<Semester IV>

</Programming for Interactive Interfaces//>

Week 4

Types of Units of Measurement

The numeric type used most frequently is <length>. For example, 10px (pixels) or 30em. There are two types of lengths used in CSS – relative and absolute. A whitespace cannot appear between the number and the unit. However, if the value is 0, the unit can be omitted. For some CSS properties, negative lengths are allowed.

Absolute

- The absolute length units are fixed and a length expressed in any of these will appear as exactly that size.
- Absolute length units are not recommended for use on screen, because screen sizes vary so much. However, they can be used if the output medium is known, such as for print layout.

Relative

- Relative length units specify a length relative to another length property. Relative length units scale better between different rendering medium.
- The benefit of using relative units is that with some careful planning you can make it so the size of text or other elements scales relative to everything else on the page

Units of Measurement

ABSOLUTE

Pixels (px)

Inches (in)

Centimeters (cm)

Millimeters (mm)

Points (pt)

Picas (pc)

RELATIVE

Percentages (%)

Font sizes (em, rem)

Character sizes (ex,ch)

Viewport dimensions (vh, vw)

Viewport max (vmax)

Viewport min (vmin)

Relative Units of Measurement

Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element

Relative Units of Measurement

em: In CSS , the “em” unit refers to the font-size of its parent element, defaulting to the root element (<html>) if it’s the only one in the HTML document. Or the default font size set by the browser.

Eg. If a parent div has a font-size of 20px, and it has a child <p> element where the font size is set to 2em, then the text in that <p> will be 40px.

rem: In CSS, rem is based on the font-size of the root element (<html>) and stays the same in all cases.

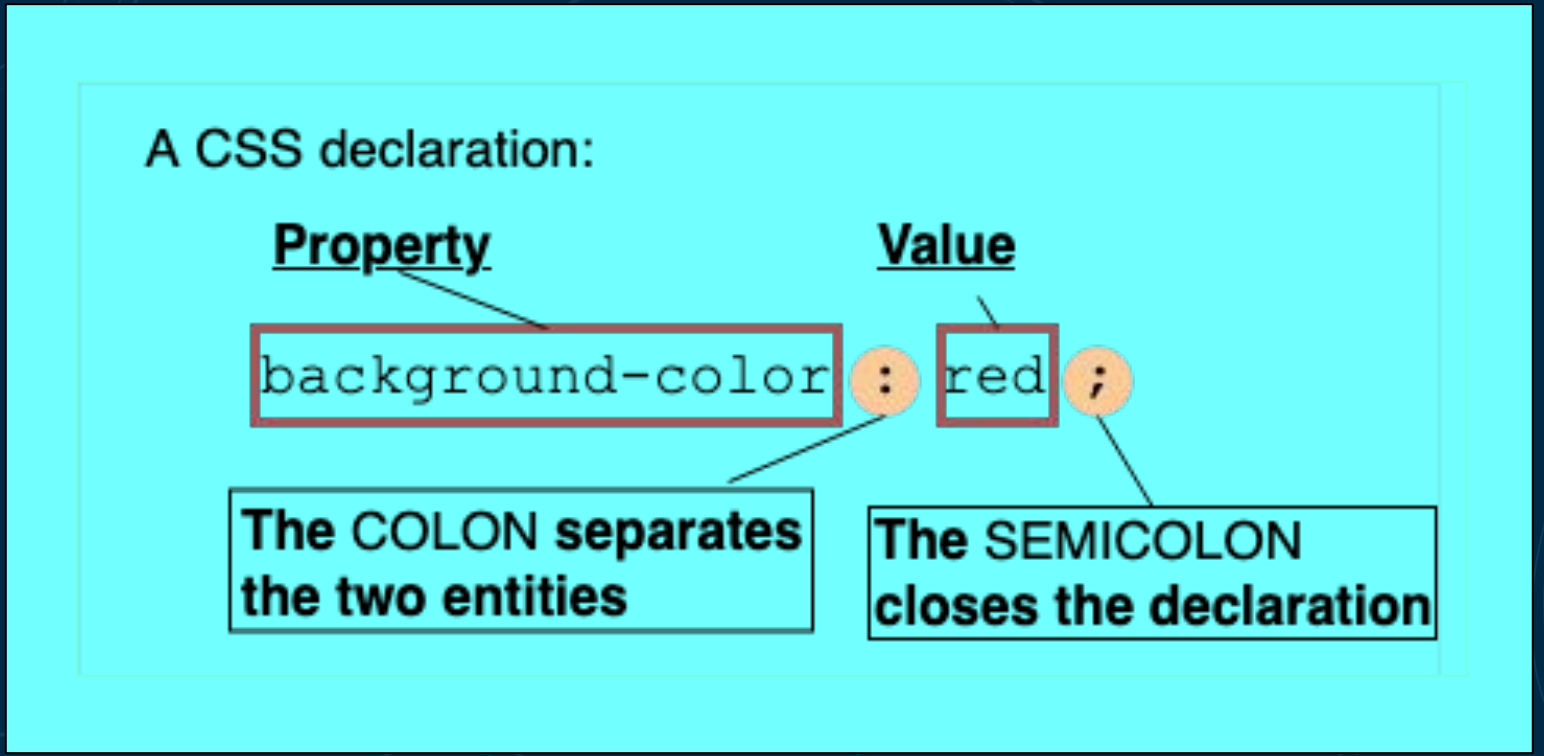
vw or View-Width: In CSS, vw depends on the viewport width, which changes with screen size, so an Android phone has a smaller vw than an HD TV. It is given in percentage %

vh or View Height: The vh unit in CSS is 1% of the viewport height, useful for responsive design_to scale elements with the window size.

Eg. .myImg { height: 10vh; width: 50vw;}

In this code, the image’s height is 10% of the viewport height and its width is set to 10% of the viewport width

CSS Syntax


A CSS declaration:
Property
Value
background-color
:
red
;
The COLON separates the two entities
The SEMICOLON closes the declaration

CSS Syntax

CSS Pseudo Class

A pseudo class is basically a pseudo state of element which can be later targeted with CSS. They can be used to target and style those elements which cannot be targeted with normal CSS classes or Id. A CSS *pseudo-class* is a keyword added to a selector that lets you style a specific state of the selected element(s). For example, the pseudo-class :hover can be used to select a button when a user's pointer hovers over the button and this selected button can then be styled.

Pseudo-classes let you apply a style to an element not only in relation to the content of the document tree, but also in relation to external factors like the history of the navigator (:visited, for example), the status of its content (like :checked on certain form elements), or the position of the mouse (like :hover, which lets you know if the mouse is over an element or not).

Categories of Pseudo Classes

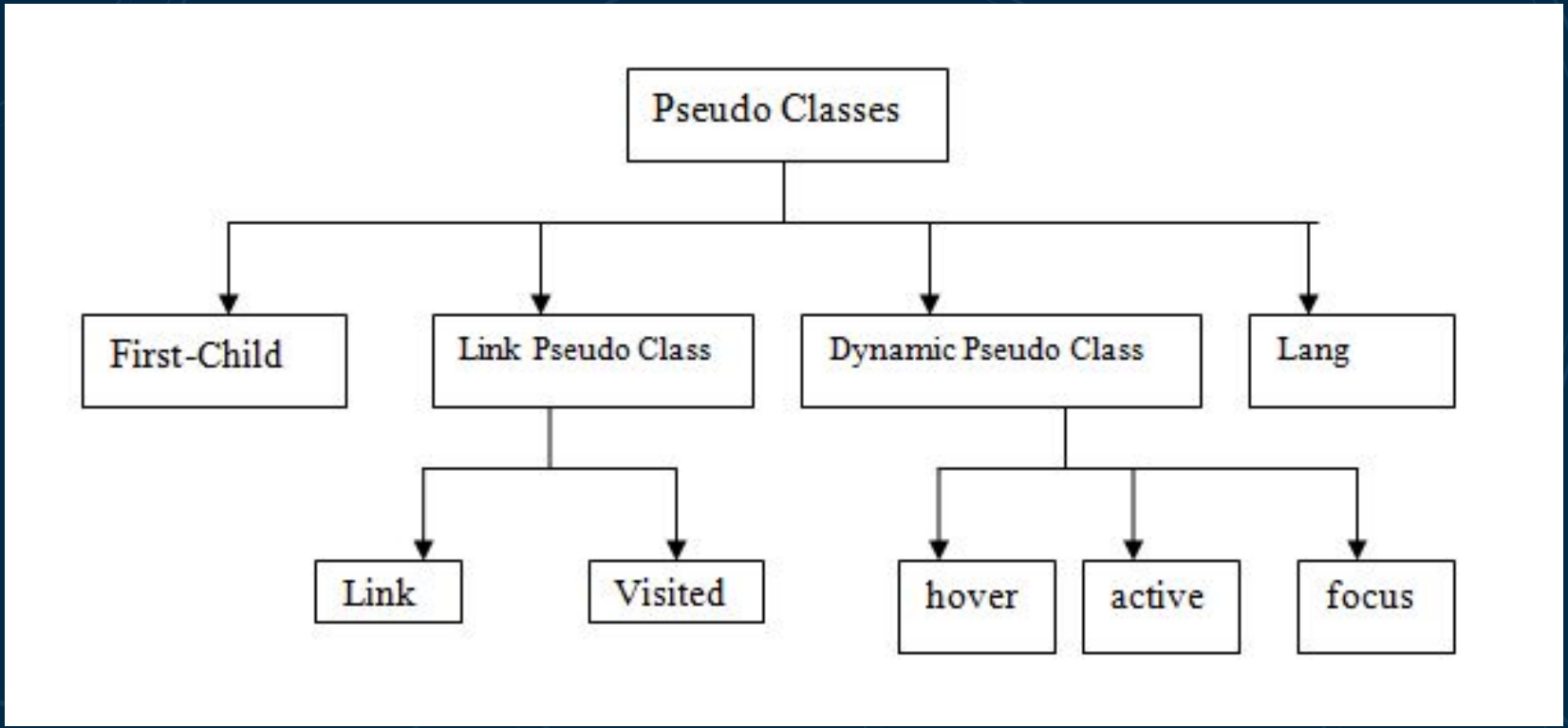
Dynamic

- Some pseudo-classes only apply when the user interacts with the document in some way. These **user-action pseudo-classes, also called dynamic pseudo-classes**, act as if a class had been added to the element when the user interacts with it.
- Examples include: hover, focus, active, disabled, visited, required, etc.

Structural

- Structural pseudo-classes are a subset of CSS pseudo-classes that allow you to select and style elements based on their position in the document tree.
- They include :root, :nth-child(), :nth-last-child(), :nth-of-type(), :nth-last-of-type(), :first-child, :last-child, :first-of-type, :last-of-type, :only-child, :only-of-type, and :empty.
- Each of these pseudo-classes has a unique function that allows for more precise styling of HTML elements

CSS Pseudo Class




CSS
3
HTML
5
PSEUDO
CLASSES
button:hover
IN 7 MINUTES!

CSS Pseudo Class

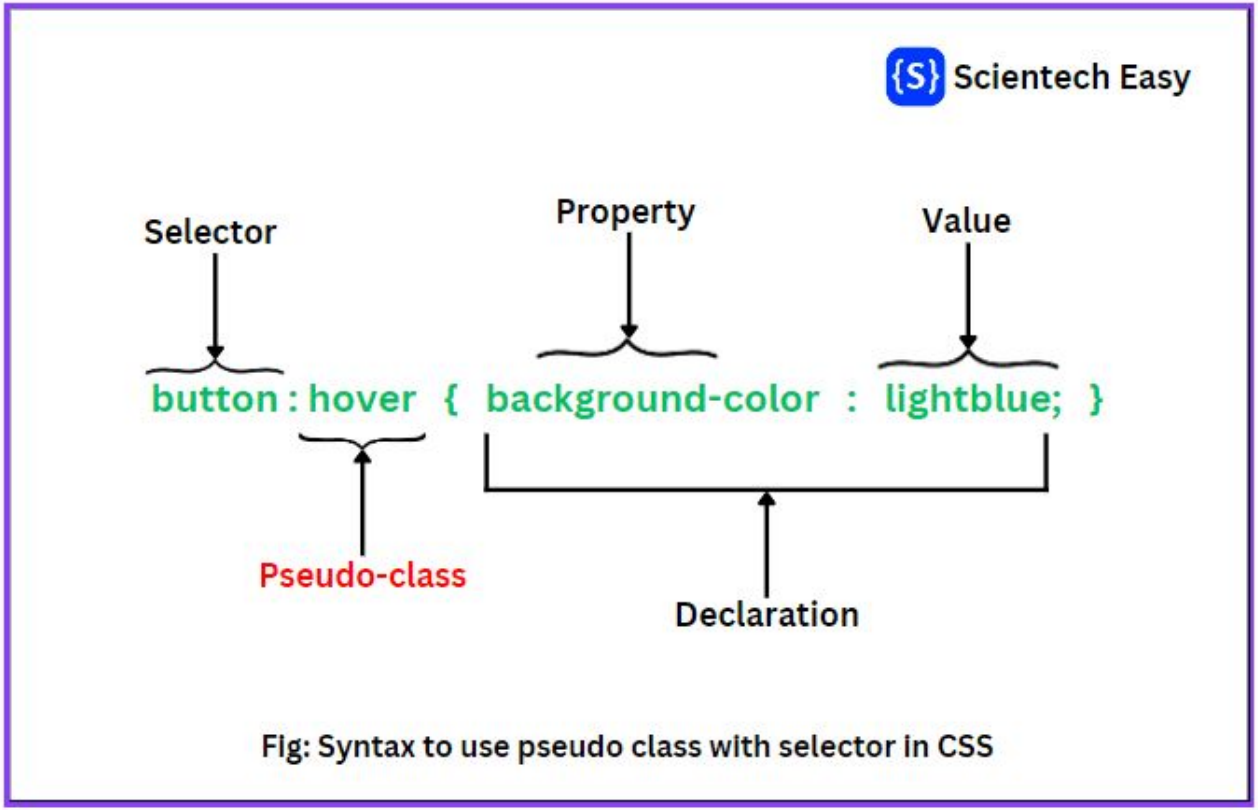
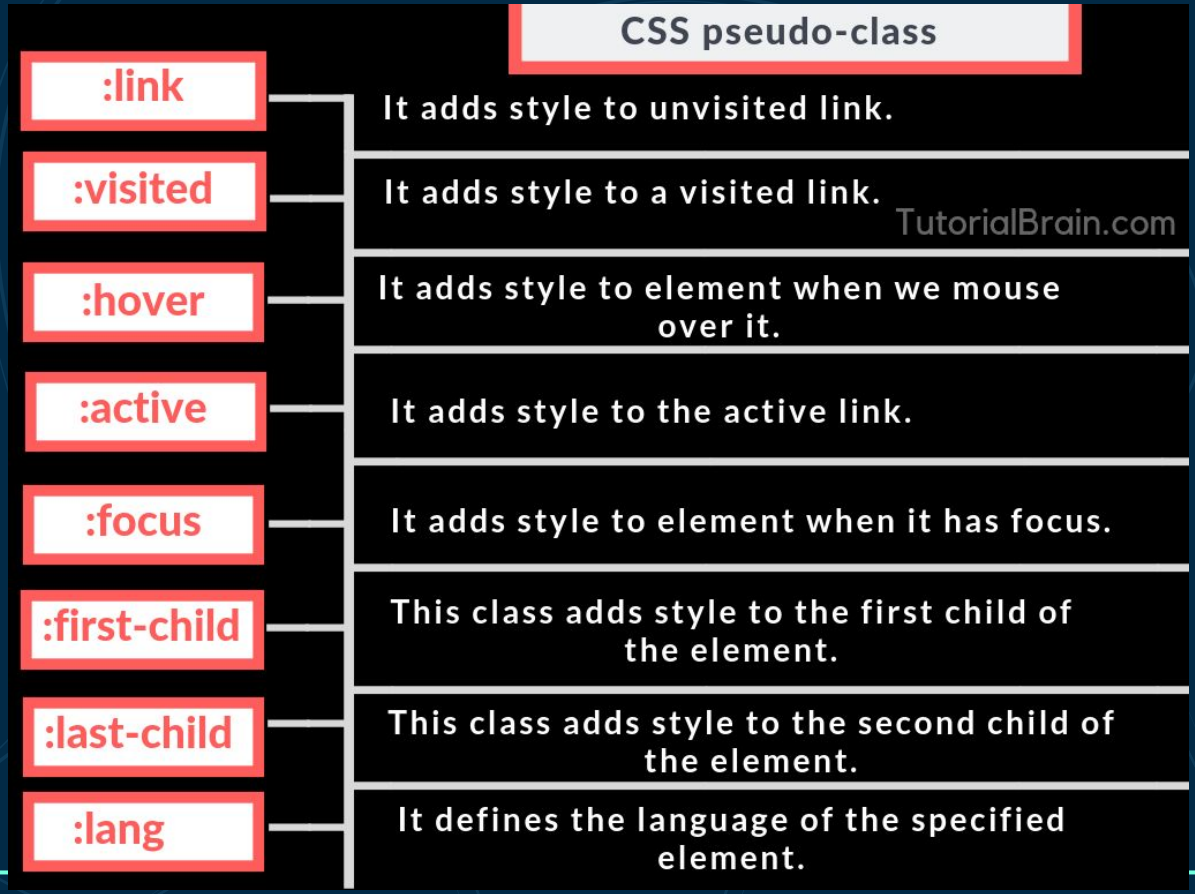


Fig: Syntax to use pseudo class with selector in CSS

CSS Pseudo Class



CSS Pseudo Elements

A CSS pseudo-element is a keyword added to a selector that lets you style a specific part of the selected element(s).

Double colons (::) are used for pseudo-elements. This distinguishes pseudo-elements from pseudo-classes that use a single colon (:) in their notation.

Pseudo-elements do not exist independently. The element of which a pseudo-element is a part is called its *originating element*. The last element in the selector is the originating element of the pseudo-element.

```
eg. div p::first-line {
 color: blue;
}
```

here div is considered the “originating element” for any pseudo elements declared here

CSS Pseudo Elements

CSS Pseudo Elements

Pseudo-elements (8)
::after
::backdrop
::before
::first-letter
::first-line
::placeholder
::selection
::slotted()

Pseudo-classes (48)
Linguistic pseudo-classes (2)
Location pseudo-classes (7)
User action pseudo-classes (5)
Time-dimensional pseudo-classes (3)
Resource state pseudo-classes (2)
The input pseudo-classes (17)
Tree-structural pseudo-classes (12)

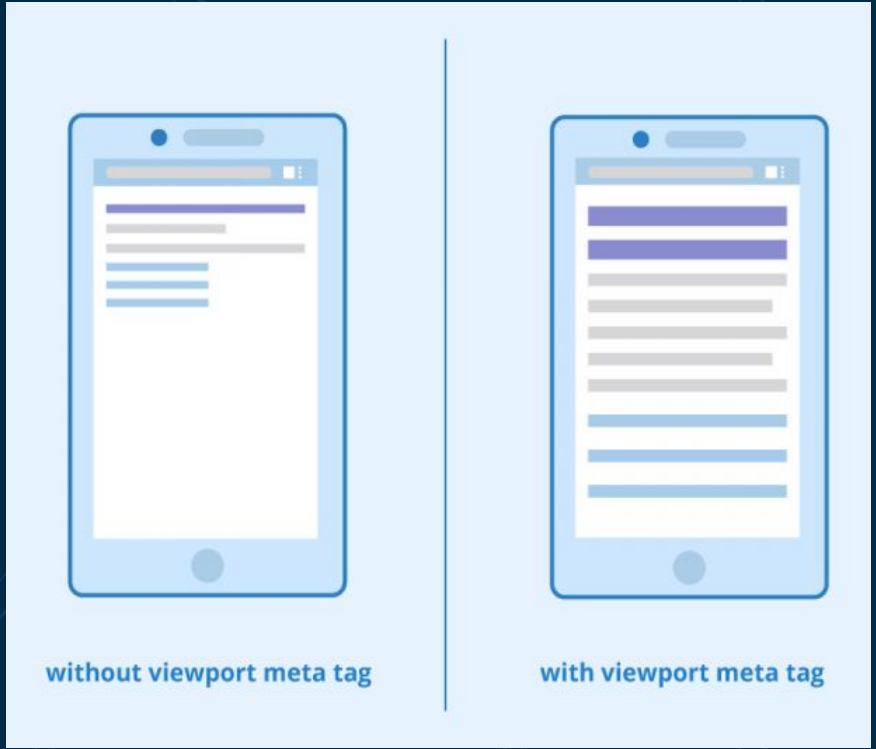
Responsive Web Design

Responsive web design, originally defined by Ethan Marcotte in A List Apart, is a design strategy that responds to users' needs and their devices' capabilities by changing a site's layout to suit the device being used. Because internet-capable devices have so many possible screen sizes, it's important for your site to adapt to any existing or future screen size. Modern responsive design also accounts for modes of interaction such as touch screens. The goal is to optimize the experience for everyone. It’s no longer enough to design for a single device. Mobile web traffic has overtaken desktop and now makes up the majority of website traffic, accounting for more than 51%. The example design scales perfectly well as the browser window resizes, but stress points quickly appear at lower resolutions.

Viewport

The term viewport refers to the size of a window or visible area on a screen. In general, this term is used for displays on mobile devices such as smartphones and tablets.

Thanks to the viewport, websites on mobile devices are not displayed in the same way as on a desktop screen. Users do not have to zoom in but can view the content of a page in a way that matches the small display. Viewports as meta elements (in combination with responsive web design) help web browsers to break up the pages and reassemble them on small screens in a way that enables users to receive a meaningful and readable image.



Viewport Meta Tag

You can integrate viewports in your HTML files in two different ways: either directly in the document or in a referenced CSS file, whereby the syntax of these two alternatives only slightly differs.

If you want to include the viewport directly into your HTML file as a meta tag, you can use the following code:

```
<meta name=viewport content="width=device-width, initial-scale=1">
```

In this case, the width is defined so that it adapts to the respective device’s screen width (width=device-width). Initial-scale stands for the initial zoom factor and signals that your page is to be displayed 1:1 on the screen of a mobile device.

If you specify the viewport in a CSS file instead, you should put it right at the beginning of the file to ensure a correct display. Here's an example of how that code could look like: @viewport {

```
width: device-width;}
```

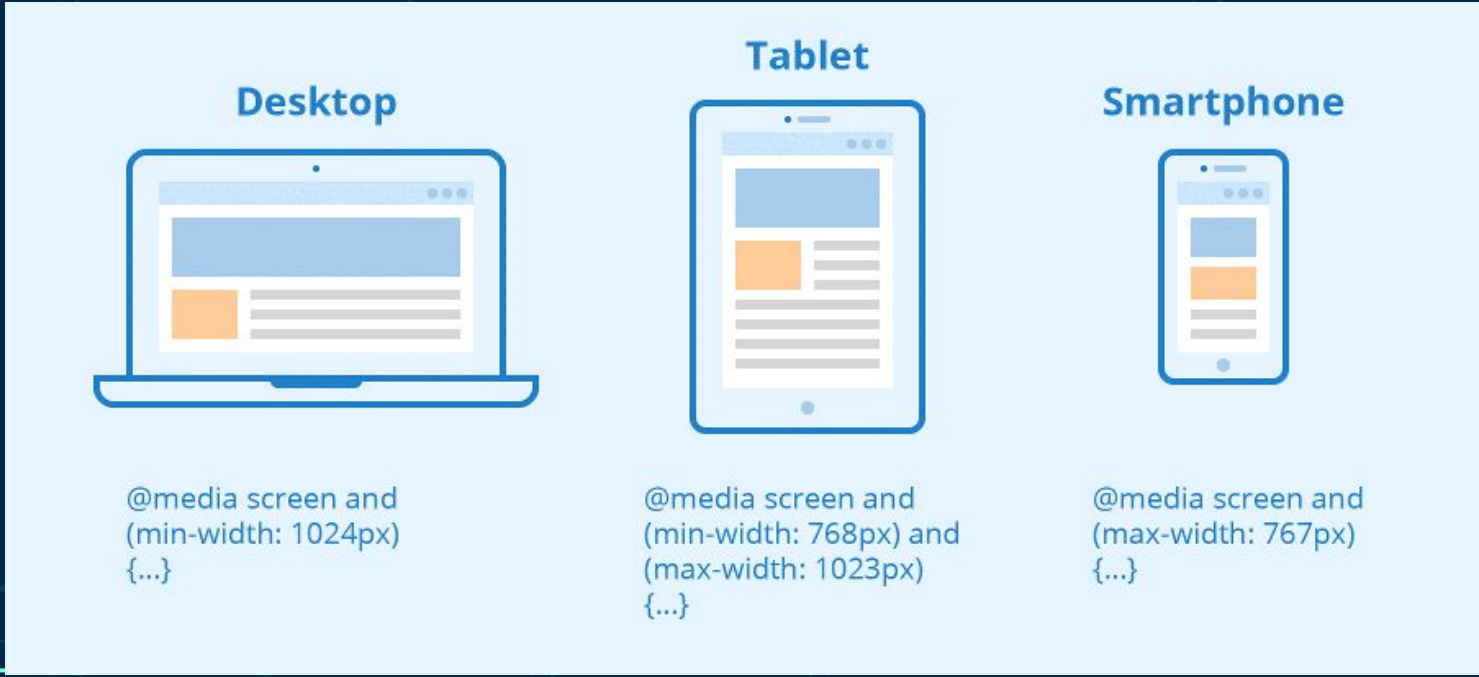

meta viewport
explained

Media Queries

Media queries are a CSS3 function that allows you to specify the display of a document for different output media and screen sizes. The display is adjusted according to certain conditions such as media type, screen orientation or screen resolution. Since June 2012, media queries have been a standard for responsive web design recommended by the World Wide Web Consortium (W3C). An essential part of responsive design is creating the right user experience for each type of device. Media Queries make it possible to respond to a client browser with a customized display of a web page based on inherent properties of the device it is running on. Before a web page is loaded and displayed by a browser, this media information is retrieved via media queries. CSS media queries thus offer an easy way to respond to many output devices and are therefore an important part of mobile optimization and responsive web design.

Media Queries

A media query is a fundamental part of CSS3 that lets you render content to adapt to different factors like screen size or resolution.



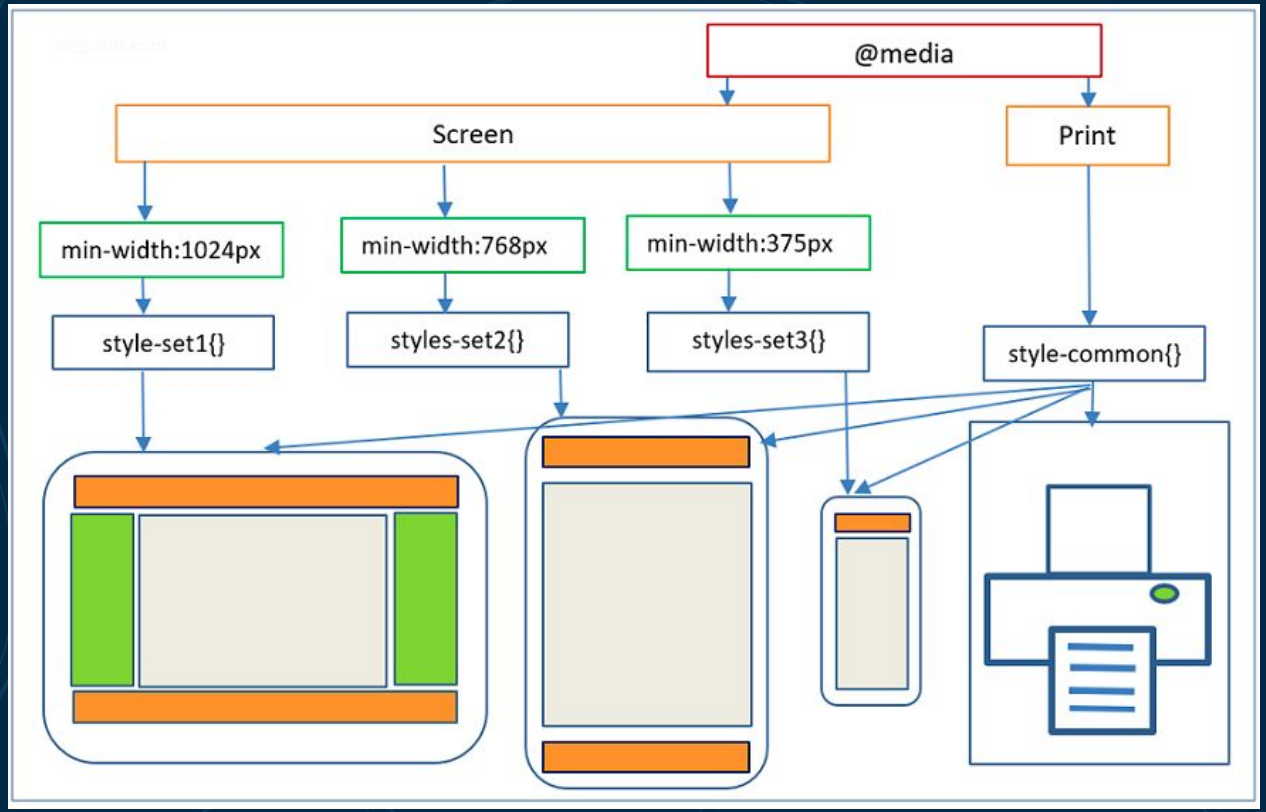
Media Queries Syntax

- The keyword media has to be used after an @ symbol. Then, specify media type and media features e.g.:

```
@media only screen and (max-width: 600px) {
    body {
        background-color: lightblue;}
    }
```

- Media types define the broad category of device for which the media query applies: all, print, screen.
- Media features describe a specific characteristic of the user agent, output device, or environment. Eg. device-width, hover, height, display-mode, aspect ratio, etc
- Logical operators can be used to compose a complex media query: not, and, and only. You can also combine multiple media queries into a single rule by separating them with commas.

Media Queries Syntax



Media Queries

Responsive Images

Homework:

Compulsory Reading:

Responsive Images - MDN

And apply what you learnt from this to your code.

Homework: Try different units of measurement, pseudo classes and pseudo elements. Add the meta tag and media queries to your website. Make your images responsive

W3Schools HTML Tutorial

HTML: HyperText Markup Language | MDN

Reference

1. CSS Units W3Schools
2. CSS Pseudo Class Basic - Javatpoint
3. CSS Pseudo Class Types - Open Genius
4. Structural Pseudo Classes - Site Point
5. Pseudo Elements in CSS - W3Schools
6. Importance of Responsive Web Design - Web.Dev
7. Responsive Web Design Beginner’s Guide - Kinsta
8. Meta Viewport Tag - SEObility
9. Media Queries in CSS - SEObility
10. CSS Media Queries Syntax - MDN

Fonts & colors used

This presentation has been made using the following fonts:

Blinker
(https://fonts.google.com/specimen/Blinker)

Inconsolata
(https://fonts.google.com/specimen/Inconsolata)