CSS Architecture Cheat Sheet

by sathiyapriyan_j via cheatography.com/182774/cs/39105/

Line end

Intro to css

👉 Cascading style sheets

👉 CSS describes the visual style and presentation of the content written in HTML

👉 CSS consists of countless **properties** that developers use to format the content: properties about font, text, spacing, layout, etc.

Basic property to style

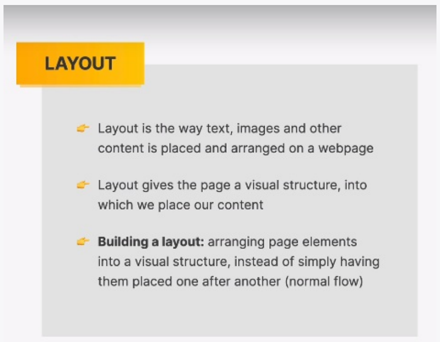
- ⏩ font-size: 26px;
- ⏩ font-family: sans-serif;
- ⏩ text-transform: uppercase;
- ⏩ font-style: italic;
- ⏩ text-align: center;
- ⏩ line-height: 1.5;
- ⏩ font-weight: bold;

Colour

Dev tools

`- So, Every modern browser contains a set of tools which makes our lives as developers a lot easier. And we call these tools collectively developer tools or for short dev tools.

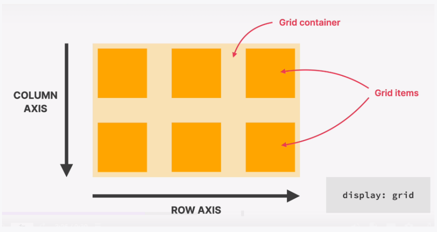
Flex



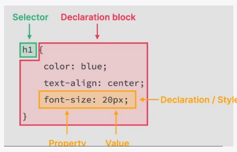
LAYOUT

- Layout is the way text, images and other content is placed and arranged on a webpage
- Layout gives the page a visual structure, into which we place our content
- **Building a layout:** arranging page elements into a visual structure, instead of simply having them placed one after another (normal flow)

Grid



Syntax



```
h1 {
    color: blue;
    text-align: center;
    font-size: 20px;
}
```

Combain selectors

Develop code by combining code selectors

Doing that

Combining a selector into two ways

1.List selector

2.descent selector

Suppose if we change html and edit it is hard to maintain future. that is not good idea

Best practice :

Set a selector name and select them

Style link

```
a:link {
color: aqua;
text-d eco ration: none;
}
a:visited {
color: aqua;
}
a:hover {
color: red;
font-w eight: bold;
text-d eco ration: underline /
dooted or wavy / orange;
```

Style link (cont)

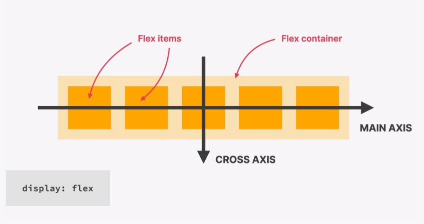
> }

a:active {

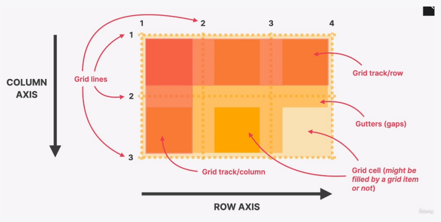
background-color: #1098ad;

}

Flex



Terms



3 ways to add css

1.Inline

2.Internal

3.External

Combain selector name

Ids and classes

biggest difference is not allowing to repeat id names

or in other words use id only one times

A Class name can be used by multiple HTML elements, while an ID name must only be used by one HTML element within the page

id = #

class = .

CSS Architecture Cheat Sheet

by sathiyapriyan_j via cheatography.com/182774/cs/39105/

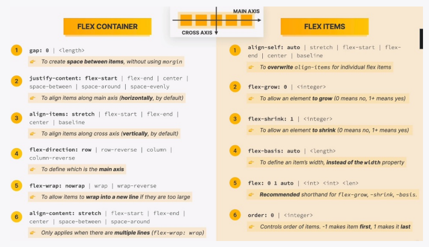
Combain selector name (cont)

Confilicts : skill now use multiple propery in different places but apply id or class property skip !!
id vs class class is always better . to avoid bugs

Pseudo classes

```
/* CSS automatically figure out
Element inside a container */
/ It is use full for colour
tables for backgroud colour /
When we mix multiple elements
inside of a parent element,
then these pseudo -cl asses
don't work really well.
A pseudo -class is used to
define a special state
 of an element.
li:fir st- child {
font-w eight: bold;
}
li:las t-child {
font-s tyle: italic;
}
/ Target a specific element in a
group /
li:nth -ch ild(2) {
color: red;
}
/*
li:nth -ch ild(2) {
color: red;
}
parameter - odd or even
*/
article p:last -child {
color: red;
}
```

Flex property


MAIN AXIS
CROSS AXIS

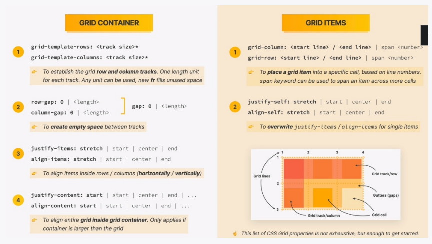
FLEX CONTAINER

1. `gap: 0 | <length>`
   To create **space between items**, without using `margin`
2. `justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly`
   To align items along main axis (**horizontally**, by default)
3. `align-items: stretch | flex-start | flex-end | center | baseline`
   To align items along cross axis (**vertically**, by default)
4. `flex-direction: row | row-reverse | column | column-reverse`
   To define which is the **main axis**
5. `flex-wrap: nowrap | wrap | wrap-reverse`
   To allow items to **wrap into a new line** if they are too large
6. `align-content: stretch | flex-start | flex-end | center | space-between | space-around`
   Only applies when there are **multiple lines** (`flex-wrap: wrap`)

FLEX ITEMS

1. `align-self: auto | stretch | flex-start | flex-end | center | baseline`
   To **overwrite** `align-items` for individual flex items
2. `flex-grow: 0 | <integer>`
   To allow an element **to grow** (0 means no, 1+ means yes)
3. `flex-shrink: 1 | <integer>`
   To allow an element **to shrink** (0 means no, 1+ means yes)
4. `flex-basis: auto | <length>`
   To define an item's width, **instead of the `width`** property
5. `flex: 0 1 auto | <int> <int> <len>`
   Recommended shorthand for `flex-grow`, `-shrink`, `-basis`.
6. `order: 0 | <integer>`
   Controls order of items. -1 makes item **first**, 1 makes it **last**

Properties


1
2
3
4
Grid lines
Grid track/row
Gutters (gaps)
Grid track/column
Grid cell

GRID CONTAINER

1. `grid-template-rows: <track size>*`
   `grid-template-columns: <track size>*`
   To establish the grid **row and column tracks**. One length unit for each track. Any unit can be used, new **fr** fills unused space
2. `row-gap: 0 | <length>`
   `column-gap: 0 | <length>`
   `gap: 0 | <length>`
   To **create empty space** between tracks
3. `justify-items: stretch | start | center | end`
   `align-items: stretch | start | center | end`
   To align items inside rows / columns (**horizontally** / **vertically**)
4. `justify-content: start | start | center | end | ...`
   `align-content: start | start | center | end | ...`
   To align entire **grid inside grid container**. Only applies if container is larger than the grid

GRID ITEMS

1. `grid-column: <start line> / <end line> | span <number>`
   `grid-row: <start line> / <end line> | span <number>`
   To **place a grid item** into a specific cell, based on line numbers. `span` keyword can be used to span an item across more cells
2. `justify-self: stretch | start | center | end`
   `align-self: stretch | start | center | end`
   To **overwrite** `justify-items` / `align-items` for single items

This list of CSS Grid properties is not exhaustive, but enough to get started.