|  |  |  |  |
| --- | --- | --- | --- |
| <style>  Li{  padding:0px;  }  </style> | | Agregar estilo css desde html. | |
| <a hrf=’#’ style=’padding:0px’ > | | Agregar estilo css a una etiqueta directamente desde html | |
| **SELECTORES** | |  | |
| \*{}  Li{}  .nombre{}  #nombre{}  [href]{}  [href=’www.google.com’]{}  [href^=’www’]{}  [href$=’org’]{}  [href\*=’af’]  H2[class]{}  H2, a, p{} | | Universal  Etiqueta  Id  Class  Atributo  Atributo+ dirección  Atributo + inicio con ‘www’  Atributo+ finalización con ‘org’  Atributo + que contiene en cualquier parte ‘af’  Etiqueta h2 con atributo class.  3 etiquetas con la | |
| **Descendencia Padre e hijo**  div p{}  nav>.contenedor{}  nav.contenedor{}  nav.contenedor.titulo{} | | A todos los hijos directos o indirectos  Todos los hijos directos (class=’nav’) de nav  Todos los nav con class=’contenedor’  Todos los nav con class=’contenedor’ y ‘ titulo’ | |
| **Selectores hermanos**  h2+p{}  h2~p{} | | H2 y P hermano adyacente (al lado). **Se le aplica el css a P**  H2 y P hermanos adyacentes(todos). **Se le aplica css a todos los p que están consecutivos a h2** | |
| **Pseudo clases**  **.**enlace:link{}  .enlace:visited{}  .enlace:hover{}  .enlace:active{}  <a href=’**#destino’**>i</a>  <p id=’destino>asdf</p>  #destino:target{}  **para imputs**  Input:focus{}  .contenedor:focus-within{}  .opcion:checked + label{}  :root{}  p:not(.title){}  p:not(.title):not(:hover){}  div:only-of-type:{}  .box:firt-of-type:{}  .box:last-of-type:{}  .box:nth-of-type(2){}  .box:nth-of-type(2n){}  box:nth-of-type(2n+1){}  .box:nth-last-of-type(2n){}  .box:only-child{}  p:fist-child{}  div:last-child{}  section :nth-child(3) -----nota, si es con contenedor va espacio ya que tiene hijos dentro y hay que especificar que se trata de padre e hijos  section :nth-last-child(1){} | | Selecciona links que no han sido visitadas  Selecciona links que has sido visitadas  Acción cuando se pasa encima del elemento  Cuando clic abajo  Definimos el disparador  El objetivo  Acción de cambio al ser disparado  Cuando se tiene el focus  Cuando uno de sus hijos tiene el focus modifica al contenedor  Si el checkbox es tildado, modifica a su hermano adyacente.  Selecciona a la raíz(html)(aca van las variables)  Selección a todos los p que no tienen clase ‘title’  2 condiciones -no tienen la clase title y no están con hover  Selecciona todos los únicos ELEMENTOS en se section  Que no tiene hermanos de su mismo tipo (agrega 10 especificidad)  Toma al solo al 1º de su **tipo** dentro de su **contenedor**  Toma solo al último de su tipo dentro de su contenedor.  Igualmente toma al segundo.  Selecciona si son pares  Selecciona si son impares  El mismo proceso, pero comienza desde abajo a arriba  Selecciona si son hijos únicos dentro de su contenedor. Ósea que no te tanga hermanos.  Selecciona Solo si son el primero dentro de su contenedor.  Selecciona solo si son el ultimo dentro de su contenedor.  Toma al 3er hijo dentro de ese contenedor sin importar el tipo.  Dentro del contenedor section toma al primero pero comenzando desde abajo. (tomaría al ultimo en este caso) | |
| **PSEUDO ELEMENTOS**  div::selection{}  ::selection{  Color:black;  Background-color:blue;  }  Div::first-line{}  Dic::first-letter  Imput::placeholder{}  Agregar texto antes y después de un párrafo  p::before{  content:’hola ’;  }  P::after{  Content:’adios ’;  } | | Dentro del div (asdfjasdfasdfasdfasdfasdf)  A todo el documento al seleccionar texto  Toma a la primera línea de texto del contenedor  Toma a la primer letra el contenedor  Podemos modificar el texto del placeholder de esta manera  <p>jony</p>  hola jony adios | |
| PADDING | | Para evitar que los hijos modifiquen el tamaño establecido  **Box-sizing=border-box;** | |
| Padding:10px;  Padding 10px 10px  Padding 10px 5px 10px 5px  Padding-top:10px;  Padding-bottom:10px;  Padding-left:10px;  Padding-rigth:10px; | | A todos lados  Arriba, abajo – izquierda derecha  Arriba – derecha – abajo - izquierda | |
| **Margin** | | Para evitar que los hijos modifiquen el tamaño establecido  **Box-sizing=border-box;** | |
| margin: auto;  margin:10px;  margin: 10px 10px  margin: 10px 5px 10px 5px  margin-top:10px;  margin-bottom:10px;  margin-left:10px;  margin-rigth:10px;  margin-bottom:-10px;  margin-top:auto;  margin-bottom:auto;  margin-left: auto;  margin-rigth: auto; | | Se centra el contenido al contenedor en horizontal (si es **bloque y tiene un witdh**)  A todos lados  Arriba, abajo – izquierda derecha  Arriba – derecha – abajo – izquierda  Efecto de pisado  Se apega hacia arriba dentro del contenedor  ..  ..  .. | |
| Overflow:hidden;  Box-sizing=border-box;  Box-sizing=content-box; | | Oculta todo lo que desborda del **contenedor**  Evitar que el contenedor se expanda.  default. Habilitado para poder expandirse. | |
|  | |  | |
| **Display**  Display: block;  Display: inline;  Display inline-block;  Display: none; | | En html existen en bloque (ocupan en 100% de la pantalla)  En línea (ocupa solo su contenido)  Define que ocupa todo el ancho de la pantalla.  Ocupa solo su contenido **estrictamente. No deja modificar**  Combina el estilo de línea pero se puede modificar como bq  El elemento desaparece (visual y espacio). | |
| **Especificidad**  Body{}  .contenedor  [href]  :only-of-type  #contenedor  !important  (bg-color: blue !important;) | | Determina la importancia (sobre la cascada). Es acumulativo.  1  10  10  10  100  Va dentro y determina que tiene más especificidad que cualquiera | |
| **Herencia**  Color: **inherit**;  Font-family**: inherit;**  font-family: **initial;** | | Heredamos las propiedades del contenedor padre.  Algunos valores se heredan automáticamente. Para forzar la herencia se usa **inherit**  Volver al estado por defecto que nos da css | |
| **Border** | |  | |
| Boder: 10px solid blue;  Border-width:10px;  Border-style: solid;  Border-color: blue;  border-width:10px 10px 20px 0px;  boder-style: solid double dotted groove ;  border-color: blue red green grey;  border-top:50px solid green;  border-right: 40px double blue;  border-bottom:30px groove red;  border-left: 20px dotted yellow;  border-top-width: ;  border-top-style: ;  border-top-color: ;  border-right-width: ;  border-right-style: ;  border-right-color: ;  border-bottom-width: ;  border-bottom-style: ;  border-bottom-color: ;  border-left-width: ;  border-left-style: ;  border-left-color: ;  outline: 3px solid black; | | **Tamaño** estilo **color**  Tamaño  Estilo  Color  Top right **bottom** left  Top right **bottom** left  Top right **bottom** left  **Tamaño** estilo **color**  **Tamaño** estilo **color**  **Tamaño** estilo **color**  **Tamaño** estilo **color**  Poner detalladamente las propiedades…  igual que el border pero por afuera, no modifica el tamaño del contenedor. | |
| Boder-radius  Border-radius:10px;  Border-radius:50%;  Border-radius: 10px 10px 10px 10px; | | Redondeo de bordes  Top rigth bottom left | |
| **Box-shadow**  Box-shadow: 10px 10px 10px 0px black;  Box-shadow: 10px 10px 10px 0px blue, 10px 10px 10px 0px red;  **Sombreado en .png (solo la imagen)**  Filter:drop-shadow(10px 10px 10px black); | | * Desplazamiento en x * Desplazamiento en y * Nivel de difuminado * Expandido o contraer el tamaño de la sombra * Color de la sombra   se pueden poner varias sombras.   * Desplazamiento en x * Desplazamiento en y * Tamaño de la sombra * Color de la sombra | |
| **Background** | | …/img/miimagen.jpg **PARA IR CARPETA ATRÁS**. | |
| Background-color: red  Background-image: url(img/miiagen.jpg);  Background-image: url(img/miiagen.jpg), url(img/imagen2.jpg);  Background-repeat:no-repeat;  Background-repeat: repeat-x  Background-repeat:repeat-y;  Background-position:20px 30px;  Background-position: center;  Background-position:top rigth;  Background-position:top left;  Background-position:bottom left;  Background-position:top center;  Background-position:top 100px rigth;  Blackground-size: 50px 50px;  Blackground-size: contain;  Blackground-size: cover;  background-attachment: fixed;  background-image:lineal-gradient(to top left ,black 20%, green 100%)  background-image:lineal-gradient(to top left ,black, green), url(..) | | transparent (sin color)  Dos imágenes de fondo  Las images chicas vienen con repeat. Asi lo cancelamos.  Repite en x  Repite en y  Posición de la imagen repecto a (x y)  Centrado en x y  Alineado  ..  ..  ..  Alineado además una valor extra(+100px den top)  Tamaño en width height  No corta y ni deforma  Se acomoda en todo el container **corta la imagen**  El fondo queda fijo en el fondo (**no se mueve con scroll**)  (sentido, color1 tañamo, color2 tamaño).  Poner dos imágenes de fondo. | |
| **TEXTOS** | | **En bloque¿?** | |
| Text-align: center;  Line-height:  Text-align: justify;  Text-align: left;  Letter-spacing:10px;  Word-spacing:10px  Line-hight:4px;  Text-shadow: 5px 5px 0px back  Text-transform: lowercase;  Text-transform: uppercase;  Text-transform: capitalize;  Font-variant: small-caps;  Text decoration-line: underline;  Text decoration-line: line-through;  Text decoration-line: overline;  Text decoration-style: dotted;  Text decoration-color: red;  Font-weight: bold;  Font-style: italic;  Font-size: 16px;  Font-family: times new roman, herlvetica, cursiva; | | Centrado en x  Alinea en y (**solo una línea de texto y tiene que tener el mismo higth que el contenedor**)  Justificado (se pone espacio entre letras para ocupar x)  Right  Especifica el espacio entre letra y letra.  Especifica el espacio entre palabra y palabra.  Especifica el espacio entre renglón y renglon  X y expandido color  Texto en minúscula  Texto en mayúscula  Primera letra de palabra en mayúscula resto en minúscula  Texto en mayúsculas pequeñas (tamaño mini)  Subrayado  Tachado  Linea superior  Wave(ondulado)-doble-solid-none  Solo a las decoraciones. No las letras  Negrita (también numeros 5px );  Oblique (más inclinado que italic)  Tamaño de texto.  Si no esta la letra en la pc, pasa a la siguiente definida. | |
| **LIMITADORES**  Max-width  Max-height  Min-width  Min-height | | Definir el tamaño máximo y mínimo de expansión o retracción | |
| **Unidades de medida**  1px  1rem  1em  1%  **Viewport (vh, vw)**  100vh  100vw | | 1 px de pantalla (son fijos a la pantalla)  Equivale al valor definido por el navegador(escala)  Equivale al valor definido por el contenedor padre  Porcentaje de tamaño de ventana de navegador  Como el porcentaje, pero también para **height** | |
| **Listas** | |  | |
| List-style-image:url();  List-style-type:  List-style-position:; | | Poner un png ico personalizado  None: sin ningún icono  Disc: punto negro  Circle:circulo blanco  Decimal: numero  Square: cudrado  Lower-roman:  Upper-roman:  Inside: dentro el contenedor li.  Outside:fuera del contenedor li. | |
| **Control de imágenes (img)** | |  | |
| Object-fit: | | None: no se escala  Countain: escala solo en x (no recorta)  Cover: escala en x y (recorta)  Fill: ocupa todo el countainer img sin recortar | |
| **Positions**  Position:static;  Position:relative;  Position:absolute;  Position:fixed; | | Es el que tiene html por defecto  Desbloquea nevas propiedades: top: left: rigth: bottom:  Se mueve como ventaja flotante pero mantiene su espacio  Desbloque las mismas que la anterior pero este pierde su espacio en la pantalla y flota.  **Se mueve dentro de contenedores posicionados( no estático)**  Igual que absolute pero se mueve sobre el viewport  Se queda fijo en la pantalla | |
| **Centrar con relative:**  Position: relative;  Top:50%;  Left:50%;  Transform:traslate(-50%,-50%)  **centrar con flexbox:**  contenedor…  display:flex;  hijo container…  margin:auto;  **centrar con margin top**  margin-top:calc(50vh – 50px) | | Centrar un div dentro de su contenedor  (acordar poner el padding-top:0.1px por el colapso)  (tamaño del contenedor – la mitad del tamaño del objeto) | |
| Elementos posicionados  z.index:1; | | Determina quien va por encima de quien (ir de 10 en 10)  Mientras mas grande se superpone sobre los menores.  solamente para elementos posicionados: relative, absolute y fixed. | |
| \*{  scroll-behavior:smooth;  } | | Efecto de transición al scroll. Al llamar posición con tag | |
| **Variables en css**  :root{  --mivar:20px;  }  .1{  margin:var(--mivar);  }  **FALLBACKS**  Margin:var(--mivar, 10px)  Margin:var(--mivar, var(--mivar2)); | | Definimos la variable en root (**global**)  **Scope:** depende donde la declaramos es su universo.  Asignamos la variable en una propiedad o mas  En caso que la variable este rota o no exista, toma la otra  fallback con otra variable | |
| **Formularios propiedades**  outline=none; | | **Recordar que los inputs no heredan las propiedades**  Quitar las líneas al imput al tener focus | |
|  | |  | |
| **RESPONSIVE DESIGN** | | | |
| Content=’  Initial-scale  Manimun-scale  Máximum-scale  User-scale  ‘ | | DEFINE SI SE VA PUEDE HACER ZOOM (PELLISCAR CELULAR)  Zoom inicial de la página (1 por default)  Mínimo de zoom  Máximo de zoom  Si el usuario puede o no hacer zoom. | |
| **MEDIA QUERY @media<media-type> and(*condicion*){}**  @media print{  .div{  cambios…  }  }  @media screen and (max-width:320px){  .title{}  }  @media screen and(orientation:landscape){}  @media screen and(orientation: portrait){}  @media screen and(orientation:landscape) and (max-width:200px){}  @media screen and(orientation:landscape) , (max-width:200px){} | | Cambios que hará cuando mandemos a imprimir(literal)  mientras cumpla la condición (el máximo es 320)  nota: guiarse por las herramientas de pixeles del navegador  ojo con las epecificidad  nota: poner al final por la cascada  Lo que se va a modificar    and  or | |
| **FLEX-BOX**  Display:flex;  Display: inline-flex;  Flex-direction: row;  Flex-direction: row-reverse;  Flex-direction: column;  Flex-direction: column-reverse;  flex-wrap: nowrap;  Flex-wrap: wrap;  Flex-wrap: wrap-reverse  Flex-flow: column wrap;  **ALINEACIONES CUANDO DESBORDAMOS Y HAY ESPACIO**  Align-content:….;  align-content  **ALINEACION ELEMENTOS (solo si hay espacio)**  Justify-content:flex-start;  Justify-content:flex-end;  Justify-content:center;  Justify-content:space-between;  Justify-content:space-around  Justify-content:space-evenly;  **ALINEACIONES CUANDO NO DESBORDAMOS**  Align-items:…;  align-items  Gap:10px;  -------propiedad para hijos del contenedor--------  **ESPACIO DISPONIBLE**  Flex-grow:0;  Flex-grow:1;  Flex-grow: n;  **ESPACIO FALTANTE**  Flex-shrink:0;  Flex-shrink:1;  Flex-shrink:n;  **DARLE UN TAMAÑO INCIAL**  Flex-basis  Flex-basis: auto;  Flex-basis:0;  Flex-basis:10px;  Flex: 1 1 0;  Align-self:  align-self  ------------------IMAGENES------------------  Buena practica: no usar img sino <figure> tratarlo como hijo del contenedor normalmente. Poner una back-ground url o  img dentro de figure | | Caja flexible (su contenido nunca superan al container)  Contenedor  Sus hijos:   * display in line * no rebasan al contenedor por mas grandes q sean * toman porcentajes y se distribuyen * cambian el tamaño junto con el padre   combina el posicionamiento y comportamiento de inline  **tipo y posición:**  modo fila- posición izquierda arriba  modo fila- posición derecha arriba  modo columna- posición arriba derecha  Modo columna – posición abajo derecha  Default: sin acción de desborde y se adaptan al padre  Accion cuando sus hijos rebasan el tamaño del padre  Acción en reversa  Abreviatura.  A diferencia de justyfy esta determina el como de justificado que van a tener los hijos que desbordaron el tamaño por flex-wrap.  Nota: stretch no funciona si los elementos tienen un hight definido.    default: flex-start  Gap lo que hace es poner un margen entre los hijos ya sea esten en row o column. Esto evita la suma de márgenes y paddings  Default  si se le da solo a uno este ocupa todo el espacio sobrante del contenedor flex  si se le ponemos a un grupo estos cupan todo el espacio sobrante pero además se igualan sus tamaños.  Podemos priorizar a un item es especial para darle una mayor fracción de espacio ocupado  No cambiara su tamaño dependiendo a su contenedor  Default  Determina que porcentaje se va a encoger un item al faltar espacio en el contenedor (**más grande el nro. más pequeño se vuelve el item al faltar espacio**)  Predomina sobre el hight y width(no del max-width…)  Depende de su **flex-direction (toma la del width o height)**  Defaut . Se adapta al tamaño de su contenido(max-content)  No se adapta. En caso de tener un **flex-grow:1 setear :0**  Reemplaza a height o width dependiendo…  Grow shrink basis  Al igual que **align-items**  pero solo para un solo un item en específico tiene las mismas propiedades | |
| **Peso de los elementos hijos dentro den un flexbox** | | | |
| order:0; (default)  order:1; | | **A mayor peso cae más al fondo.**  Cambiamos el orden de los elementos sin tener que tocar html combinado con **@media** podemos darles prioridad a los elementos en responsive design. | |
| Calculos en css  Height: calc(100vh-var(--height\_nav)); 100vh-(70px) | | Podemos hacer operaciones matematicas dentro de calc y no importa que sean del mismo tipo | |
| Ciclos en css  repeat(2, 20px 30px); | | Escribe 20xp 30px 20px 30px | |
| **CSS GRID** | | | |
| Conceptos básicos:  grid container  grid item  grid celd  grid lines  grid truck  grid area | | Conteiner de grid  elementos dentro de grid  celdas formadas por lineas  lineas verticales y orizontales forman celdas y columnas y filas  espacio entre lineas→ osea filas o columnas  cojunto de celdas que **forman un cuadrado** | |
| **CONTENEDOR**  display: grid;  grid-auto-flow:row;  grid-auto-flow:column;  grid-auto-column: 20px  grid-auto-flow: row dense;  grid-template-colums: 25% 25% 25%;  24px auto;  grid-template-row: repeat(4,1fr);  grid-template-column: min-content max-content;  grid-template-column:fit-content(200px);  grid-template-column: minmax(10px, 20px) minmax(…)  grid-template-columns: repeat(autofill, minmax(100px,1fr));  grid-template-column: repeat(auto-fit, 10%)  ESPACIADO  row-gap:10px;  column-gap:10px;  gap:10px 20px;  gap:20px;  AREAS  grid-template-area:  “jony jony a a”  “jony jony . .”  “jony jony gerardo alanoca”; | | **Inspeccionar elementos→layout→check grip (lineas)**  Crea la tabla con los elementos como referencia para filas o columnas  Por defecto. Crea filas implícitamente  Crea columnas implícitamente, crea automáticamente  le da un valor por defecto al elemento que desborda    intenta llenar los espacios vacíos de manera ordenada  crea 3 columnas ocupando el 75% del contenedor, el resto lo deja vacío, crea una nueva fila con los elementos sobrantes y ocupan el mismo tamaño  auto toma el espacio restante  resultado crea 4 filas con ¼ cada una o 25% c/u.s  crea dos celdas(columnas) la primera intenta no expandirse y manda lineas abajo en los hijos  la segunda: no desborda sino que crece junto a su contenido  al principio es igual al max-content luego al llegar a 200px actua como min-content  condiciona el ancho minimo y maximo de la columna  **responsivo:**autofill crea automaticamente las columnas que sean necesarios y entren.  Combinado con maxmin le decimos que los hijos no pueden tener un tamaño menor a 200px y tiene que tener si o si una columna.  Solo crea las lineas necesarias osea a diferencia de auto-fill no ocupara todo el espacio del contenedor, solamente los que necesitan los hijos  espaciado entre filas  espaciado entre columnas  short filas columnas  short a ambos  le da nombres a las celdas, las que tienen el mismo nombre son agrupadas(no pueden separarse). Sirve para hacer responsive designe.  Siempre deben formar un cuadrado o rectángulo, completar con un “.” para rellenar espacios vacíos | |
| **HIJOS**  POSICIONAMIENTO  grid-column-start:1;  grid-column-end:3;  grid-row-start:1;  grid-row-end:3;  grid-row:1 / 3;  grid-row:span 3;  grid-row: 1/-1  grid-area:1/1/4/4  AREAS  grid-area: nombreArea; | | Comienza en la linea 1(columna)  termina en la linea 3  Comienza en la linea 1(fila)  Termina en la linea 3  short  toma 2 espacios mas desde su posición inicial  es lo mismo pero le damos el conteo empieza del final  row-start / column-start /row-end / column-end  ocupara el tamaño total de las celdas con ese nombre  previamente definidos | |
| **ALINEACIONES(PADRE)**  justify-items:..  stretch  center  start  end | | stretch | Center  .  . |
| start  .  . | end  .  . |
| Align-items:..  stretch  center  start  end | | strecth | Center . |
| Start . | End . |
| Place-items: end end; | | **Short** align-items / justify-items | |
| **ALINEACIONES CUANDO DESBORDAMOS Y HAY ESPACIO(PADRE)**  Align-content:….;  align-content  **ALINEACION ELEMENTOS (solo si hay espacio)**  Justify-content:flex-start;  Justify-content:flex-end;  Justify-content:center;  Justify-content:space-between;  Justify-content:space-around  Justify-content:space-evenly;  place-content:center center; | | Alineacion de vertical  Nota: stretch no funciona si los elementos tienen un hight definido.  **NOTA IMPORTANTE:** flex-start va solamente start  de la misma forma flex-end va end    **short** align-content / justify-content | |
| **ORDER(hijos)**  order:0;  order:1; | | El peso que tienen los elementos entre si  por defecto  este tendra mas peso osea que lo acomoda mas al final posible | |
| **Z-INDEX(HIJOS)**  z-index:1; | | Le da importancia cuando dos elementos estan superpuestos uno arriba de otro.  Este tendra prioridad de visibilidad sobre uno con 0 | |
| **ALINEACION INTEPENDIENTE (HIJOS)**  Align-self  justify-self  place-self:center end; | | Igual que el align-items pero solamente al elementos  iguala que el justify-items pero solamente al elemento  **short** align-self / justify-self | |
| **CHIP PATH** | | | |
| Fundamentos:  path  puntos de referencia: | | Parte visible | |
| Clip-path: circle(50% at 50% 50%) | | - El radio: 50% del largo del contenedor  -posición en x  -posición en y  --- tambien se puede usar center top right | |
| Clip-path: ellipse(25% 40% at center); | | -define el ancho del radio **(osea 50% ocupa todo el container)**  -define el alto del radio  -define la posición | |
| Clip-path:inset(0% 0% 0% 0%) | | Complicado de explicar pero es sencillo  **top -rigth - bottom – left**  **Porcentaje de saparación de su linea**  actualmente ocupan el 100% del contenedor | |
| Combinacion para redondear  Clip-path:inset(0% 0% 0% 0% round ..)  10px  10px 20px  10px 20px 30px 40px | | 10px a todas las esquinas  10px a superior izquierda e inferior derecha ...etc  10px a la esquina superior izquierda continua en sentido del reloj | |
| Clip-path-polygon(50% 0%, 100% 100% , 0% 100%) | | Se pueden poner lo puntos deseados, siempre y cuando sean mayor a 2.  función: 50% en x 0% en y (**mirar fundamentos**) | |
| **TRANSFORM** | | | |
| Fundamentos:  cordenadas |  | | |
| Transform:none  transform: traslateX(200px)  transform:traslateY(100%)  transform:traslate(20px, 20px) | Deafault  se mueve en x  se mueve en y  se mueve en x y | | |
| Transform:rotate()  **Nota: el rotate también cambia las cordenadas, sobre todo si lo pones al principio cuando combinas varios transform ej: …: rotate(35deg) traslate(25px);** | Grados 0-360deg  radians 0 - 2rad  gradians 0 - 40grad  turn 0 -1turn | | |
| Transform: scaleX(…..)  transform:scaleY(……)  transform:scale(1 ,2) | En x → .2 2 -1  en y  en x y | | |
| **NOTA: si el padre tiene un transform distinto de none y el hijo tiene un position fixed si aplica como un position del padre** |  | | |
| **transform-origin: center right;**  transform: rotate(45deg);  transition: transform .5s  **(default center center)** | Definimos el punto de la transformacion | | |
| **TRANSITOIN** | | | |
| Generalidades:  no se hacen aplican a elementos inline  no todas las propiedades se pueden aplicar transición  debe haber un disparador  no romple el flujo osea no empuja a otros elementos |  | | |
| .container:hover .element | Se aplica la transicion animacion al elemento cuando se hace hover en el padre. Esto es para evitar errores. | | |
| Transition-property:none;  Transition-property:all;  Transition-property:color, transform; | Default  a todas las propiedades (no recomendable)  definimos a que propiedades aplicar | | |
| Transition-timing-function: linear  …: ease  ..: ease-in  …: ease-out;  …: ease-in-out  cubic-bezier(0,1.67,1,-0.43)  transition-timing-function: steps(6, end); | Lineal, no varia la velocidad  mas rapido en el medio  mas rapido al final  mas rapido al inicio  comienza rapido y finaliza rapido  se crea un estilo pesonalizado  crea en pasos salteados (bueno para gifs) | | |
| Transition-duration: .5s, 2s;  transition-delay:0s, 2s; | La primer transición dura medio segundo…  la primer comienza al instante la segunda espera 2 segundos antes de comenzar.  **Nota: estos dependen del transition property** | | |
| **SHORT**  transition: margin .5s ease 0s , transform 2s 2s | Property / duration / timing / delay | | |
| NOTA:  Se puden usar las paseudo clases de hover, active, target para usarlosn con transition  buscar mas en la documentacion | Una forma de deshacer el target creando otro a href=”#” | | |
| **ANIMACIONES** | | | |
| .objetoAAnimar{  animation: nombreAnimacion 2s inifinite, … ;  }  @keyframes nombreAnimacion{  0%{…}  25%{..}  100%{...}  } | 2s: tiempo de la animacion (pueden haber mas)  infinite: opcional para que entre en bucle. **Ya que caso contrario solamente se realizaría una vez al cargar la pagina.**  Proceso se animación  inicia su **estado inicial** en el 0%  llega a su **estado final** en 100%  **Nota: si no hay 0% o 100% toma el que tiene como default en .objetoAAnimar{}** | | |
| POR PARTES  animation-name: animacion1, animacion2  animation-duration: 2s, 2s;  animation-timing-function: …;  animation-iteration-count  animation-direction:  animation-fill-mode:..;  animation-delay:  animation-play-state: | Default: none  ease / lineal / ease-in /ease-out /ease-in-out  1: default (una vez)  1.5 una vez y se para en la mitad  infinite: se repite una y otra vez  normal: default  reverse: comienza en 100% y termina en 0%  alternate: va de 0% a 100% a 0% a 100% etc.  sobre todo cuando tenemos **infinite**  alternate-reverse: lo mismo pero en reversa  usar **cuando no es infinite**  none: al terminar la animación vuelve a su estado original(de golpe si no lo configuramos bien)  fordwards: las propiedades quedan una vez que llegan al 100%  0s default  2s : espera 2 segundos antes de comenzar  running: default  paused: pausa la animacion ( combinar con hover) | | |
|  |  | | |