MAQUETACIÓN

- Position
- Diseño fluido
- Diseño flexible
- Diseño en rejilla

Position

Basado en el empleo de la propiedad css position.

```
position: static;
position: relative;
position: absolute;
position: fixed;
position: sticky;
```

static

Esta palabra reservada permite que el elemento utilice el comportamiento normal. Las propiedades top, right, bottom, left and z-index no se aplican.

relative

Dispone todos los elementos como si el elemento no tuviera posición y luego ajusta la posición del elemento sin alterar la disposición (por tanto dejando un vacío donde debería estar el elemento si no estuviera posicionado).

absolute

No deja espacio para el elemento. En su lugar, lo posiciona en unas coordenadas determinadas relativas a la posición más cercana de su elemento padre o del bloque contenedor inicial.

fixed

No deja espacio para el elemento. En su lugar, lo posiciona en unas coordenadas determinadas relativas a la ventana de visualización, que no se mueve al moverse la página. (O cuando se imprime, la posición es fija en cada página).

sticky

La posición de la caja se calcula de acuerdo con el flujo normal.

Diseño fluido

Una web tiene diseño fluido cuando su tamaño se ajusta a la dimensión horizontal de la pantalla de forma automática y sin necesidad de una barra de desplazamiento horizontal (scroll). El diseño se expande al ancho disponible de la pantalla porque el tamaño de los distintos elementos (div) es un porcentaje del total disponible (100%) de la pantalla.

```
float: left;
float: right;
float: none;
clear: left;
clear: right;
clear: both;
```

float

La propiedad float especifica si un elemento debe salir del flujo normal y aparecer a la izquierda o a la derecha de su contenedor, donde los elementos de texto y los en línea aparecerán a su alrededor.

clear

La propiedad clear especifica si un elemento puede estar al lado de elementos flotantes que lo preceden o si debe ser movido (cleared) debajo de ellos.

Diseño flexible

Lo que caracteriza un diseño flexible es su habilidad para alterar el ancho y alto de sus elementos para ajustarse lo mejor posible al espacio disponible en cualquier dispositivo. Un contenedor flexible expande sus elementos para rellenar el espacio libre, o los comprime para evitar que rebasen el área prevista.

```
display: flex
flex-direction: row | row-reverse | column | column-reverse;
justify-content: flex-start | flex-end | center | space-between |
        space-around | space-evenly;
align-items: flex-start | flex-end | center | baseline | stretch
order: 1;
min-width: 10%;
min-height: 10%;
flex-basis: 100px;
flex-grow: 3;
flex-shrink: 1;
```

flex-direction

La propiedad flex-direction establece el eje principal.

justify-content

La propiedad justify-content define cómo los elementos flexibles se disponen a lo largo del eje principal en la línea en curso.

align-items

La propiedad align-items define cómo los elementos flexibles se disponen a lo largo del eje secundario de la línea en curso.

order

La propiedad order asigna elementos a grupos ordinales y determina qué elementos aparecen primero.

min-height y min-width

La propiedades min-height y min-width tienen un nuevo valor, auto que establece el tamaño mínimo de un elemento flexible.

flex-basis

La propiedad flex-basis especifica el tamaño principal inicial de un elemento flexible.

flex-grow

La propiedad flex-grow especifica el factor de crecimiento de un elemento flexible. Especifica la cantidad de espacio dentro del contenedor flexible que el elemento debería tomar.

flex-shrink

La propiedad flex-shrink especifica el factor de reducción de un elemento flexible. El elemento flexible se reducirá para encajar en el contenedor de acuerdo al factor de reducción.

Diseño en rejilla

Grid Layout es un modelo de diseño que tiene habilidades para controlar el diseño de cuadrícula, siendo optimizado para diseños bidimensionales (aquellos en los que se desea alinear el contenido en ambas dimensiones).

Diseño sin áreas

Es posible crear cuadrículas con un tamaño explícito. Para ello, sólo tenemos que usar las propiedades CSS grid-template-columns y grid-template-rows, que sirven para indicar las dimensiones de cada celda de la cuadrícula, diferenciando entre columnas y filas.

```
#grid {
  display: grid;
  grid-template-columns: 50px 300px;
  grid-template-rows: 200px 75px 75px;
}
/*Esto significa que tendremos una cuadricula con 2 columnas y con 3 filas.*/
```

grid-template-columns

Esta propiedad establece el número y tamaño de las columnas (eje horizontal).

grid-template-rows

Esta propiedad establece el número y tamaño de las filas (eje vertical).

```
#title { grid-column: 1; grid-row: 1; }
#score { grid-column: 1; grid-row: 3; }
#stats { grid-column: 1; grid-row: 2;}
#board { grid-column: 2; grid-row: 1 / span 2; }
#controls { grid-column: 2; grid-row: 3; justify-self: center; }
```

grid-column

Esta propiedad establece en que columna de la rejilla será incluido el elemento identificado.

grid-row

Esta propiedad establece en que fila de la rejilla será incluido el elemento identificado.

Diseño con áreas

También es posible trabajar con áreas, empleando las siguientes propiedades:

grid-template-areas

Esta propiedad nos permite enumerar las distintas áreas.

```
grid-template-areas:
            "titulo      titulo       titulo     titulo     titulo"
            "descripcion contexto     producto   cronograma secuencia"
            "descripcion competencias producto   recursos   secuencia"
```

grid-area

Esta propiedad nos permite indicar en que área debe mostrarse la etiqueta.

```
#title      { grid-area: titulo;}
```