)
border: 10px solid royalblue	an image with border-radius half to width will have rounded edge
SHADOWS	provide cleaner markup than many classes and ids but are more fragile, take more time to lookup
Box model	used to style elements in a particular state that are supplied by browsers by default, denoted by a colon, :
margin collapsing	used to style part of an element that are supplied by browsers by default, denoted by a double colon, ::

overflow	positions element relative to its normal position, then set left, right, top, bottom property to add space
box-sizing property	positions element relative to its container, the container has to have position set to relative, then you can position this element absolute and set top, right, left, bottom property to set position in container, also removes this element from normal flow of page
absolute MEASUREMENT UNITS	positions element relative to viewport, set the position for element to fixed, then set top, left, bottom, right property for positioning
relative measurement units	<!-- should not use to build layouts, but if you are maintining old websites we may come into contact w floating elements for layouts --> <!-- by default, parent elements don't see floated child elements --> <!-- set float property to whatever side you want to push element to, rest of page will flow around this element and treat floated element as inline, set clear property for an element to treat the floated element as a block element--> <!-- float collapse happens when the parent collapses bc it does not have enough content to contain floated element --> <!-- you can clear the float by introducing a generic clearfix class below --> <style>.clearfix::after{content:'';display:block;clear:both;}</style>
use which: relative or absolute measurement units?	used for laying out elements in one direction, row or column, <!-- set parent container to flex, flex-direction to row or column --> <!-- two axis: main and cross -->
z-index: number	<!-- used to display content in both rows and columns --> <!-- popular use is photo galleries, or major page areas --> <!-- set container to display: grid, and grid-template-rows and grid-template-columns properties, grid template is shorthand property for the two-->

position: relative; left:4rem; bottom:4vh	happens when fixed size container cannot fit the entire content, use overflow property to deal with overflow behavior, shorthand for overflow-x: and overflow-y properties

position: absolute
right:0;
bottom:0;
	by default, height and width properties are applied to content box and do not include padding, border, margin areas.
	box-sizing property can change this default behavior, box-sizing: border-box

position: fixed; top:0	absolute: px
FLOATING ELEMENTS	%: relative to size of container, vw,vh: relative to the viewport, em,rem: relative to the font size of the current element for em and relative to the font size of the current root element for rem, if there isn't font size for element then it inherits from parent or html element if no parents have font size
FLEXBOX, flexible box layout	With relative units, layouts can be more scalable and responsive. There are still some use cases for px.
GRID	property that controls depth axis, positive number bring element closer to you, negative number moves it further away

HIDING ELEMENTS	we can tell browser to use default system font stack on user's computer, different than web safe fonts bc they are more up to date, benefits: can boost performance, no FOUT, native look, overall better experience, disadvantages: default fonts vary,
MEDIA QUERIES	property to size fonts, do not use px bc pixels are not consistent among devices, retina display, use relative units like rem
3 font families	margin: use margin to keep elements that are related closer to each other line-height: good benchmark is 1.5, no units, to multiply font-size by 1.5
Where to implement font stack style?	letter-spacing:controls space between letters of text, don't use rem values use px word-spacing: controls space between words ideal line length should be between 50-70 characters so you can use this trick to achieve styling, p { width: 50ch}, sets width to 50 zeros
web safe fonts	raster: made up of pixels, come from cameras/scanners, JPG, PNG, GIF, etc, can be blurry if picture is too small and scaled up, takes long time to load if big size vector: defined by mathematical vectors like lines and curves, SVG (scalable vector graphics) format, look sharp at any size, much smaller than raster images, be sure to use for logos, graphics, icons, and simple backgrounds!
Flash of Unstyled Text (FOUT)	optimization technique that implements fewer http requests by combining images into one image so there is only one http request for all of the images, and splices the images in css to use individual image, disadvantages: file size can get too larges, sprites are not flexible --> advantages: effective for icons and logos

system font stack	display: none: property hides element and removes from DOM visibility: hidden: property that hides element but it is still on DOM
font-size: 2rem;	@ media typeOfDevice and (condition){ styles to apply if condition is true} implement responsive design, easier to go mobile first approach, then adjust for bigger screens as you go
VERTICAL SPACING	serif: more professional, ex georgia, times new roman sans-serif: more playful, ex avenir, arial, futura, helvetica, roboto monospace: all widths of characters are the same, ex consolas, courier, ubuntu
HORIZONTAL SPACING	implement font stack style in body so all elements can inherit the font
2 formats of images	fonts most browsers are guaranteed to be able to implement but outdated
CSS SPRITES	may occur if it takes awhile to load font resources. Backup font loads first, then once the resources load then the font will change. font-display property can provide you options to deal with this problem, mosh's prefferred value is font-display: optional

DATA URLS, formerly known as data URIS

WebP format is a format of raster images, benefit is much smaller file size, cost is that it's not quite as sharp as jpg, but very close, not supported by IE

physical resolution

ue transition property to animate one or more properties instead of going from point A to B instantly, we use it for simple animations

logical resolution

follow naming convention
create logical sections in your stylesheet, have multiple stylesheets that get combined with sass or less
avoid over-specific selectors: avoid direct child selectors to avoid breaking changes, remove element types in hierarchies and instead resort to classes, use specific names for classes to avoid having to keep overwriting styles in different areas, Limit nesting to two or maximum three selectors

Device Pixel Ratio (DPR)

avoid !important keyword,
extract repetitive patterns with object-oriented CSS,
implement DRY principle, (don't repeat yourself: use css variables

High Density Screen

```
:root {
/* where we can define custom properties or global variables */
--color-primary: yellow;
--border-size:2px;
--border-radius: 10px;
}
.box { background: var(--color-primary);}
```

how to support high density screens for fixed image size:

seperate container from content: style components without container, and then you can reuse those components in different containers without having to redefine styles, seperate structure from skin: define style with common structure for components, and then for specific "skin" or look define seperate class with that specific skin, ex .btn and .btn-gold, always list more generic class first then more specific class next

USING MODERN IMAGE FORMATS	another optimization technique to reduce HTTP requests by embedding image directly into stylesheet or HTML document, mosh personally doesn't think benefits outweigh costs
transition: propertyWeAreUpdating duration ease-in/linear/ease-out/cubic-bezie()	based on actual number of pixels on device
best practices for writing clean, maintable css part 1	how the screens behave in displaying pixels
best practices for writing clean, maintable css part 2	ratio of physical/logical
CSS VARIABLES, or custom properties	screen with DPR > 1
OBJECT-ORIENTED CSS principles	to solve blurry image problem, we resize image to equal original image size x DPR

BEM, Block Element Modifier

media object, or media components

call out components

color palette

popular css naming conventio that thinks of webpage as reusable components that contain elements and can be modified, blockName__elementName, blockName--modifier, blockName__elementName--modifier

popular design for component where you have icon on left/right and then heading and text on other side

call on user to take action

composed of primary, secondary, and accent color, and other main colors