**FLEXBOX**

[**https://flexboxfroggy.com/#es**](https://flexboxfroggy.com/#es)

· **justify-content**

la cual alinea elementos horizontalmente y acepta los siguientes valores:

flex-start: Alinea elementos al lado izquierdo del contenedor.

flex-end: Alinea elementos al lado derecho del contenedor.

center: Alinea elementos en el centro del contenedor.

space-between: Muestra elementos con la misma distancia entre ellos.

space-around: Muestra elementos con la misma separación alrededor de ellos.

· **align-items**

para ayudar a las ranas a llegar al fondo del estanque. Esta propiedad CSS alinea elementos verticalmente y acepta los siguientes valores:

flex-start: Alinea elementos a la parte superior del contenedor.

flex-end: Alinea elementos a la parte inferior del contenedor.

center: Alinea elementos en el centro (verticalmente hablando) del contenedor.

baseline: Muestra elementos en la línea base del contenedor

stretch: Elementos se estiran para ajustarse al contenedor.

· **flex-direction**

Esta propiedad CSS define la dirección de los elementos en el contenedor, y acepta los siguientes valores:

row: Elementos son colocados en la misma dirección del texto.

row-reverse: Elementos son colocados en la dirección opuesta al texto.

column: Elementos se colocan de arriba hacia abajo.

column-reverse: Elementos se colocan de abajo hacia arriba.

· **Order**

propiedad de elementos individuales

Por defecto, los elementos tienen un valor 0, pero nosotros podemos usar esta propiedad para establecerlo a un número entero positivo o negativo.

ejm: order:-1;, primera posicion

· **align-self**

propiedad que acepta los mismos valores de align-items y sus valores son usados para un elemento específico

ejm: align-self:flex-end;

· **flex-wrap**

distribuirlas correctamente, cual acepta los siguientes valores:

nowrap: Cada elemento se ajusta en una sola línea.

wrap: los elementos se envuelven alrededor de líneas adicionales.

wrap-reverse: Los elementos se envuelven alrededor de líneas adicionales en reversa.

Las dos propiedades flex-direction y flex-wrap son usadas en conjunto con la propiedad flex-flow

Esta propiedad abreviada, acepta un valor de cada una separada por un espacio.

ejm: flex-flow:column wrap;

· **align-content**

establece como múltiples líneas están separadas una de otra, esta propiedad acepta los siguientes valores:

flex-start: Las líneas se posicionan en la parte superior del contenedor.

flex-end: Las líneas se posicionan en la parte inferior del contenedor.

center: Las líneas se posicionan en el centro (verticalmente hablando) del contenedor.

space-between: Las líneas se muestran con la misma distancia entre ellas.

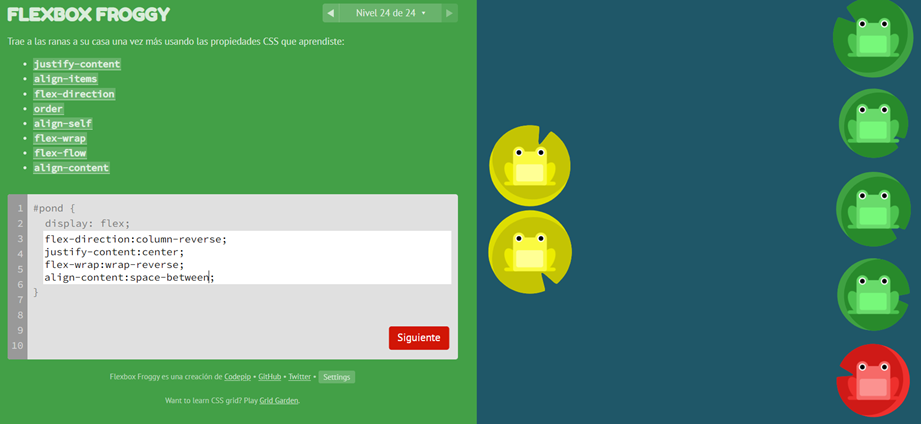
space-around: Las líneas se muestran con la misma separación alrededor de ellas.

stretch: Las líneas se estiran para ajustarse al contenedor.

Esto puede ser confuso, pero align-content determina el espacio entre las líneas, mientras que align-items determina como los elementos en su conjunto están alineados dentro del contenedor. Cuando hay solo una línea, align-content no tiene efecto.

**EJEMPLO COMPLEJO**





**GRID**

· **grid-column-start**

el 3er borde vertical contando desde la izquierda de la cuadrícula. la expansión por defecto del elemento en la cuadrícula será de exactamente una columna

ejm: grid-column-start: 3;

grid-column-start:-3;

**grid-column-end.**

se especifica la linea vertical final del relleno, ¡No queremos malgastar agua!

ejm:

grid-column-start: 1;

grid-column-end:4;

**span**

en lugar de definir un elemento en la cuadrícula basado en la posicion inicial y final,

puedes definirlo basado en la longitud de columnas deseada

span solo funciona con valores positivos.

ejm:

grid-column-start: 2;

grid-column-end: span 2;

**grid-column**

propiedad abreviada en un alemento en especifico que acepta ambos valores a la vez, separados /

ejm: grid-column: 2 / 4; , grid-column:4/6;

**span en grid-column**

La palabra clave span también funciona con esta propiedad abreviada

ejm: grid-column: 2/span 4;

**grid-row-start**

Una de las cosas que diferencia las cuadrículas de CSS de flexbox es que puedes posicionar los elementos fácilmente en 2 dimensiones: columnas y filas. grid-row-start funciona de manera semejante a grid-column-start pero a lo largo del eje vertical.

ejm: grid-row-start:3;

grid-row:3/6; especificando inicion y final

otro ejm:

grid-column:2;

grid-row:-2;

otro ejm:

grid-column:2/6;

grid-row:1/6;

**grid-area**

admite cuatro valores separados por barras oblicuas: grid-row-start, grid-column-start, grid-row-end, seguido de grid-column-end.

ejm: grid-area: 1 / 1 / 3 / 6;.

ejm: grid-area:1/2/4 /6;

**order**

Por defecto, el valor de order de todos los elementos es igual a 0, pero puede ser establecido a cualquier valor positivo o negativo, de manera similar a z-index.

ejm: order:-1;

**grid-template-columns**

define los nombres de línea y las funciones de tamaño de seguimiento de las columnas de la cuadrícula

ejm: grid-template-columns: 50%;

grid-template-columns:repeat(8,12.5%);

grid-template-columns: 100px 3em 40%; usando unidades de pixels y ems

**fr**

una nueva medida, cada unidad fr asigna una porción del espacio disponible.

Por ejemplo, si dos elementos están establecidos a 1fr y 3fr respectivamente el espacio se divide en 4 porciones iguales; el primer elemento ocupa 1/4 del espacio y el segundo elemento los 3/4 restantes.

Ahora las malas hierbas ocupan 1/6 de tu primera fila y las zanahorias los restantes 5/6. Crea dos columnas con ese ancho empleando fr.

ejm: grid-template-columns: 1fr 5fr;

grid-template-columns: 50px 1fr 0 0 50px 0fr;

grid-template-columns: 75px 6fr 4fr;

grid-template-rows: 50px 0 0 0;

**grid-template**

combina grid-template-rows y grid-template-columns

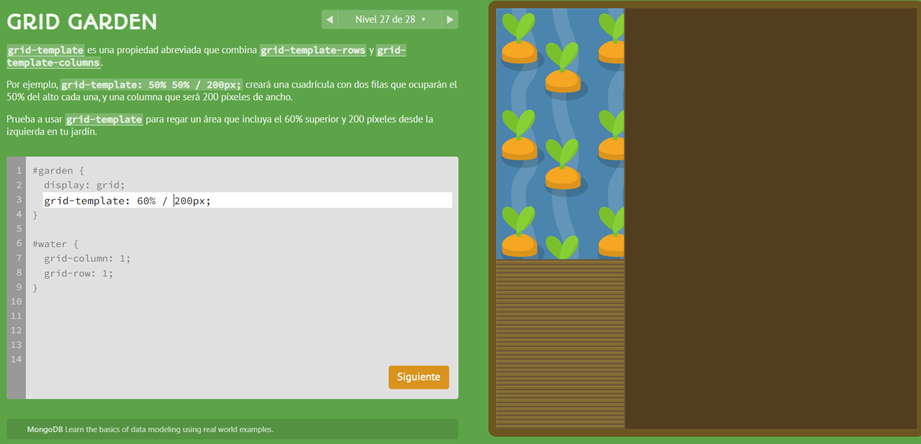
ejm: grid-template: 60% / 200px;

***no entenidi***









***no fui capaz***

