|  |  |
| --- | --- |
| **Name-Knowledge** | **Meaning** |
| Element | Element = tag open + content + tag close eg <p> Hello </p> |
| Attribute | <div class =”main”> content </div> | class=name attribute, main=value At., class=”main” At. |
| Name attribute | Told what is the nature. Eg width, height, position |
| Value attribute | Told what is value of name attribute eg 120px, left |
| <!—content --!>  </\* --------------\*/> | Use comment in html |
| Link css file to html file | <link rel="stylesheet" type="text/css" href="name of css" /> use in tag head |
| Element[attribute] { } | You can specific the element with attribute |
| Box model | Box: content 🡪 padding 🡪 border🡪 margin |
| px | 1/96th = 1 inches = 2.54 cm |
| rem | 1 time of html file = 16px |
| em | Reference from parent element eg parent = 16px 2em = 32 px |
| RGB | rgb(0,0,205,0.8) 0.8 = alpha value |
| Hexadecimal | #97 05 15 |
| declaration | { property: property value; } eg { background-color: property red; } |
| Property | What’s the style to change eg background-color |
| Property value | Value of the style eg red |
| flex | 1 container with item1 item2 item3 item4 item5  Element with display:flex; will be container. The child in that element will be item  When area in container < all item (all item will be shrank in to container)  When area in container > all item (all item will show the real area the other area is contain.) |
| Main-axis | Normally it is Row side with left to right we have to define it what’s direction |
| cross-axis | 90 degree from main-axis in this case it is column side from top to bottom |
| &nbsp | Use like spacebar in html |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| <!DOCTYPE html> | Told the browser which what’s version in html. (We use xhtml transitions) |
| <html> </html> | Told the browser which in browser arrange by html. |
| <head> </head> | Component of all the back office in website. Normally it did not show up. |
| <meta charset = “utf-8”> | It has many meta in one website. It explains any information. eg. Keyword seo, writer. |
| <title> </title> | Component of Name in website. (tab in website not in www.) |
| <body> </body> | Component that control the detail in front office at website. (pic table video link etc) |
| <h1></h1> - h6 | Size of Header.h1 is biggest |
| <p></p> | Paragraph |
| <div></div> | Line Breaks (Block element) |
| <span></span> | Line Breaks (Inline element) |
| <br>,</br>,<hr>,</hr> | Line Breaks --- hr will add line |
| <i></i>,<em></em> | Italic |
| <b></b>,<strong></strong> | Bigger |
| <mark></mark> | Highlight |
| <small></small> | Smaller |
| <del></del> | Cross out in center of word |
| <ins></ins> | Underline |
| <sub></sub> | Hang under the word |
| <sup></sup> | lift over the word |
| <a>content</a> | Link to other thing eg other website. Use with attribute href, target |
| /href=”www.google.com” | Link to google |
| /target=”\_blank” | Open the link with new tab |
| <img/> | Show the picture in website Use with attribute src, alt, width, height |
| src=”link wit .jpg” | Use the picture from that website |
| alt=”content” | Content will show in case of link is unavailable |
| /width=”a” | Wide of the things eg pic, table |
| /height=”a” | long of the things eg pic, table //height:100vh; = viewport height 100% normally use in body |
| <table></table> | Inform html it’s table in there |
| /align=”center” | Specify the position of the things eg table to center, pic to the left |
| <thead></thead> | Header of table (uppest) |
| <tbody></tbody> | Detail of table (body) |
| <tfoot></tfoot> | Footer of table (lowest) |
| <tr></tr> | Use in thead, body, foot will show the detail in one row |
| <th></th> | Write the content in table (normally Bold) eg 3 rows wiul lhave <th> 3 times |
| <td></td> | Use like the <th> but it normal text |
| /colspan=”a” | Merge column from 2 to 1 |
| /rowspan=”a” | Merge row from 2 to 1 |
| <ol></ol> | Order list eg 1. |
| <ul></ul> | Unordered list eg black point |
| <li></li> | Tag in ol or ul will show what is content in 1 2 or unordered list |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| <link rel="icon" href="www."> | When you use the website the icon in link www. Will show to your icon website |
| <form></form> | Tag to show the following is the form |
| <label></label> | Name or information to tag input |
| <input /> | Use in form for show what the attribute |
| /type=”text” | Text box |
| /name=”content” | Variable name use with type=text, radio etc |
| /type=”number” | number box |
| /type=”range” | Use like the number but it’s range |
| /min=””, max=”” | Min and max of number eg min 1 max 500 raise number 1,2,3,…,500 use with number,range |
| /step=”” | The number will raise by step eg step=10 1,11,21,31 |
| /type=”radio” | Choose only one in checkbox use with name, value attribute |
| /value=”content” | Use it to show content in data. It can use in any attribute in tag form |
| /type=”date” | Choose the date |
| /type=”reset” | Click to reset all data in form |
| /type=”submit” | Click to submit the data |
| /type=”checkbox” | Use like radio but we can choose multi |
| /type=”password” | Use like text but it show \*\*\*\*\* |
| <select></select> | It shows all data to drop down data |
| <option>context</option> | Use in tag select to show the context in each drop down list (normally use with value=””) |
| <datalist></datalist> | Use to show the drop down list in empty text form when you click on the empty text drop downlist with content 1 will show up  <input List=”text” name=”test”/> <datalist id=”text”> <option value=”content1” </datalist> |
| <textarea></textarea> | Use this to expand area of textbox in form  <textarea name="add" rows="3" cols="40"></textarea> |
| <button>context</button> | Button with name context, pic will show up normally use with type=”submit” |
| /form=”text” | Use to connect with what ‘s form |
| <style></style> | Write in tag head only for inform all css |
| <section></section> | represents a generic standalone section of a document, which doesn't have a more specific semantic element to represent it. Sections should always have a heading, with very few exceptions. |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \color: ; | Change color to text eg pink |
| \font-size: ; | Change font-size to text eg 150px, smaller |
| \font-family: ; | Change the font eg Arial, sans-serif, Helvelica --- can find in cssfontstack.com |
| \font-weight: ; | Change weight of font eg normal, bold , lighter, bolder, 100 to 900 |
| \font-style; : | Style of font eg italic, bold, |
| \font: A B C/D E; | \font: italic bold 30px/1.4 Arial; = Font-style weight size/height family |
| \background-color: ; | Change background color to text eg pink value of it can be transparent |
| \text-align: ; | Change the position of text eg. Left(start), right(end), center, justify  Text will start with value |
| \text-align-last: ; | Text will start in the left and end in the value |
| \text-decoration: ; | Décor to text eg underline, overline, underline overline, way underline |
| \text-transform: ; | Transform the text box eg capitalize = He does horse = He Does Horse |
| \line-height: ; | Line-height makes more space between lines of text eg 1.4 |
| \letter-spacing: ; | Each letter will have spacing eg Normal, 1px |
| \background-image: url(https:) ; | Use image to background normally use it width height, background will run continued on eg pic = 300x300 size = 600x600 the background will appear 2 pic  Eg background-image: url("xxx.gif");  background-image: url("img\_tree.gif"), url("paper.gif"); img-tree will top of paper |
| \background-repeat: repeat-x/y/n/no-repeat; | x = it repeats pic in x but y is the real background / y = it repeats pic in y but x is the real background / n, no-repeat it’s not repeats all the x and y |
| \background-size: cover; | When use it the picture will adapt to 1 pic to any size  Eg. background-size: auto; it will adjust size auto  background-size: 300px 100px; 300px = wide , 100px = high |
| \background-position: ; | Set the position of background value is left, center, right, top, bottom or %, px  left top = center, x% y%, 0px 0px = top left corner first is far from left then second is right |
| \background-attachment: ; | scroll The background image will scroll with the page. This is default  fixed The background image will not scroll with the page (fixed)  local The background image will scroll with the element's contents |
| \cursor: pointer; | Mouth in viewport will transform to the hand with index finger |
| \box-shadow: ; | It’s shadow of box eg box-shadow; 5px 10px 20px red; = right shadow = 5px bottom shadow =10px blur radios is 20px and color of shadow is red |
| \letter-spacing: ; | It defined what spacing of each alphabet |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \height: 100 px;  \widht: 100 px; | Elements wide is 100px, long 100px |
| \border: 3px solid red; | Border with color type of line and width if value is none it be gone |
| \outline: green solid 5px; | Outline with color type of line and width (it outside the border) it can be none |
| \border-radius; 100%; | The corner of box/border will curve to circle eg border-radius: 20px;, 100% |
| \padding: 30px; | value is % , px , pt , cm , etc if specific to padding-top: / padding-right: / padding-left: / padding-bottom:  shortcut to padding: 6px 11px 4pm 9px; = top right bottom left  if padding: 6px 0; = top&bottom=6 left&right=0 |
| \margin: 30px; | value is % , px , pt , cm , etc if specific to margin-top: / padding-right: / padding-left: / padding-bottom:  shortcut to margin: 6px 11px 4pm 9px; = top right bottom left if margin: 6px 0; = top&bottom=6 left&right=0  if margin-left: auto; that css with this attribute will push to the right side |
| \min-width: ; \max-width: ; | Min and Max width if use it in browser it help you to read the website |
| \overflow: ; | If the element is intersect one is a two is b(b has many content than a)  :visible; – show normally (a and b show)  :hidden; - hide the content which over(a and b show but part of b not show)  :scroll; - can scroll the content which over(a will strict but can scroll to all of b) |
| \position: static; | normal |
| \position: relative; | Normal until add top: ; or left; : eg top: 200; left: 350; it’s far from top 200 px and far from left 350 px |
| \position: fixed; | Normal until add top: ; or right; : but it compare with the view port eg top: 50; right: 50;  it’s far from top of view port 50 px and far from right 50 px |
| \position: absolute; | It like the relative until you set position: relative; in parent the box with absolute position will reference from the box with relative position |
| \position: sticky; | Normally it’s relative position in case it ‘s big until scroll bar happen the position will be fixed |
| \z-index: a; | + will show before eg 4 3 2 1 0 -1 -2 the element with z-index 4 will show before 3 0 and -2 |
| \ !important | When use it to element that element will show even if it happen after |
| \opacity: ; | Adjust the transparent 0-1 |
| \transform: ; | rotate() eg 0 90 -0.25 3.142rad -- rotate  skew() eg 0 30deg 30deg,60deg [x,y] / skewx skew only x -- skew  scale() eg 1 1.3,0.4 -0.5,1 [x,y] == scale  translate() eg 0 32px,18px -2.1rem,-2ex 3ch,3mm [x,y] -- move |
| \transitions-property: ;  \transitions-duration: ;  \transitions-delay: ;  \transitions-timing-function: ;  Shortcut to transition: property duration timing timing-function delay | border-radius -- it focus to border-radius  eg 2s – it will use 2s to change activity  eg 5s – it will delay 5s to first activity before change  linear – it will smooth when changed  step(4,jump-start) – it take 4 step to change since start  ease-in -- slow in start and fast when near end  ease-out -- fast in start and slow when near end  ease-in-out -- slow in start, then fast and slow when near end |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \box-sizing: border-box; | Normally box-sizing in any element is different of size may be from padding or border then we use box-sizing: border-box; to defined it which equal sizing  Basically we use it with \* eg box-sizing with width 60px even border 100px width is 60px |
| \visibility: ; | Visible = default /hidden = hide that |
| @keyframes test {  from {  }  to {  }  } | In <style>  Test is a name to link with element or class or id  animation will start with from and end with to  from and to can change to 0%, 20%, 50%, 100% |
| \animation-name: ;  \animation-duration: ;  \animation-timing-function: ;  \animation-delay: ;  \animation-iteration-count: ;  \animation-direction: ;  Shortcut to \animation: name duration timing-function delay iteration-count direction; | Link to @keyframes in CSS  Eg 3s – time to animation  Linear, step(a,jump-start), ease-in, ease-out, ease-in-out  Eg 3s – delay 3s before anime start  Repeatation eg infinite  Normal= 0%-100%  Alternate = 100%-0% |
| \transform: translate(x,y) | Move to wherever px you use normally use with @keyframes eg 0% = (0,0) 100% = (200px,0) it will move form the corner of left to the right in x |
| \display: flex; | Defined that element to be container which has many child (will be inside the container) |
| \display: inline-flex; | The same of display:flex but it defined our child will inline |
| \display: inline-block; | In case of the element is inline element margin-top will not response if you want to make the margin to that element you have to use display: inline-block; eg <a> is inline element margin-top can not use |
| \flex-direction: row;  \flex-direction: column;  \flex-direction: row-reverse;  \flex-direction: column-reverse; | Default is row side with left to right  Column side with top to bottom  Row side with right to left  Column side with bottom to top |
| \justify-content: flex-start;  \justify-content: flex-end;  \justify-content: center;  \justify-content: space-around;  \justify-content: space-between;  \justify-content: space-evenly; | Default setting 123456………………  Every item will ship to the right ………………123456  All item is center  Each item will have space(left of1 and right of6 will have space half of the other)  Left of1 and right of6 will have not space the other have space  Every item and border will have space to equal |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \flex-wrap: ; | Main axis/flex-direction is row side. in case we raise the width in container with flex-wrap: wrap some item will go to line 2 if we use justify-content space will calculate in each line |
| \flex-wrap: wrap; | All item will be real size or width but still in container (split to 2 line), |
| \flex-wrap: wrap-reverse; | Main axis still the same of wrap; but item will start from cross axis bottom to top |
| \flex-wrap: nowrap; | default |
| \flex-wrap: wrap-reverse wrap-reverse; | The items will start from cross axis bottom to top and main axis right to left |
| \align-items: ; | Main axis/flex-direction is row it will manage only cross axis |
| \align-items: center; | The position will move to the center of cross axis top to bottom |
| \align-items: flex-start; | It will start to default point at the top of cross axis |
| \align-items: flex-end; | It will start to the ending point at the bottom of cross axis |
| \align-items: baseline; | Items are positioned at the baseline of the container eg size of text or no. of line |
| \align-items: stretch; | It span size of item to maximum in container |
| align-items vs align-content | This can be confusing, but align-content determines the spacing between lines, while align-items determines how the items as a whole are aligned within the container. When there is only one line, align-content has no effect. |
| \align-content: ; | Flex direction is column and flex wrap it not changing item it’s just moving of line if it not the flex:wrap it turn back from 2 line to 1 line |
| \align-content: flex-start; | All item will to start move from right to left |
| \align-content: flex-end; | All item will move to start point from main axis right to left space bel gone |
| \align-content: center; | All item will move to end point from main axis left to right space be gone |
| \align-content: space-between; | Items 1 and last column will close to border the others item will have space equal |
| \align-content: space-around; | Every item and border will have the same space |
| \align-content: space-stretch; | Every item will expand to the container |
| \align-self: ; | Focus to each item ed item6 by use id in item6 Flex direction is row |
| \align-self: flex-start; | Default |
| \align-self: flex-end; | Item6 will move to end point from cross axis top to bottom |
| \align-self: center; | Item6 will move to center of cross axis |
| \align-self: baseline; | Item6 will expand reference from base line of text of no.of line |
| \align-self: stretch; | Item6 will expand to equal container of cross axis |
| \flex-basic: ; | It we code width 1 value before when you specific value in flex-basic they will use value in flex-basic instead eg flex-basic: 50px; |
| \flex-grow: ; | Value is 0-1000/ we have to use it in id and link id to any item  If value =1 in item6 it will expand to the empty space in container to the border  If value =1 to all item it will expand in equal to the empty space in container to the border  If item6 value =1, item5 value = 2 it5 and it6 will expand to the empty space in container to the border by it5 has more space than it6 2 time |
| \flex-shrink: ; | It looks like a grow but in the opposite way value is 0-1000  In case container width = 500 px 6 item have 100 px per item  If value = 0 in item 6 the others item has shrank but item6 will has more space than the others  If value = 1 it’s default  If value = 1 in item6 it6 will shrink more than the others around 2 time |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \min-width: ; | Eg value is 20 px normally use in item6  The minimum width is 20 px when use it with grow and shrink  Flex-grow if item 6 real size grow from 5 to 10 px it will become 20 px  Flex-shrink if item 6 real size shrink from 50 to 10 px it will become 20 px |
| \max-width: ; | Eg value is 50 px normally use in item6  The maximum width is 50 px when use it with grow and shrink  Flex-grow if item 6 real size grow from 20 to 100 px it will become 50 px  Flex-shrink if item 6 real size shrink from 100 to 80 px it will become 50 px |
| Shorthand  \flex: flex-grow;  \flex: flex-basic;  \flex: flex-grow flex-basic;  \flex: flex-grow flex-basic flex-shrink; | 1 value is number it be flex-grow eg flex: 2;  1 value is %,em,px,min-content eg flex: 30%;  2 value 1 is number 2 is %,em,px,min-content eg flex: 2 30%;  3 value 1 is number 2 is number 3 is %,em,px,min-content eg flex: 2 1 30%; |
| @media | Each device has various resolution @media will help about this  One condition will happen if something changed |
| @media(width:a px) {  Element: {  }  } | Eg (width:500px)color:green if browser shrink to 500 px the text of element in there will change to green |
| @media(min-width:a px) {  } | Eg (min-width:500px) with color:green if browser grow more than 500 px the text of element in there will change to green |
| @media(max-width:a px) {  } | Eg (max-width:500px) with color:green if browser shrink more than 500 px the text of element in there will change to green |
| @media(max-width:a px) and (min-width:a px) {d  } | Eg (max-width:500px) and (min-width:200px) with color:green if browser grow or shrink between 200-500 px the text of element in there will change to green |
| @media(orientation:landscape) {  } | with color:green If our device will rotate to horizon side the text of element in there will change to green |
| \order: a; | Value is number it and be … -2 -1 0 1 2 … it defined what’s item who the first to be left size  Eg if we order item 1 is 1, item 2 is -1, item 3 is 0  Flex item is it2 it3 it1  Eg if we order item 1 is 1, item 2 is 1, item 3 is 1  Flex item is it1 it2 it3 |
| Shorthand  \flex-flow: value1 value2; | Value1 is flow-direction, value 2 is flex-wrap eg flex-flow: row wrap; |
| Game-flex-grid | <https://codepip.com/games/grid-garden/>  <https://codepip.com/>  grid garden, nth cart, disarray code crunchers |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| Grid container | Container that has many item in there eg 1 container in square |
| Grid cell | Each item in container eg 1 container has 12 grid cell row = 4 column = 3 |
| Grid column, Grid row | The line in column side / row side with all grid cell in their line eg first grid column has 4 grid cell / first row cell has 3 grid cell |
| Grid line | It’s the line which separate row and column out eg column grid line has 2 line |
| Grid area | We call the a x b to define what’s the area we use eg grid area 3x3 has 9 grid cell it can be 2x2 3x2 2x3 |
| Gap/Gutter | The distance between column or cell eg grid raw 1 and 2 distance around 10px (gap = 10px |
| \display: grid; | Must set for defined this container is grid  Eg we set 1 container has 4 grid cell |
| \grid-template-column: ;  \grid-template-column: apx bpx;  \grid-template-column: a% b%;  \grid-template-column: afr bfr;  \grid-template-column: repeat(a, bfr);  \grid-template-column:  repeat(a, bfr cfr); | Set template to column side if 1 value it’s default, if2 value the 4 grid cell will transfrom to 2 column 2 row  eg : 200px 100px; it has 2 columns 1 size = 200 px 2 size= 100px  : 50% 50%; it has 2 columns but 1 and 2 = half screen of browser  : 1fr 2fr; 2 column 1 = 1/3 2=2/3  : repeat(4, 1fr) = :1fr 1fr 1fr 1fr;  : repeat(4, 1fr 2fr) = 1fr in first column 2fr in second column 4 time |
| \grid-template-row: ;  \grid-template-row:  apx minmax(bpx,auto); | Use like grid-template-column: ; but it use in row the value in there is the same normally if it has only 1 value it’s = default (it must have 1 column)  It’s to set minmax size of grid cell  We set the container with 4 grid cell to 2x2 grid-template-column is 1fr 1fr grid-template-row: 100px minmax(100px,auto) and add <div> xxx</div> 10 times in item4  ==== grid cell all will be 2x2 and grid cell in grid row 2 will expand with the text of xxx until all xxx is in item 4 and if you delete xxx until it’s one left size is minimum at 100px |
| \column-gap: ;  \row-gap: ;  \gap: row-gap column-gap; | We set the container with 4 grid cell to 2x2 grid-template-column is 1fr 1fr grid-template-row is 100px 100px  Value is number it’s the distance between column eg 10px the distance between grid column 1 and 2 is 10px  Value is number it’s the distance between row  Eg \gap: 10px 20px; = distance of row is 10px distance of column is 20px |
| \grid-auto-flow: row;  \grid-auto-flow: column; | It will auto adjust the grid row eg if we have 4 item it will be default  It will auto adjust the grid column eg if we have 4 item it will be 4 grid column |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| Display: grid; | R1\C1 C2 C3 C4 grid will have a column and row line reference form R1\C1  R2 you can check it in inspect– element– layout– show line number  R3 |
| \grid-column-start: ;  \grid-column-end: ;  \grid-row-start: ;  \grid-row-end: ; | The value in there is number reference from R1\C1  Eg grid-column-start: 2 ; grid-column-end: 3; it start C2 end in C3  Eg grid-row-start: 1 ; grid-row-end: 3; it start R1 end in R3  Eg column start 2 end 3 row start 1 end 3 it can be R1 R3\C2 C3  - If you want to count grid lines from the right instead of the left, you can give grid-column-start and grid-column-end negative values. Eg -1 = C4  - Instead of defining a grid item based on the start and end positions of the grid lines, you can define it based on your desired column width using the span keyword. Keep in mind that span only works with positive values. |
| \grid-column: ;  \grid-row: ; | : number-start/number-end; eg grid-column:2/4; = start C2 end C4  eg grid-column:2/span 2; = start C2 end C4 (+2 from span)  eg grid-row: 1/span 3; = start R1 end R4 (+3 from span)  eg column 2/span 2 row 2/span 3 = C2 C4 R2 R5  - Typing both grid-column-start and grid-column-end every time can get tiring. Fortunately, grid-column is a shorthand property that can accept both values at once, separated by a slash. |
| \padding-inline: apx; | Normally it use in container for easy to teach because container will have the space between eg : 100px; it has space around container 100 px |
| \grid-auto-flow: row dense;  \grid-auto-flow: column dense; | It will auto adjust the grid to beautiful grid even if you do it the mess  R1\C1 C2 C3 C4 C5 all items has 15 item (item 15 will end at C3 R4 )  R2  R3  R4  R5  use grid-column-start:3; in item 2,4,7,8 it2 is C3 R1/ 3 is C3 R2 and next  when using grid-auto-flow: row dense; it will back to the first template but the number will arrange in row side (2 4 7 8 be the same)  when using grid-auto-flow: column dense; it will back to the first template but the number will arrange in column side (2 4 7 8 be the same) |
| \grid-template-areas: ‘ ‘;  Shorthand grid-template: row / column | It defined what’s our template of grid by use in container eg  grid-template-areas: ‘h h h h ‘  ‘s m m m’  ‘f f f f’  grid-template is a shorthand property that combines grid-template-rows and grid-template-columns. For example, grid-template: 50% 50% / 200px; will create a grid with two rows that are 50% each, and one column that is 200 pixels wide. |

|  |  |
| --- | --- |
| **Name** | **Meaning** |
| \grid-area: ;  Shorthand grid-row-start, grid-column-start, grid-row-end, followed by grid-column-end  Eg grid-area: 1 / 1 / 3 / 6;. | It use to link the position of item to grid template areas normally it use in item eg  \grid-area: h; in item1  \grid-area: s; in item2  \grid-area: m; in item3  \grid-area: f; in item4  Now all items will change to the grid-template-areas  - If typing out both grid-column and grid-row is too much for you, there's yet another shorthand for that. grid-area accepts four values separated by slashes: grid-row-start, grid-column-start, grid-row-end, followed by grid-column-end. One example of this would be grid-area: 1 / 1 / 3 / 6;.  - If grid items aren't explicitly placed with grid-area, grid-column, grid-row, etc., they are automatically placed according to their order in the source code. We can override this using the order property, which is one of the advantages of grid over table-based layout. By default, all grid items have an order of 0, but this can be set to any positive or negative value, similar to z-index. |
| justify vs align | Justify is row/x which left to right, align is column/y which top to bottom |
| \justify-content: start;  \justify-content: end;  \justify-content: center; | Focus in container and x-axis, it will close to the left side  Focus in container and x-axis, it will close to the right side  Focus in container and x-axis, it will close to the center |
| \align-content: start;  \align-content: end;  \align-content: center; | Focus in container and y-axis, it will close to the left side  Focus in container and y-axis, it will close to the right side  Focus in container and y-axis, it will close to the center |
| \justify-items: start;  \justify-items: end;  \justify-items: center; | Focus in container and x-axis, all items will close to the left side  Focus in container and x-axis, all items will close to the right side  Focus in container and x-axis, all items will close to the center |
| \align-content: start;  \align-content: end;  \align-content: center; | Focus in container and y-axis, all items will close to the left side  Focus in container and y-axis, all items will close to the right side  Focus in container and y-axis, all items will close to the center |
| \justify-self: start;  \justify-self: end;  \justify-self: center; | Focus in each item and x-axis, that item will close to the left side  Focus in each item and x-axis, that item will close to the right side  Focus in each item and x-axis, that item will close to the center |
| \align-self: start;  \align-self: end;  \align-self: center; | Focus in each item and y-axis, that item will close to the left side  Focus in each item and y-axis, that item will close to the right side  Focus in each item and y-axis, that item will close to the center |
| \place-items: ; | It use in grid, can shortcut from align-items + justify-items  Eg place-items: start center; (align-items property value is 'start' justify-items property value is 'center')  normal legacy Default. The element's default place-items value. The default value for align-items is 'normal', and the default value for justify-items is 'legacy'. Baseline Items are positioned at the baseline of the container  center Align items to the center of the grid cell  end Align items at the end of the grid cell  start Align items at the start of the grid cell  stretch grid items to fill the grid container if the grid item sizes are not set. |

|  |  |  |
| --- | --- | --- |
| **Name-Knowledge** | **Meaning** | |
| h1,h2,p { } | h1 { } | h2 { } | p { } | |
| A { } | | Selects all elements of type A. Type refers to the type of tag  div { } selects all div elements. |
| #id { } | | Selects the element with a specific id  #cool { } selects any element with id="cool" |
| A B { } | | Selects all B inside of A. B is called a descendant because it is inside of another element.  p strong { } selects all strong elements that are inside of any p |
| #id A { } | | Selects all element A inside of id=”id”  #cool span { } selects all span elements that are inside of elements with id="cool" |
| .classname { } | | The class selector selects all elements with that class attribute. Elements can only have one ID, but many classes.  .neato selects all elements with class="neato" |
| A.className { } | | selects all A elements that have class="className"  ul.important { } selects all ul elements that have class="important"  #big.wide { } selects all elements with id="big" that also have class="wide" |
| A, B { } | | this selects all A and B elements  p, .fun { } selects all p elements as well as all elements with class="fun"  a, p, div { } selects all a, p and div elements |
| \* { } | | You can select all elements with the universal selector!  p \* { } selects any element inside all p elements.  ul.fancy \* { } selects every element inside all ul class="fancy" elements. |
| A + B { } | | This selects all B elements that directly follow A. Elements that follow one another are called siblings. They're on the same level, or depth. In the HTML markup for this level, elements that have the same indentation are siblings.  p + .intro { } selects every element with class="intro" that directly follows a p  div + a { }selects every a element that directly follows a div |
| A ~ B { } | | You can select all siblings of an element that follow it. This is like the Adjacent Selector (A + B) except it gets all of the following elements instead of one.  A ~ B { } selects all B that follow a A |
| A > B { } | | You can select elements that are direct children of other elements. A child element is any element that is nested directly in another element. Elements that are nested deeper than that are called descendant elements.  A > B { } selects all B that are a direct children A |
| :first-child | | selects all first child elements. A child element is any element that is directly nested in another element (first line of that element)  p:first-child { } selects all first child p elements.  div p:first-child { } selects all first child p elements that are in a div |
| :only-child | | Selects that element has only 1 child span:only-child { } selects the span elements that are the only child of some other element.  ul li:only-child { } selects the only li element that are in a ul. |
| :last-child | | selects all last-child elements.  span:last-child { } selects all last-child span elements.  ul li:last-child { }selects the last li element that are in a ul. |
| :nth-child(A) | | Selects the nth (Ex: 1st, 3rd, 12th etc.) child element in another element.  :nth-child(8) { } selects every element that is the 8th child of another element.  div p:nth-child(2) { } selects the second p in every div |

|  |  |
| --- | --- |
| **Name-Knowledge** | **Meaning** |
| :nth-last-child(A) | Selects the children from the bottom of the parent. This is like nth-child, but counting from the back!  :nth-last-child(2) { } selects all second-to-last child elements. |
| :first-of-type | Selects the first element of that type within another element. span:first-of-type { } selects the first span in any element. |
| :nth-of-type(A) | Selects a specific element based on its type and order in another element - or even or odd instances of that element.  div:nth-of-type(2) { } selects the second instance of a div.  .example:nth-of-type(odd) { } selects all odd instances of a the example class. |
| :nth-last-of-type(A) | This is like nth-of-type, but counting from the back |
| :nth-of-type(An+B)  :nth-child(An+B) | The nth-of-type formula selects every nth element, starting the count at a specific instance of that element.  span:nth-of-type(6n+2) { } selects every 6th instance of a span, starting from (and including) the second instance. |
| :nth-child(A) vs :nth-of-type(A) | Eg div:nth-child(1) vs div:nth-of-type(1) and parent= <div></div> and child is <p> <div> <div>  nth-child will link to <p> in <div> // nth-of-type will link to first <div> in <dib> |
| :only-of-type | Selects the only element of its type within another element. (only 1 p is specific if 2 p in there is did not)  p span:only-of-type { } selects a span within any p if it is the only span in there. |
| :last-of-type | Selects each last element of that type within another element.  div:last-of-type { } selects the last div in every element.  p span:last-of-type { } selects the last span in every p. |
| :empty | Selects elements that don't have any other elements inside of them.  div:empty { } selects all empty div elements. |
| :not(X) | You can use this to select all elements that do not match selector "X".  :not(#fancy) { }selects all elements that do not have id="fancy".  div:not(:first-child) { } selects every div that is not a first child.  :not(.big, .medium) { } selects all elements that do not have class="big" or class="medium". |
| [attribute]  A[attribute]  [attribute="value"]  [attribute^="value"]  [attribute$="value"]  [attribute\*="value"] | Attributes appear inside the opening tag of an element, like this: span attribute="value". An attribute does not always have a value, it can be blank!  [value] { } selects all elements that have a value="anything" attribute.  [type] selects all elements that have a type="anything". attribute  a[href] { } selects all a elements that have a href="anything" attribute.  input[disabled] { } selects all input elements with the disabled attribute  input[type="checkbox"] { } selects all checkbox input elements.  .toy[category^="Swim"] { } selects elements with class toy and either category="Swimwear or category="Swimming".  img[src$=".jpg"] { } selects all images display a .jpg image.  img[src\*="/thumbnails/"] { } selects all image elements that show images from the "thumbnails" folder.  [class\*="heading"] { } selects all elements with "heading" in their class, like class="main-heading" and class="sub-heading" |

|  |  |
| --- | --- |
| **Name-Knowledge** | **Meaning** |
| : | It appears when element has some state |
| :hover | When moving the mouse to their element it appear (No click) |
| :active | Use this to add special style to an active element (click on web it changed) |
| :link | Use this to add special style when unvisited link (It always show) |
| :focus | Use this to add special style to an element while the element has focus (use in from text) |
| :visited | Use this to add special style when visited link (after click on web it changed) |
| :: | It appears when element has some status or some condition (has so many) |
| ::first-letter | Focus in first letter in any element |
| ::first-line | Focus in first line in any element |
| ::selection | Focus when cover the mouse (click and move) |
| ::marker | Use with tag ul, ol is change the style of list eg 1 2 3 |
| ::before | Create pseudo element that is the first child of selected element |
| ::after | Create pseudo element that is the last child of selected element |