Checkpoint2

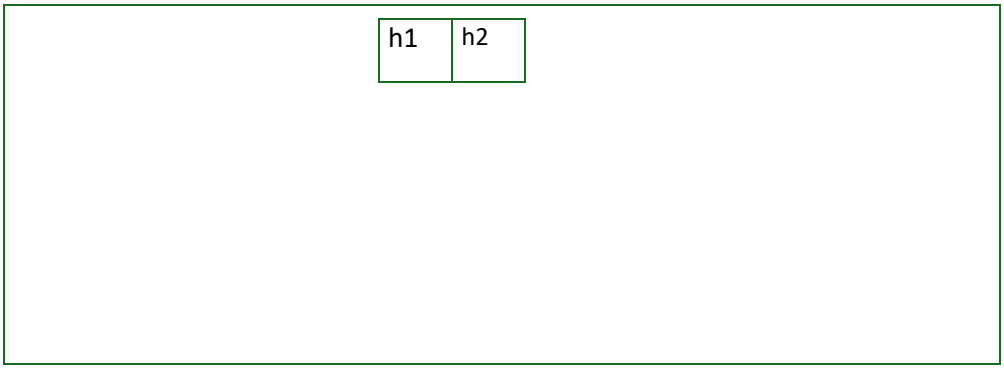
1.Justify-content: 4 valores

Justify-content: center;

Esta característica se usa para centrar uno o más elementos dentro de un contenedor flex. Las imágenes por ejemplo se sitúan en el centro de la página.

Aunque tengamos dos imágenes o dos textos, si se encuentran en el mismo div se piensa que es un solo elemento. Coge todo y lo traslada al centro. No deja espacio arriba ni abajo, todo va pegado al centro de la línea.

El espacio entre el comienzo de la página(según la vista) y el contenedor y entre el contenedor y el final de la página es el mismo.



Justify- content: flex-start;

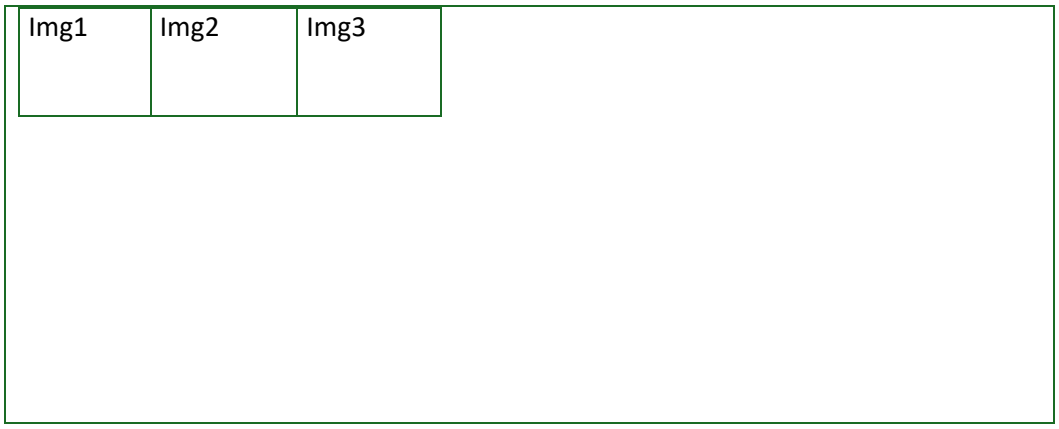
Por defecto, aunque no pongamos nada en justify-content, si tenemos un contenedor flex todo lo que esté dentro de él se coloca a la izquierda de la pantalla, al principio de nuestra visión.

Esto se puede arreglar sino con flex-start.

El márgen del primer elemento se coloca pegado al borde del comienzo de la línea y los siguientes elementos van seguidos.

Tenemos tres imágenes que queremos que estén al comienzo de la página.

Si utilizamos flex-start se situaran al comienzo de la línea, una detrás de otra. Porque por defecto los elementos se colocan en línea.

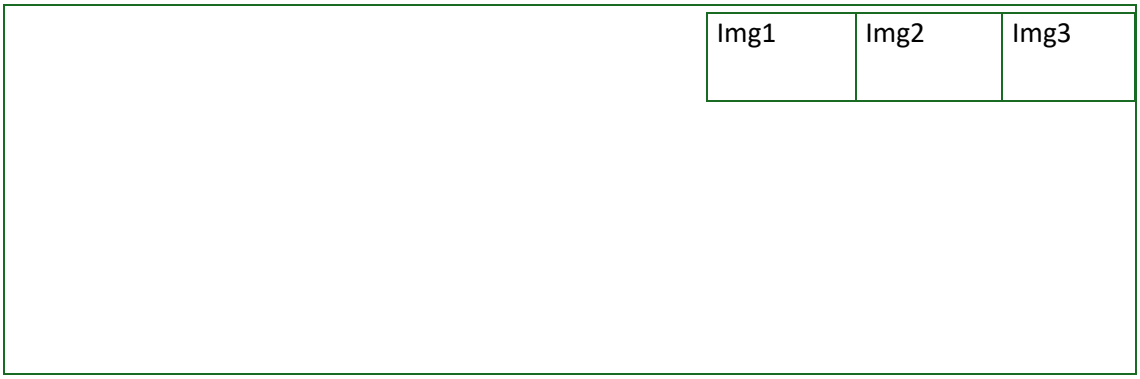


Justify-content: flex-end;

Es la propiedad contraria a flex-start. Sitúa los elementos o el elemento, al lado derecho de la página. Todo se mueve al final de la página.

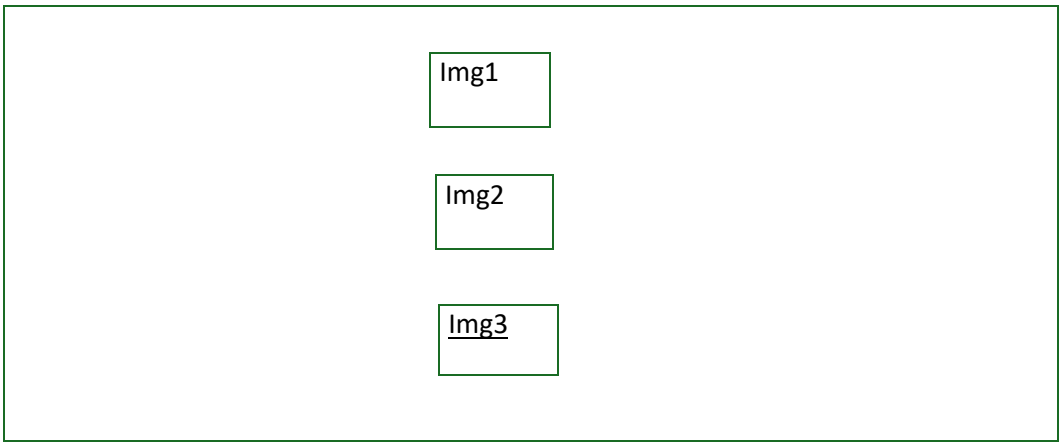
Se coloca el primer elemento pegado al borde del final de la línea.

Las mismas tres imágenes de antes , ahora se van a ajustar al lado derecho de la pantalla y siguiendo el orden.



Justify-content: space-around;

Lo que nos permite hacer esta propiedad es distribuir las imágenes (por ejemplo) desde el principio de la página hasta el final y uniformemente. Es valioso si no quieres que lleguen hasta los costados (bordes). Los elementos se centran en la página pero no completamente de lado a lado o de arriba abajo. Siempre deja un espacio alrededor de cada elemento y este espacio siempre es el mismo.



2. Objetivo primordial.

El objetivo primordial es aquel para el que va a crearse una aplicación. Todas las acciones, todos los elementos, en si, todo el contenido debe ir encaminado a lograr ese objetivo primordial. Es lo que quieres que hagan las personas al ver tu aplicación. Todo debe ir hacia este objetivo.

Tenemos una historia final que hay que contarla mediante distintas características.

Ejemplo: El objetivo primordial de una aplicación como Zara es que el usuario compre su ropa.

Para ello te mostrará su colección en fotos con modelos y la facilidad para que puedas comprar al instante si algo te gusta.

3. Mapa de sitio.

Hay que tener claros los objetivos a los que queremos llegar para construir un mapa de sitio web.

Hay que identificar las páginas que necesitamos en la aplicación y dentro de ellas, el tipo de elementos que queremos incluir para acertar con lo que necesita el usuario. Debe responder claro a las historias de los usuarios.

Cuando se ha identificado la experiencia de usuario hay que averiguar cómo quiere cada usuario manejar la aplicación (no todos los usuarios van a querer lo mismo, ni utilizarla igual) y esto hay que plasmarlo en el mapa. Este irá directamente a los desarrolladores para que sepan lo que tienen que crear. Traducir el mapa a código.

En el mapa puedes añadir o cambiar elementos sin necesidad de manipular el diseño ya acabado.

Por lo tanto, el mapa es un puente de comunicación entre el creador y el desarrollador que es el último en recibir el mapa.

Esto no es el resultado final de la aplicación, ni cómo va a lucir. Es muy genérico.

Sólo con mirar el mapa ves organizado el contenido y la estructura de la aplicación y como se conecta todo en las páginas.

4. Wireframe de baja fidelidad

Aquí vas a tener en cuenta realmente los objetos que necesitas que existan, todo lo que va a aparecer en cada página. No interesa para nada que le guste o no al usuario, es lo que tiene que aparecer para que funcione la aplicación. No hay que ponerlo bonito.

Si necesitamos un botón en determinado lugar se pone y punto. Luego ya llegará el diseño pero ahora sólo se planifica lo que se va a aparecer en las páginas.

Esto significa que no se piensa en la interfaz de usuario en absoluto. Hay que concentrarse en lo que va a necesitar el usuario para utilizar esta herramienta, que sea útil y funcione bien.

Aquí se verá como quedan las páginas con sus componentes y dónde se colocan.

Estos wireframes se realizan después del mapa porque ya se empieza a necesitar un contexto más amplio.

5. Git

Es como una nube para desarrolladores. Un sistema que se utiliza para incluir proyectos y los cambios que se van realizando en ellos. Está abierto a todo el mundo , si quieres.

Los archivos y proyectos se almacenan en el propio ordenador. Puedes hacer cambios siempre que lo desees y compartirlos con otros desarrolladores.

Git permite trabajar en equipo. Dos personas pueden estar trabajando en un mismo archivo y los cambios se van fusionando automáticamente.

A Git se accede por la terminal del ordenador. En este sistema se crean repositorios que es donde colocas tus archivos y puedes volver a ellos con tu historial y cambiarlos siempre que quieras.

Los desarrolladores son su principal usuario para almacenar el código de las aplicaciones, pero es libre para cualquier persona, de código abierto.

6. Flex-direction.

Establece la dirección de los elementos en una página.

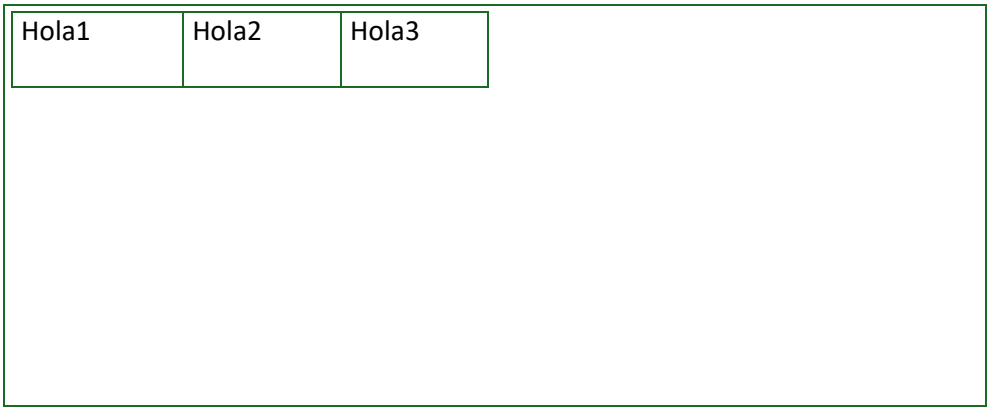
Te especifica cómo se van a colocar dentro de un contenedor. Porque para poder utilizar esta propiedad tiene que haber un contenedor flex.

Esto va a definir el eje principal de la página y su dirección dentro de ella, de forma normal o al revés.

Flex-direction: row;

Los elementos se colocan en la misma dirección, como si fuera un texto. Este es el valor predeterminado y se organiza horizontalmente, en modo fila.

La dirección flex por defecto siempre es fila.

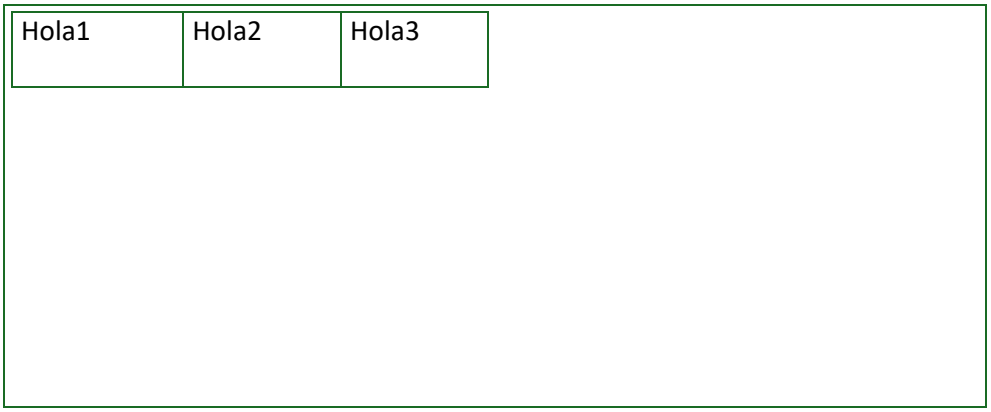


Si tenemos estos tres encabezados aunque no pondríamos nada en flex-direction, los tres aparecerían así colocados si usamos display: flex.

Flex- direction : column;

Los elementos se colocan empezando por arriba hacia abajo. Todo se pone verticalmente. Se apilan los elementos uno encima de otro.

Cuando ponemos display: flex los elementos se colocan como en el siguiente cuadro.



Si utilizamos flex-direction: column, los elementos se colcarán así:

Flex-direction: column-reverse;

Los elementos se colocan verticalmente como en una columna pero en el orden opuesto a como estaban colocados. Se colocan de abajo hacia arriba.