CSS Flexbox

Programación

Investigar los conceptos básicos sobre Flex Box y responder las siguientes preguntas

**Conceptos Básicos**

1. ¿Qué es el Flex Box? ¿Qué propiedades tiene?

Flexbox es un modelo de caja, permite alinear y distribuir los elementos dentro de un contenedor de manera más eficiente. Muy utilizado en aplicaciones o páginas responsivas.

Tiene muchas propiedades como:

* Display: flex; hace q los elementos hijos se conviertan en elementos flexibles.
* Flex: direction; muestra en que dirección van a ir los elementos (row: horizontalmente/row-reverse: horizontal pero al revés)(column: en columna/column/reverse: columna inversa).
* Justify-content: alinea los elementos en el eje principal horizontalmente (flex-start: inicio/flex-end:final/center:centra los elementos/space-between: el espacio restante lo divide entre los elementos/space-around: distribuye el espacio pero con un margen al rededor de cada elemento).
* Align-items; alinea los elementos en el eje principal verticalmente (stretch: estira todos los elementos para que llenen la caja, es el predeterminado/flex-start: inicio/flex-end: final/center: en el centro).
* align-content: Alinea múltiples líneas de elementos flexibles cuando hay espacio adicional en el contenedor. Aplica si hay más de una línea de elementos.
* Flex-wrap: wrap; si se achica la pagina los elementos se posicionan abajo.
* Flex-grow: permite definir cuanto puede crecer una caja y ocupa todo el espacio disponible.
* Flex-basis: establece un tamaño base de la caja.
* Flex-shrink: permite definir el máximo que se puede achicar una página.
* Order: cambia el orden de los elementos.
* Align-self: permite agarrar un solo elemento y ubicarlo donde queramos.

1. ¿Qué propiedades de CSS se utilizan para crear un contenedor Flex y alinear sus elementos hijos en una fila?

Al contenedor principal se le aplica display:flex; y luego flex-direction:row; para ubicarlos en una fila.

1. ¿Cuál es la función de la propiedad flex-direction y qué valores puede tomar?

* Flex: direction; muestra en que dirección van a ir los elementos (row: horizontalmente predeterminado / row-reverse: horizontal pero al revés)(column: en columna / column-reverse: columna inversa).

1. Explica la diferencia entre justify-content y align-items en el contexto de Flexbox.

* Justify-content: alinea los elementos en el eje principal horizontalmente
* Align-items; alinea los elementos en el eje principal verticalmente o en el eje perpendicular.

**Propiedades de Flex Ítems**

1. ¿Cómo afecta la propiedad flex-basis al tamaño de un elemento Flex?

La propiedad flex-basis define el tamaño inicial de un elemento flexible (flex item). Establece el tamaño que el elemento debería tomar antes de aplicar el crecimiento o disminución basado en las propiedades flex-grow y flex-shrink.

1. ¿Qué hace la propiedad align-self y cómo difiere de align-items?

La propiedad align-self permite sobrescribir la propiedad align-items para un solo elemento flex. Mientras que align-items se aplica a todos los elementos flex en un contenedor, align-self.

**Ejemplos Prácticos**

1. Imagina que tenes un contenedor Flex con cuatro elementos. Si utilizas justify-content: space-between;, ¿cómo se distribuirán los elementos dentro del contenedor?

El primer elemento se alinea al inicio del contenedor, El último elemento se alinea al final y Los dos elementos del medio se colocan en la misma distancia que hay entre ellos, llenando el espacio restante.

1. Si quieres centrar un elemento dentro de un contenedor Flex, ¿qué combinaciones de propiedades usarías para lograr esto tanto en el eje principal como en el eje transversal?

.caja {

display: flex;

justify-content: center; /\* Centra en el eje principal \*/

align-items: center; /\* Centra en el eje transversal \*/

}

**Problemas a resolver y realizar en código las soluciones**

1. ¿Qué propiedad y valores puedes usar para hacer que todos los elementos hijos de un contenedor Flex tengan el mismo tamaño en una fila?

.caja {

display: flex;

}

.caja > .item {

flex: 1; /\* Hace que todos los elementos crezcan y ocupen el mismo espacio \*/

}

1. ¿Cómo podes alinear un elemento al final del contenedor Flex, mientras se mantienen los otros elementos alineados al inicio?

.caja {

display: flex;

border: 2px solid #333;

padding: 10px;

}

.item {

padding: 10px;

background-color: #4CAF50;

color: white;

margin-right: 10px;

}

.item3 {

margin-left: auto; /\* Empuja este elemento al final \*/

background-color: #f44336; /\* Color diferente para resaltar \*/

}

1. Centrar un solo elemento dentro de un contenedor Flex tanto horizontal como verticalmente. El contenedor tiene un tamaño fijo, y el elemento debe estar centrado en el medio del contenedor. Cómo se soluciona?

.caja {

display: flex;

justify-content: center; /\* Centra el elemento horizontalmente \*/

align-items: center; /\* Centra el elemento verticalmente \*/

width: 300px;

height: 300px;

border: 2px solid #333;

}

.item {

padding: 20px;

background-color: #4CAF50;

color: white;

text-align: center;

}

1. En un contenedor Flex, queremos que el primer elemento tenga un ancho fijo de 100px, mientras que los otros elementos deben crecer y ocupar el espacio restante de manera equitativa. Cómo lo soluciono?

.container {

display: flex;

border: 2px solid #333;

}

.item-fixed {

width: 100px; /\* Ancho fijo \*/

background-color: #4CAF50;

color: white;

padding: 10px;

}

.item-flex {

flex: 1; /\* Ocupa el espacio restante de manera equitativa \*/

background-color: #f44336;

color: white;

padding: 10px;

text-align: center;

}

1. Queremos que los elementos dentro de un contenedor Flex no se reduzcan por debajo de un tamaño mínimo especificado. Además, los elementos deben distribuirse uniformemente en la fila. Cómo establezco tamaño mínimo de los elementos?

.caja {

display: flex;

justify-content: space-between; /\* Distribuye los elementos uniformemente en la fila \*/

border: 2px solid #333;

}

.item {

flex: 1; /\* Permite que los elementos crezcan para llenar el espacio \*/

min-width: 100px; /\* Tamaño mínimo especificado \*/

background-color: #4CAF50;

color: white;

padding: 10px;

text-align: center;

}