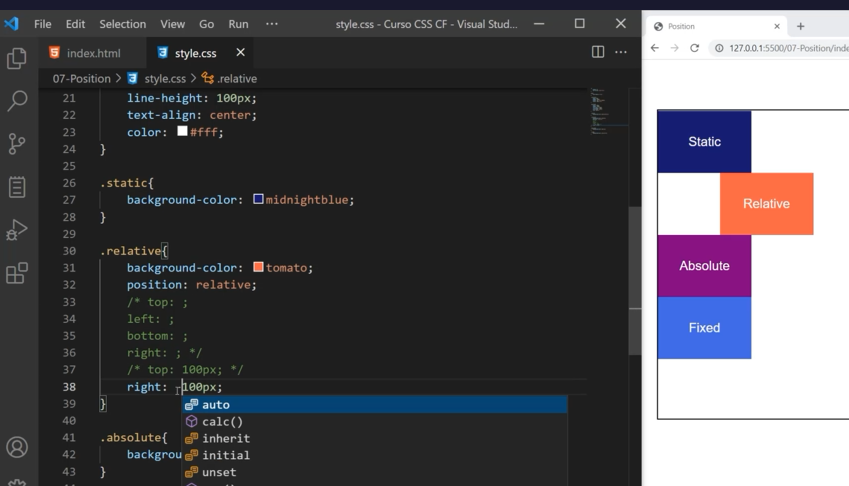
**Position**

Modificar el flujo de un elemento en el navegador, los elementos siempre van de arriba – abajo, de izq a derecha.

*Por defecto está en Static ->* position: static;

position: relative; desbloquea top-bottom-left-right, permiten desplazar al elemento, se sobrepone a los demás elementos. Siempre guarda el espacio del elemento en el flujo del navegador.

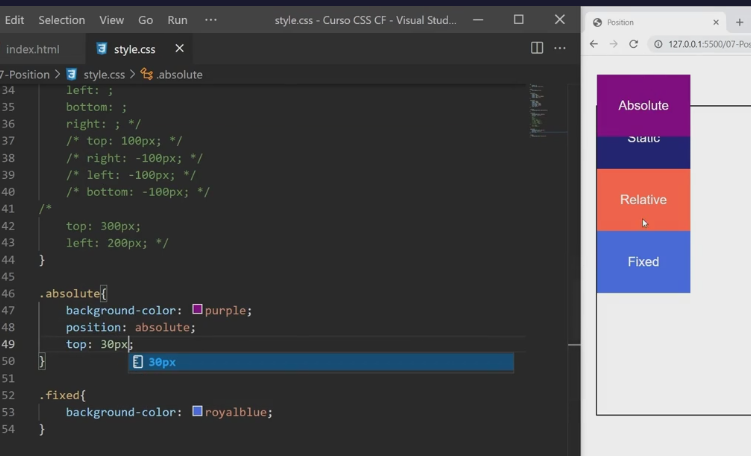


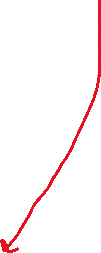


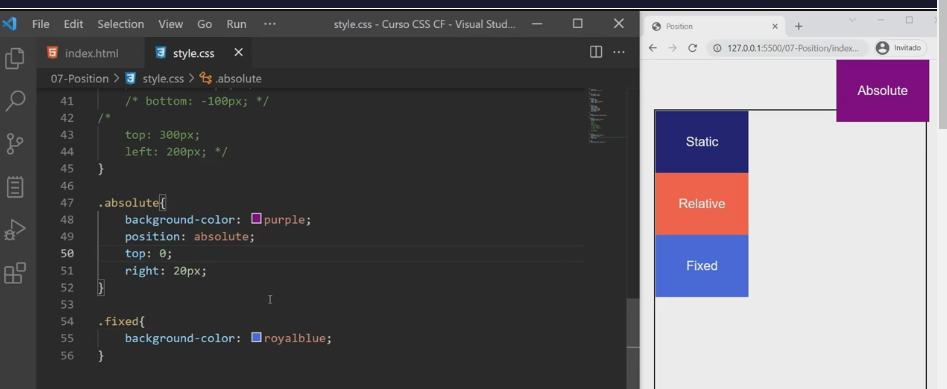


position: absolute; desbloquea top-bottom-left-right, el elemento se remueve del flujo natural del navegador por tanto su espacio no es reservado. Se mueve en base al ancestro contenedor más cercano con position diferente de static, si ninguno lo tiene, se liga al body (view port), al hacer scroll el view port se cambia y el elemento se queda en el primero.

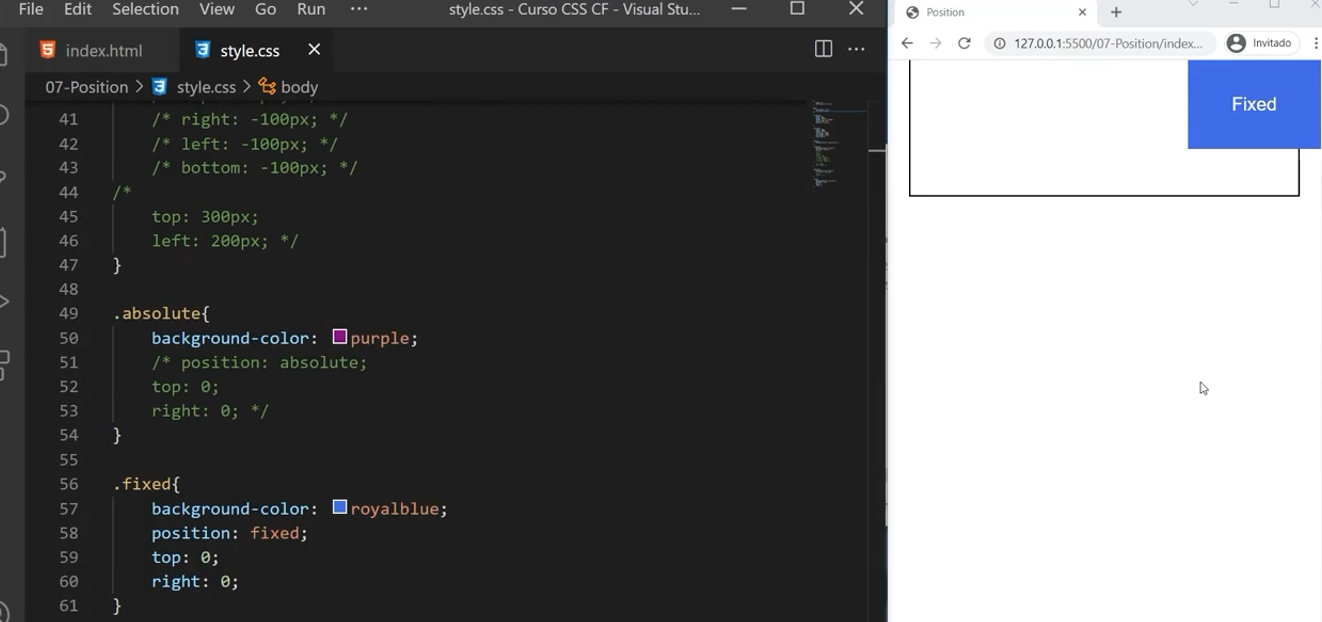




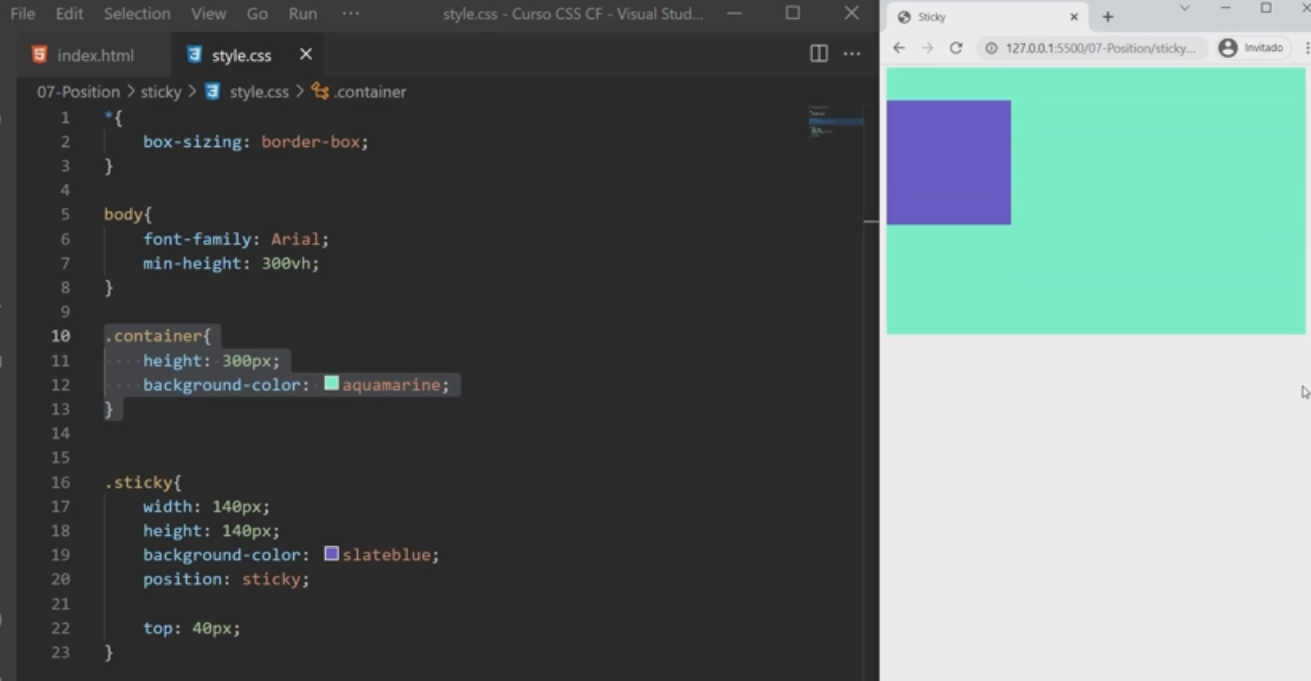


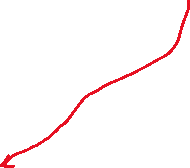
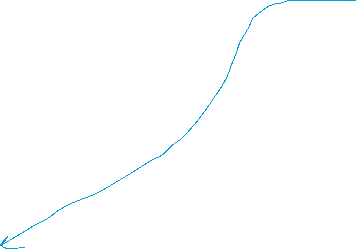


position: fixed; pierde el flujo en el navegador y se peque en el navegador al hacer scroll, siempre se posiciona al view port



position: sticky; mezcla entre relative y fixed, se trata como relative hasta un punto donde se pueda pegar como fixed, siempre se define con top y bottom y se pega al view port o su padre más cercano (para que funcione el padre debe tener un height definido)

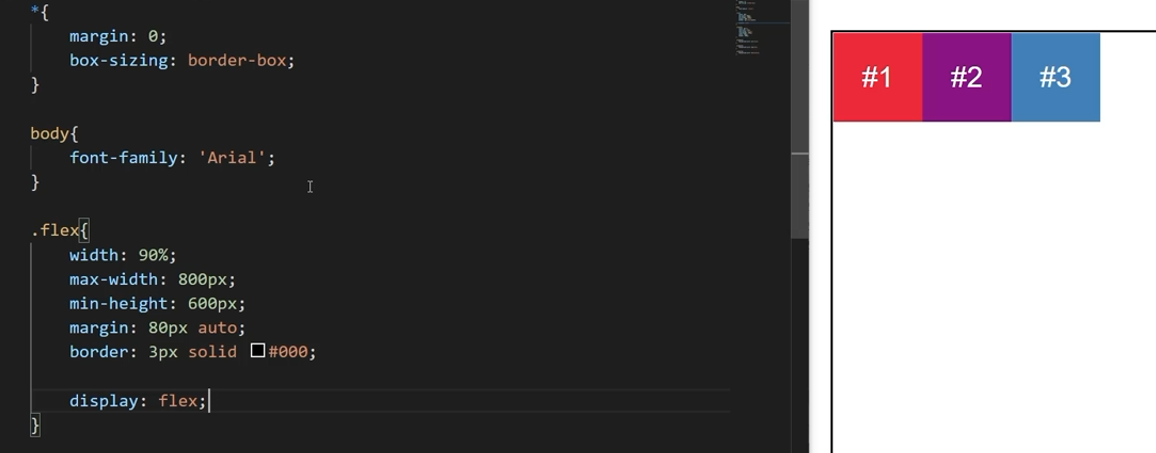


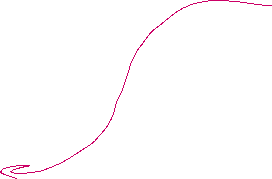


**FLEXBOX**

Adaptable a dispositivos móviles. Se debe tener un contenedor con al menos un elemento

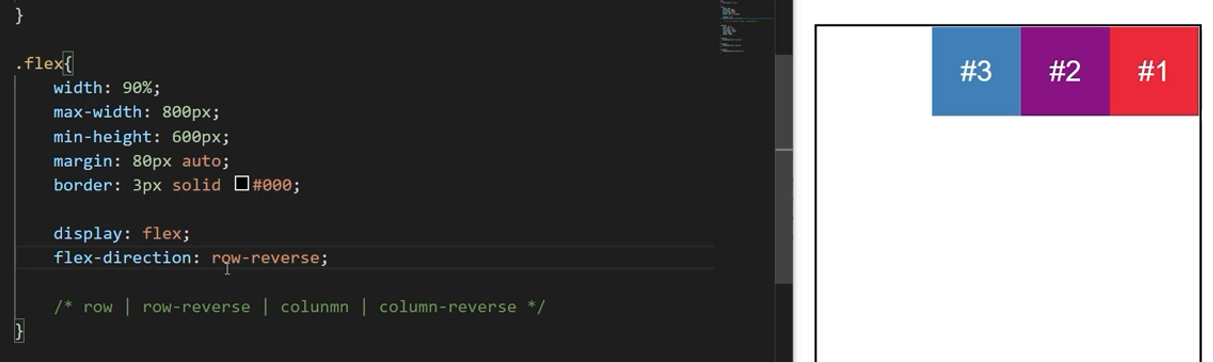
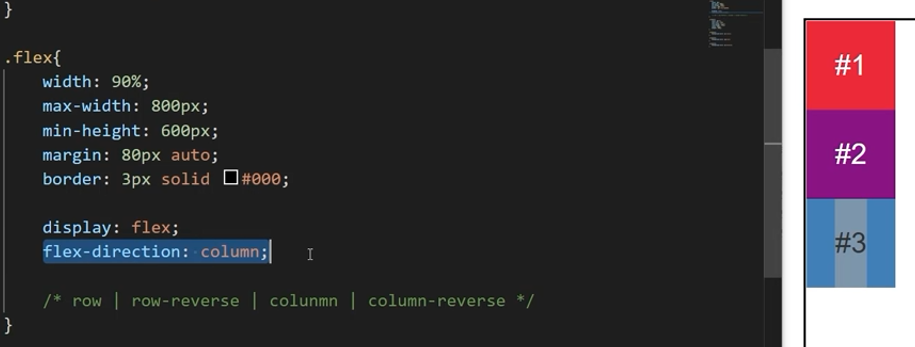
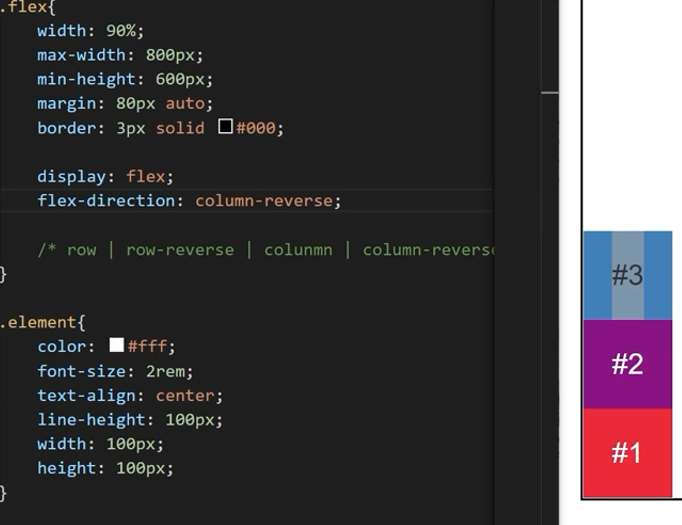
Se declara: display: flex; se alinean de izq-derecha

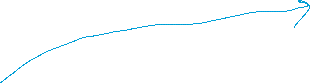
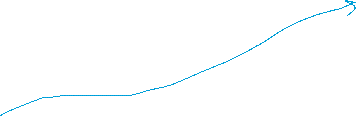
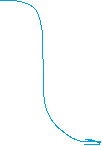




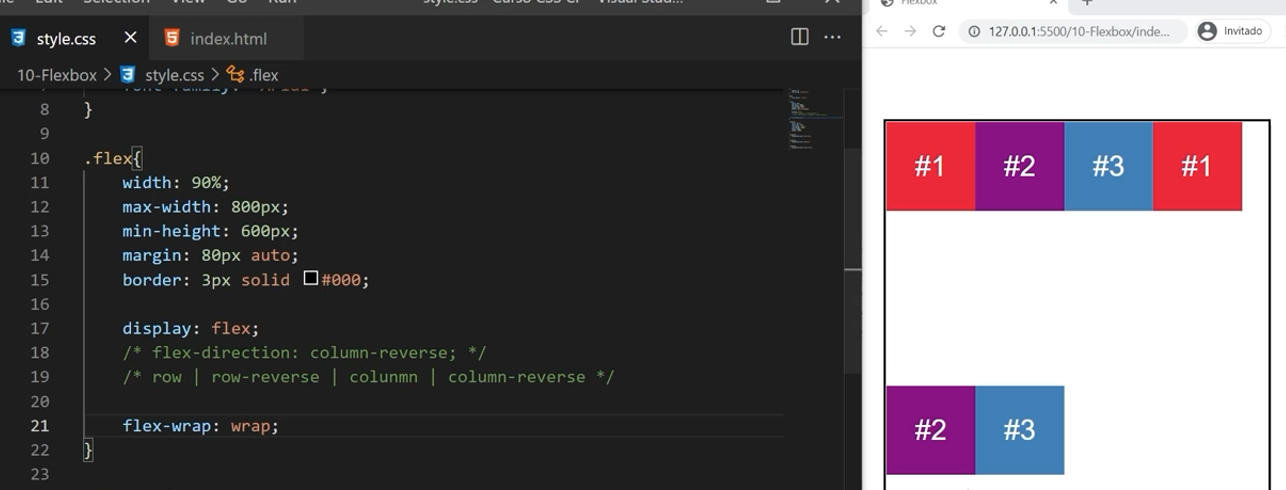
Siempre tiene un **main-axis** y un **cross-axis**, y se cambia con **flex-direction**

flex-direction: row | row-reverse | column | column-reverse;



flex-wrap: nowrap | wrap | wrap-reverse; define el comportamiento de los flex-items y sirve cuando el tamaño de estos (sumado) sobrepasa al width de su padre.

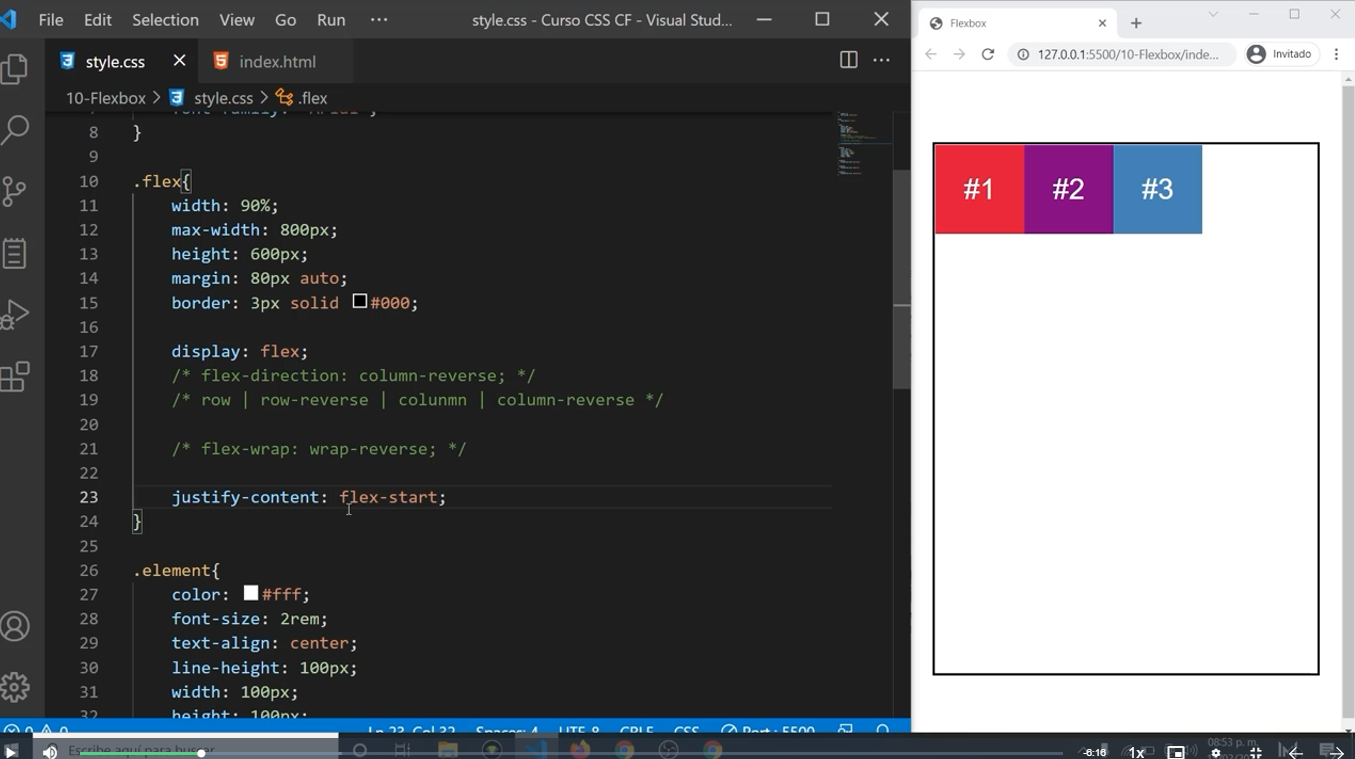


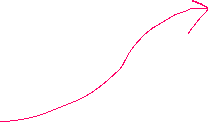


**Alineación**

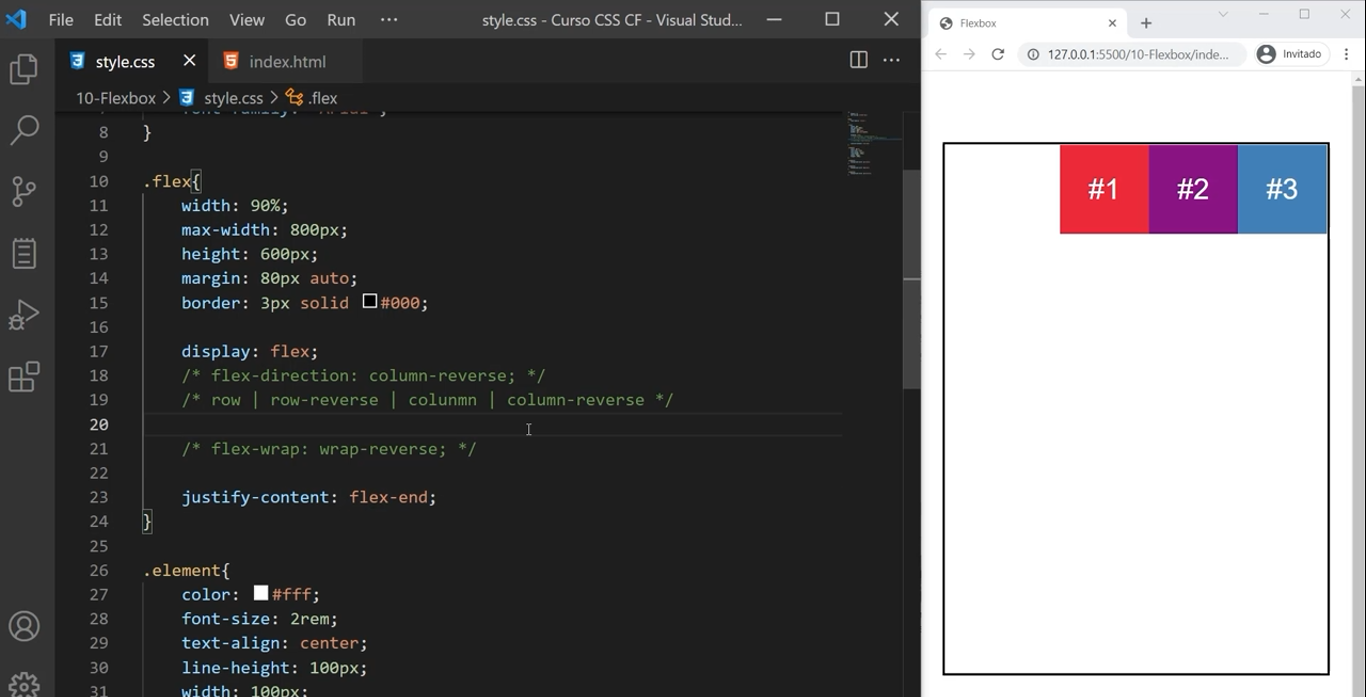
Debe haber espacio disponible en el main-axis para que funcione.

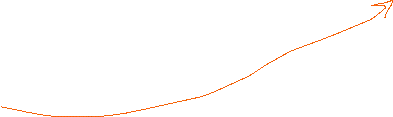
justify-content: flex-start | flex-end | center; agrupa a los flex-items al inicio del main-axis (flex-start por defecto), debe haber espacio suficiente



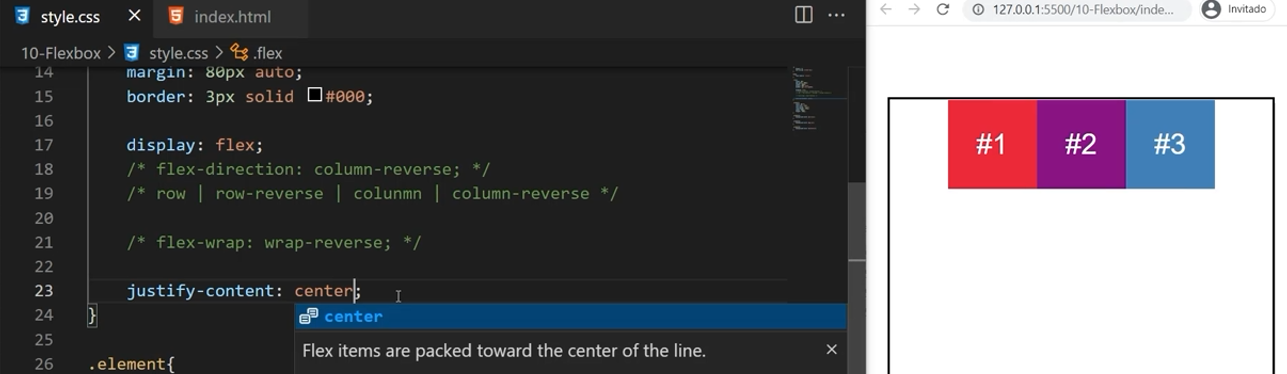


justify-content: flex-end; Los alinea de derecha a izquierda



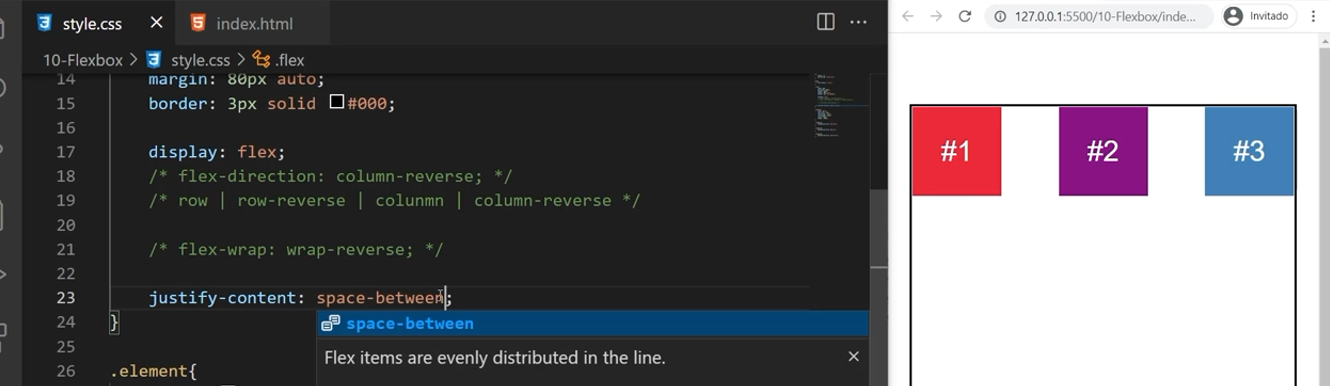


justify-content: center; centra los flex-items en la mitad del container padre





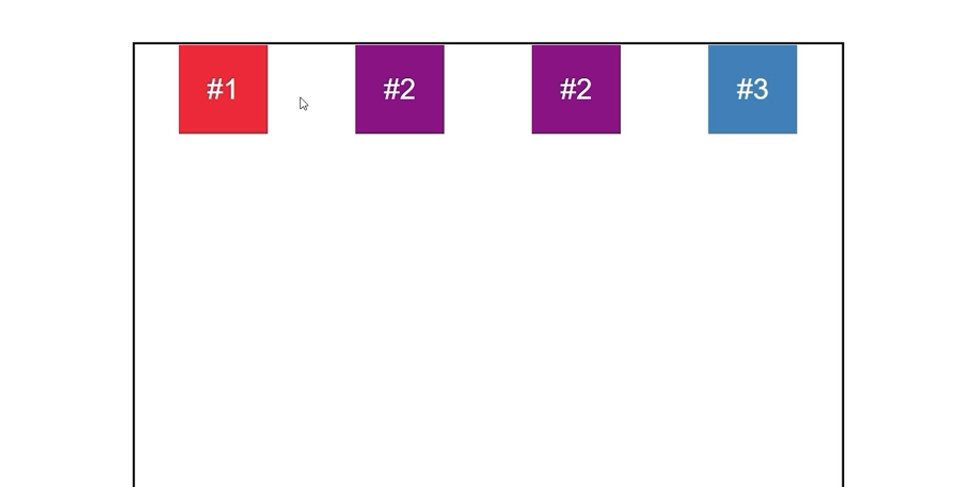
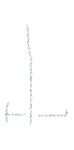
justify-content: space-between; el espacio sobrante se distribuye el espacio en partes iguales entre los flex-items teniendo en cuenta que el primer elemento queda a la izquierda y el último a la derecha.

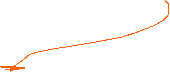




justify-content: space-around; el espacio Sobrante se distribuye equitativamente en ambos lados de cada uno de los flex-items, si hay dos elementos cerca, el espacio se suma.

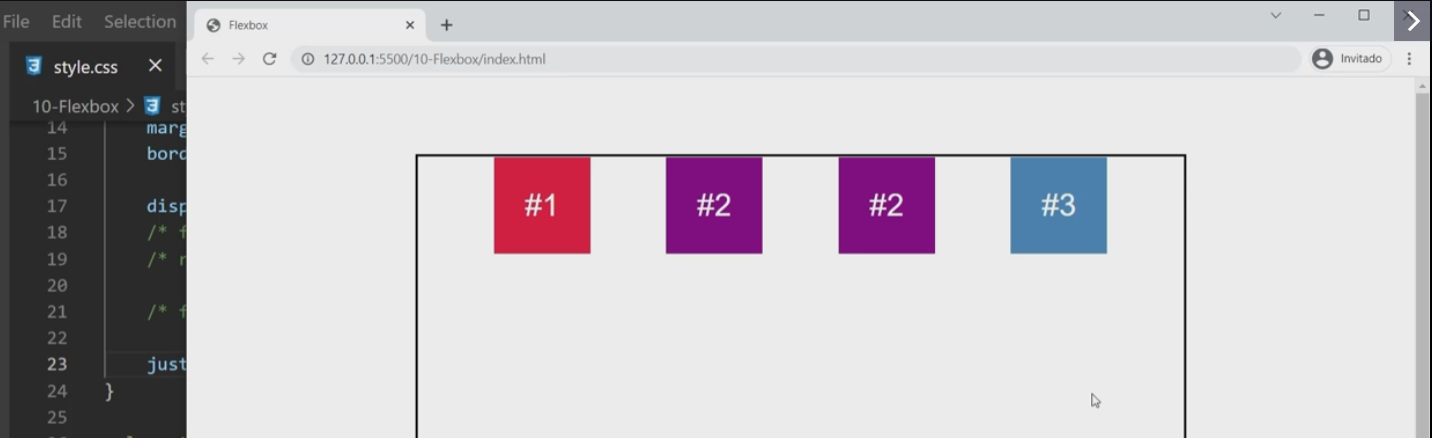


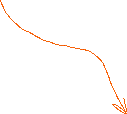


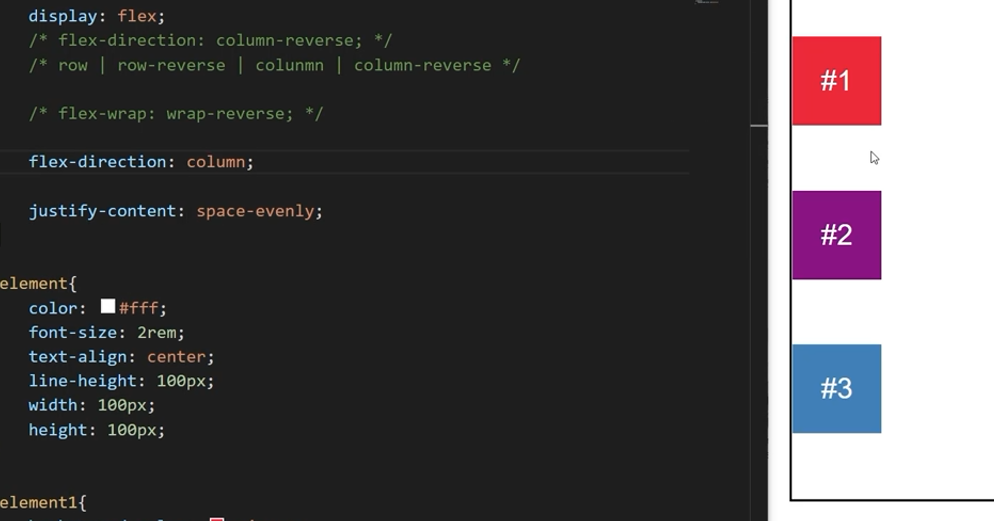


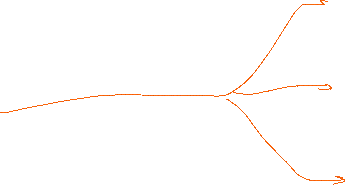
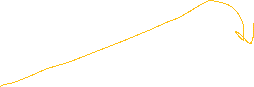
justify-content: space-evenly; el espacio Sobrante se distribuye equitativamente en todo caso, es decir, el espacio a la izquierda del primer elemento, a la derecha del último y entre flex-items es el mismo siempre.











**Alinear elementos en el cross-axis con align-items**

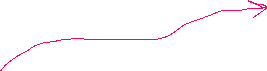
align-items: ; stretch por defecto, debe haber espacio disponible, este estira los elementos por todo el cross-axis siempre y cuando no tengan un height definido

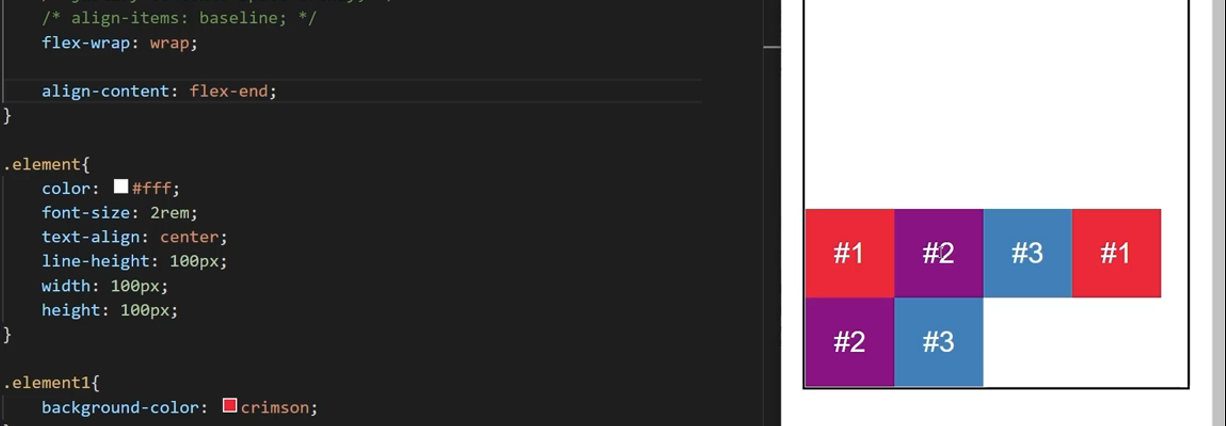


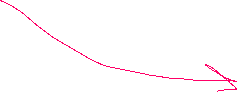
align-items: flex-start | flex-end | center; hace lo mismo que el **justify-content** pero en el cross-axis.

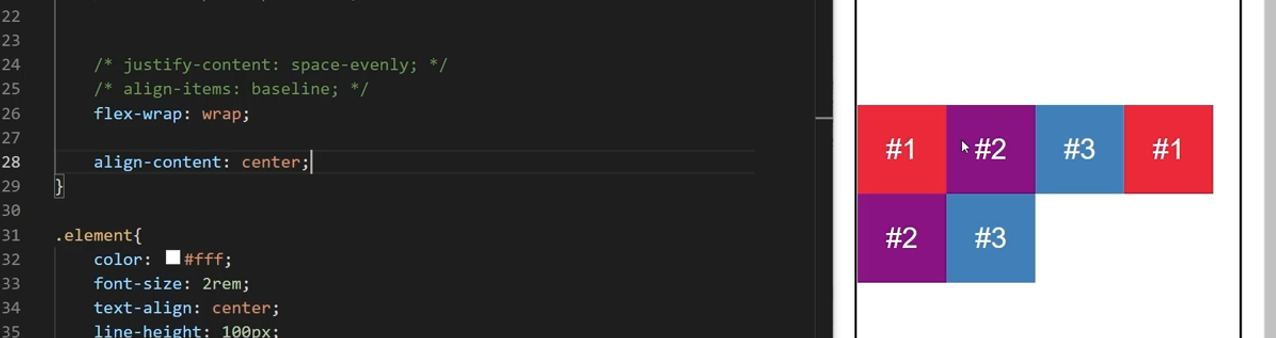
align-content: ; solo funciona cuando hay más de una fila de elementos, es decir, cuando el **flex-wrap: wrap;** y hayan suficientes elementos que se puedan desbordar. hace lo mismo que el **justify-content** pero en el cross-axis.



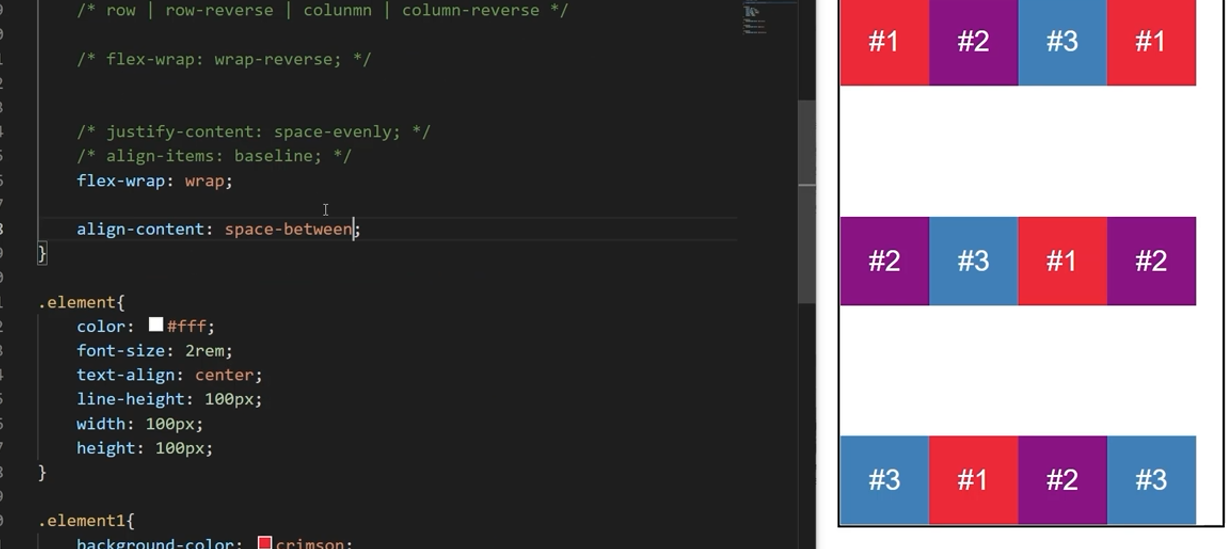


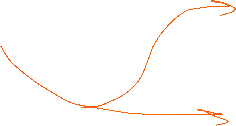




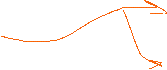




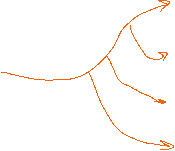






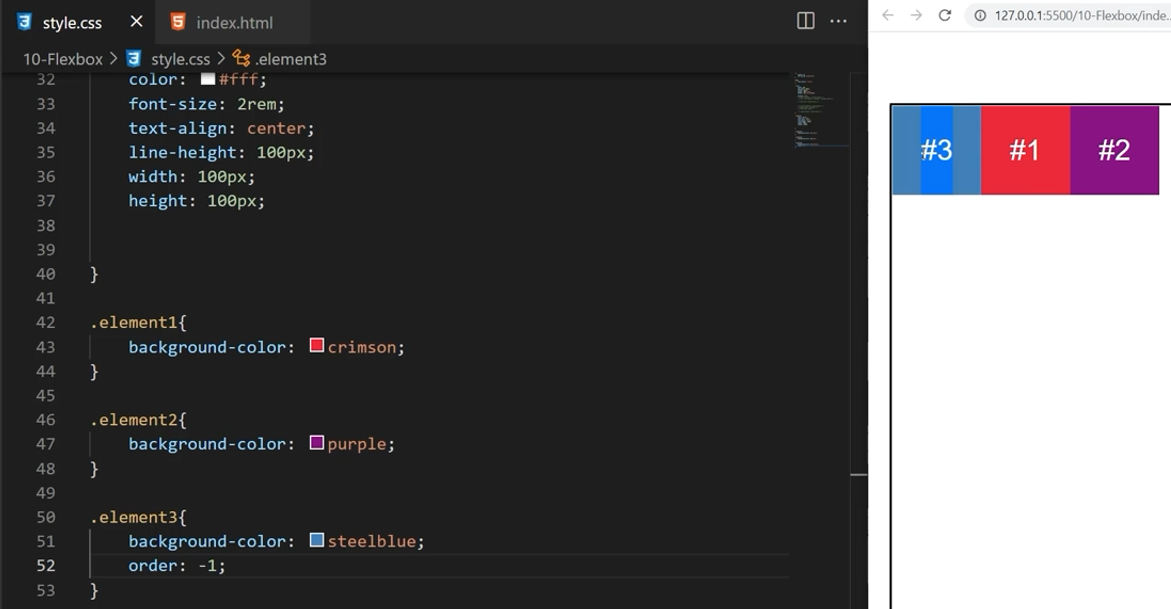


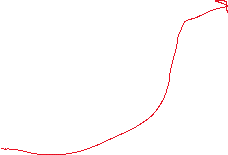




**Propiedades para los flex-items**

order: 0; ordenar los flex-items, controlar la posición de ellos, defecto es order: 0; para cambiar su orden cada uno de los elementos debe tener un order: x; y dependiendo de este valor se acomodan de menor a mayor (si no tienen, se ubican según el html).

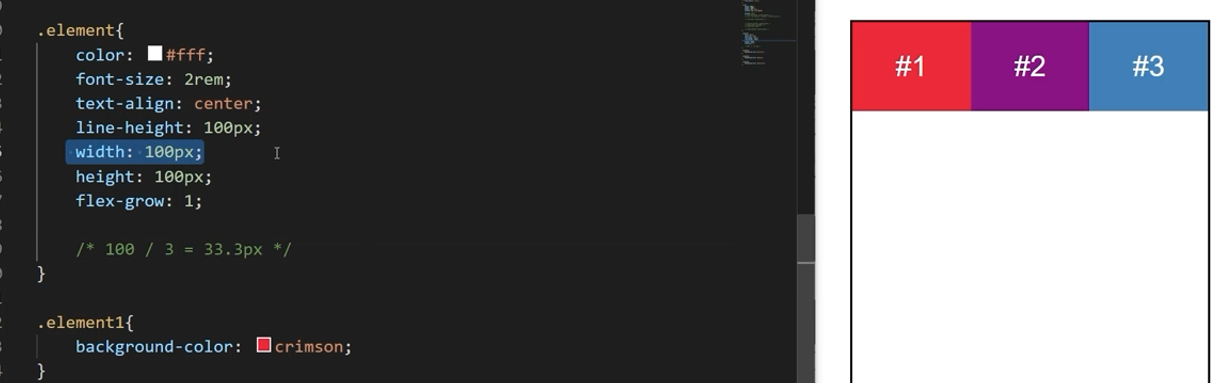




*Como el elemento #3 tiene el menor order, se ubica primero.*

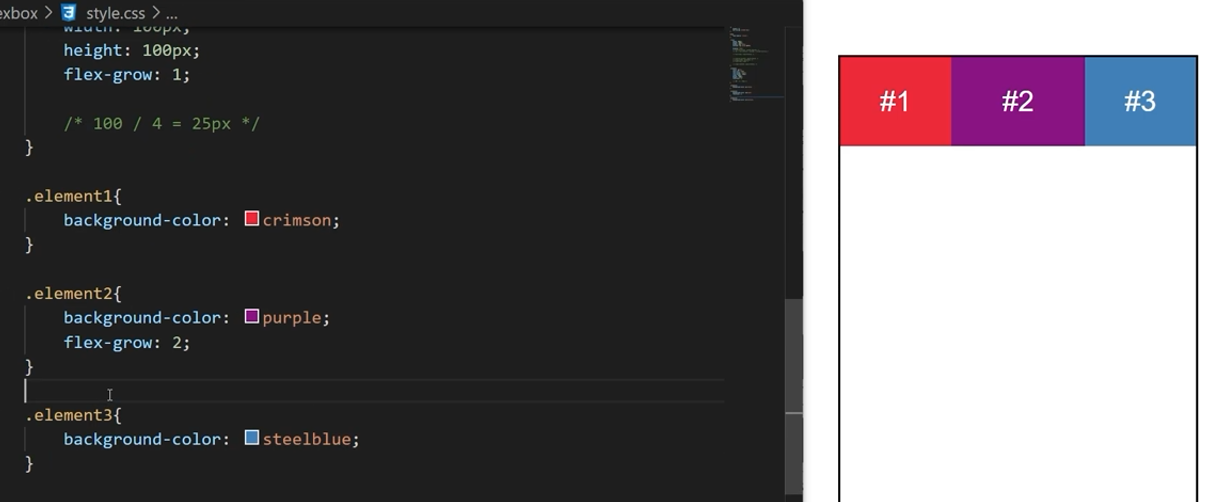
***ESPACIO SOBRANTE***

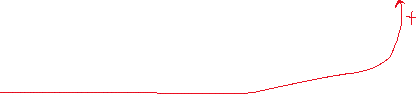
flex-grow: ; factor de crecimiento de los flex-items, se distribuye el espacio sobrante





*Con un flex grow de 1 indica que todos los elementos crecen de igual manera según el espacio sobrante.*

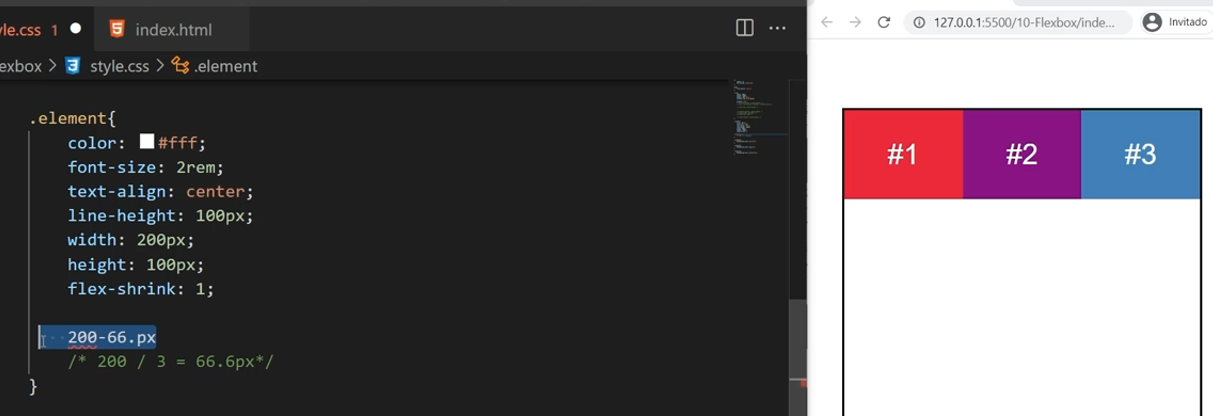




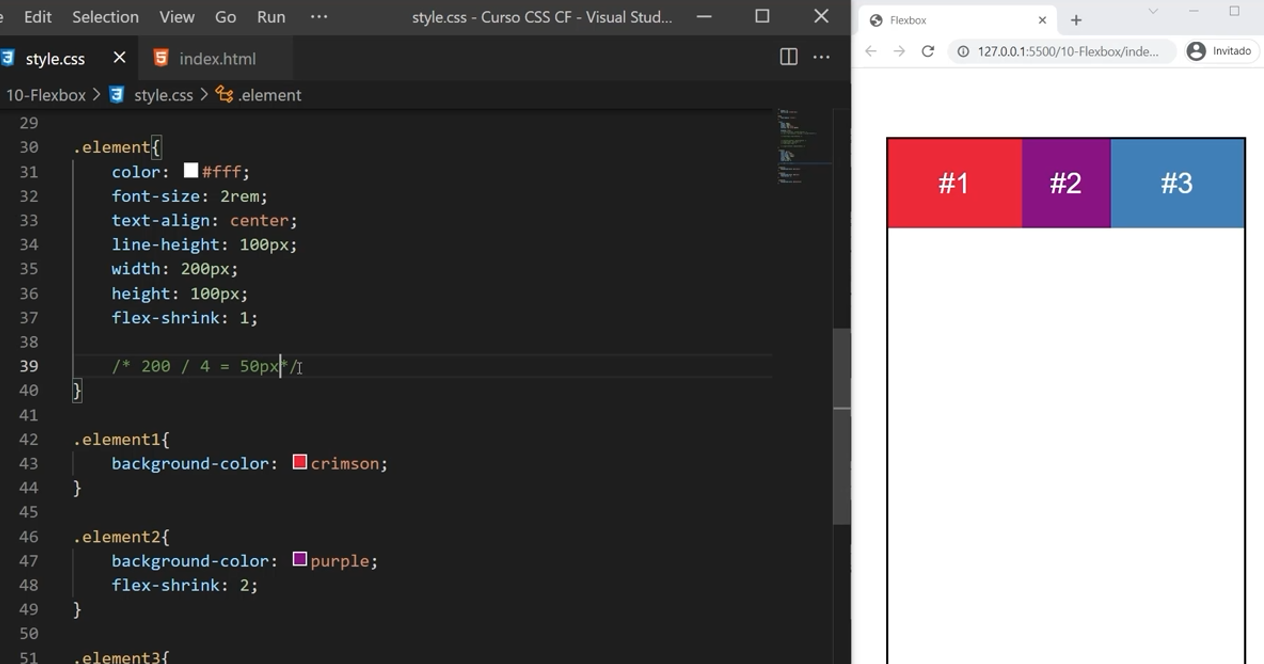
*Al tener el elemento 2 un flex-grow: 2; este se hará más grande que los otros dos, por tanto como son 2 elementos (1 y 3) con un flex-grow: 1; y el elemento 2 con flex-grow: 2; el espacio sobrante se divide entre 4 y se le suma a cada width de cada elemento, por consiguiente, el flex-item #2 será más grande.*

***ESPACIO FALTANTE*** se resta el espacio faltante a los flex-items y por eso se ven más pequeños

flex-shrink: ; no reparte espacio sobrante sino faltante, por defecto es 1, por tanto, la suma de este y la cantidad de elementos, el espacio faltante se divide y el resultado se le resta a cada uno de ellos. Con un valor de cero no se respeta el flex







*Al tener el elemento 2 un flex-shrink: 2; este se hará más pequeño que los otros dos, por tanto, como son 2 elementos (1 y 3) con un flex-shrink: 1; y el elemento 2 con flex-shrink: 2; el espacio faltante se divide entre 4 y se le resta a cada width de cada elemento, por consiguiente, el flex-item #2 será más pequeño.*

flex-basis: auto; defecto es auto. Define el tamaño de los flex-items en el main-axis antes de aplicarles el flex-grow y flex-shrink (es más importante que el width).

align-self: flex-start | flex-end | stretch | center; alinea a través del cross-axis, hace lo mismo que el align-items. Pero se aplica de manera personal, es decir, para 1 solo elemento. Tiene los mismos valores que align-items.

