

# Tema 5

Patrones de diseño web adaptativo. Frameworks web

# 5. Patrones de diseño web adaptativo. Frameworks web

## Introducción

El diseño web adaptativo busca que los sitios web se adapten a distintas resoluciones y tamaños de pantalla, un reto que ha aumentado con la diversidad de dispositivos, como móviles, tabletas y televisores inteligentes.

Inicialmente, algunas empresas optaban por crear versiones específicas para cada tipo de pantalla, una solución costosa y poco eficiente.

La opción más económica y flexible es implementar un diseño que ofrezca una experiencia óptima sin importar el navegador, dispositivo o tamaño de pantalla.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.1 Estrategias del diseño adaptativo

El diseño web adaptativo (responsive web design) es un conjunto de técnicas que permite que el diseño de un sitio web se adapte a las características del dispositivo en el que se visualiza. Las principales herramientas son las *media queries* de CSS3, que permiten aplicar estilos específicos según la resolución de pantalla, y otras técnicas como las cuadrículas flexibles y el contenido adaptativo, ajustando imágenes, vídeos y texto a la pantalla.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.1 Estrategias del diseño adaptativo

Antes del diseño adaptativo, se usaban dos estrategias:

- **El diseño de ancho fijo:**

El diseño de ancho fijo permite mayor control visual, pero presenta problemas en dispositivos con pantallas muy diferentes. Por ejemplo, en pantallas pequeñas, requiere desplazamientos constantes, mientras que en pantallas grandes genera espacios en blanco.

- **el diseño líquido o fluido:**

Por otro lado, el diseño líquido se adapta al ancho de la ventana usando unidades relativas, como porcentajes, permitiendo un mejor uso del espacio, aunque dificulta el control visual y puede crear líneas de texto demasiado largas en pantallas grandes o contenido muy ajustado en pantallas pequeñas.

FIGURA 1.2. Disseny líquid



# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.1 Estrategias del diseño adaptativo

- **El diseño adaptativo:**

Ajusta su composición en función del ancho del navegador, modificando columnas y descartando elementos según el dispositivo. Esto permite una adaptación más precisa a diferentes tamaños de pantalla, ofreciendo una experiencia de usuario óptima en dispositivos de escritorio, móviles y tabletas.

**FIGURA 1.3.** Disseny adaptatiu



# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2 Técnicas de diseño adaptativo

Combina 3 conceptos principales:

- **Parrillas flexibles** para la escritura.
- **Contenido adaptativo** para imágenes, video y texto.
- **Consulta de medios CSS** (media queries) para crear estilos diferentes según la resolución de pantalla del dispositivo del usuario.

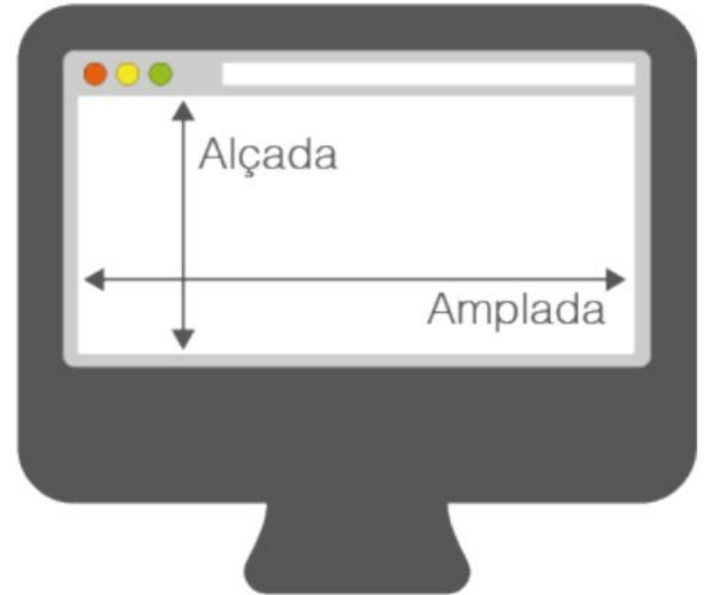
# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.1 Campo de visión

El **campo de visión** o viewport es el área de pantalla en la cual se visualiza la página web. Esta área varía según el dispositivo, y es más pequeña en un dispositivo móvil que en una pantalla de ordenador.

En móviles, este campo suele coincidir con el ancho de pantalla.

La etiqueta meta viewport en HTML permite ajustar la visualización de la página según el dispositivo, mejorando la legibilidad sin necesidad de hacer zoom.

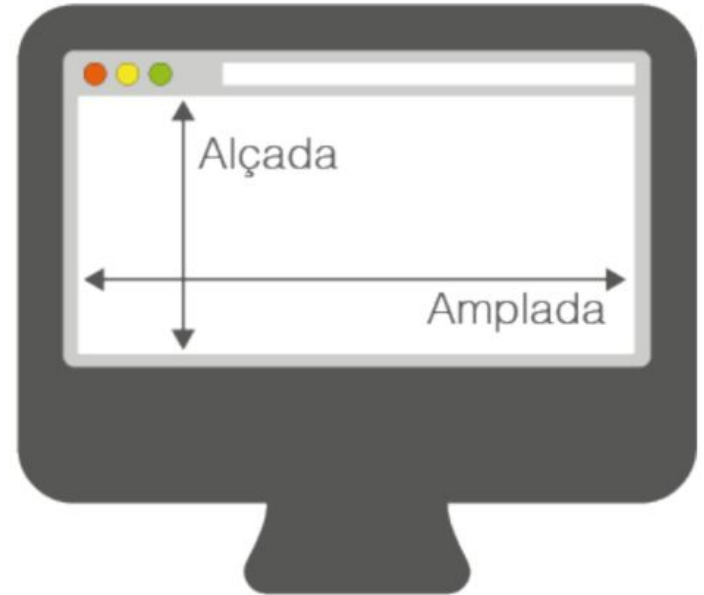


# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.1 Campo de visión

En dispositivos móviles, si no se define el viewport, los navegadores muestran la página como en un equipo de escritorio, escalada para ajustarse a la pantalla, lo que puede dificultar la lectura.

La configuración `width=device-width` indica al navegador que ajuste el contenido al ancho real del dispositivo, adaptándose incluso cuando el dispositivo cambia de orientación (vertical a horizontal).





## 5. Patrones de diseño web adaptativo. Frameworks web

### 1.2.1 Campo de visión

El atributo `initial-scale` define el nivel de zoom al cargar la página. Un valor de 1 muestra la página sin zoom, mientras que valores mayores o menores modifican el tamaño de la visualización.

Otros atributos del viewport incluyen opciones para la altura, el zoom mínimo y máximo, y si el usuario puede ajustar el zoom, lo cual es relevante para la accesibilidad.

| Paràmetre                  | Descripció  |
|----------------------------|---|
| <code>height</code>        | Alçada del camp de visió  |
| <code>minimum-scale</code> | Escala mínima configurable del document   |
| <code>maximum-scale</code> | Escala màxima configurable del document   |
| <code>user-scalable</code> | Possibilitat de zoom. No és recomanable assignar valor <code>no</code> per temes d'accessibilitat, per exemple per a usuaris amb dificultats visuals. |

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

**CSS** incluye el concepto de ***media queries***, que permite asignar **estilos** a una **página** basándose en el **ancho y alto del navegador** del usuario. Esto ayuda a que el sitio web se adapte al dispositivo con el que se está visualizando.

Una ***media query*** le dice al navegador algo como: “Muestra el sitio web en dos columnas, pero si la pantalla es más ancha de 640 píxeles, muéstralo en tres columnas”.

Las media queries **nos permiten ver una apariencia diferente en una tables, un monitor y un móvil.**

Como que también hay muchos tipos de resoluciones en cada uno de los dispositivos mencionados, se ha de intentar que la web se vea bien en todas y cada una de las resoluciones.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estructura:

Ejemplo sencillo de cambio de color dependiendo de la resolución.

```
1 body {  
2     background-color: grey;  
3 }  
4  
5 @media only screen and (min-width: 600px) {  
6     body {  
7         background-color: darkblue;  
8     }  
9 }
```

Las pantallas estrechas tendrán un color gris y las pantallas más anchas tendrán un color azul oscuro.

**Nota:** Es importante recordar que las declaraciones de las hojas de estilo sobrescriben cualquier declaración previa. Por tanto, las media queries se deben de poner después de cualquier declaración de un mismo selector o propiedad.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estructura:

Los diferentes tipos de medios que se pueden especificar son:

**screen**(default): Indicado para la presentación en pantallas de ordenadores no paginados.

**print**: Indicado para un documento que va a ser impreso y visualización de documentos en pantalla en modo vista de impresión.

**speech**: Destinado a sintetizadores de voz para personas con disminución visual.

**all**: Indicado para todos los dispositivos de salida.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

Se pueden redactar consultas con not, and y only.

Las listas separadas por “,” funcionan como un or.

```
1 body {
2   background-color: red;
3   color: black;
4 }
5
6 /* A pantalles amb amplades inferiors de 992px, el color de fons és blau */
7 @media screen and (max-width: 992px) {
8   body {
9     background-color: blue;
10    color: white;
11  }
12 }
13
14 /* A pantalles amb amplades inferiors de 600px, el color de fons és verd oliva */
15 @media screen and (max-width: 600px) {
16   body {
17     background-color: olive;
18     color: white;
19   }
20 }
```

---

```
1 <h1>Si canviem la mida de la finestra, canviarà el color</h1>
2 <p>Per defecte (pantalles més amples de 992 px) el color de fons és vermell. Si l'amplada de la pantalla és inferior a 992 px, el color canviarà a blau. Si és de 600 px o menys, canviarà a verd oliva.</p>
```

| Criteri   | Valor                   | Descripció  |
|---|-------------------------|---|
| width, min-width, max-width   | Llargada                | Per examinar l'amplada de la zona de visualització del navegador.   |
| height, min-height, max-height  | Llargada                | Per examinar l'altura de la zona de visualització del navegador   |
| device-width, min-device-width, max-device-width                      | Llargada                | Per examinar l'amplada física de la pantalla de difusió.  |
| device-height, min-device-height, max-device-height                   | Llargada                | Per examinar l'altura física de la pantalla de difusió.   |
| orientation   | Landscape o portrait    | Per examinar si l'usuari utilitza la tauleta tàctil verticalment ( <i>portrait</i> ) o horitzontalment ( <i>landscape</i> ).                          |
| aspect-ratio, min-aspect-ratio, max-aspect-ratio                      | Ràtio                   | Per examinar el coeficient amplada/alçada.  |
| device-aspect-ratio, min-device-aspect-ratio, max-device-aspect-ratio | Ràtio                   | Per examinar el coeficient físic amplada/alçada de la pantalla.   |
| color, min-color, max-color   |                         | Per examinar si el suport de difusió utilitza el color (valor per defecte en cas que no s'hagi especificat), el blanc i negre o una escala de grisos. |
| color-index, min-color-index, max-color-index                         | Xifra                   | Per examinar el nombre de colors de la taula de colors.   |
| monochrome, min-monochrome, max-monochrome                            | Xifra                   | Per examinar el nombre de nivells de gris per als dispositius monocroms.  |
| resolution, min-resolution, max-resolution                            | DPI                     | Per examinar la resolució de la pantalla de visualització expressada en DPI.  |
| scan  | Progressive o interlace | Per examinar el tipus d'exploració de les pantalles de televisió.   |
| grid  | Xifra                   | Per examinar si la pantalla de difusió utilitza una quadrícula amb una única mida de font.  |

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios



# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

La principal estrategia para representar páginas en diferentes dispositivos es la prueba y el error. Hay algunos cambios de diseños que acostumbran a hacerse cuando se trabaja con ***media queries***.

Cambios en la **disposición de columnas**: La disposición de columnas en un monitor puede ser buena pero en una tablet o un móvil puede no ser adecuado.

**Anchos variables**: Para un telefono movil o una tableta se debe de utilizar un tamaño del 100% con un diseño liquid aprovechando al máximo la anchura de estos dispositivos. Para un dispositivo de escritorio puede interesar un estilo fix. Las media queries pueden cambiar el estilo de fix a liquid según la resolución de pantalla.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

#### Reducción de **espacios en blanco**:

Los **espacios en blanco** entre titulares, imágenes y otros elementos de la página hacen que el diseño respire en un monitor, pero genera un diseño disperso y desaprovechamiento del espacio de una pantalla de móvil. Se deben reducir el margin y el padding con media queries del modelo de caja en móviles.

Ajustes de la **medida de las fuentes**. Los titulares grandes y el texto pequeño queda bien en monitores pero como en las otras ocasiones no van bien en pantallas pequeñas. Vale la pena hacer un poco más grande la letra del cuerpo de la página.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

Cambios en el **menú de navegación**. Los menús de navegación tienen un número elevado de elementos y un diseño recargado. Una de las soluciones más habituales es cambiar a los menús estilo hamburguesa para móviles completando las media queries con un poco de Javascript.

**Ocultación de contenido** en dispositivos móviles. Las media queries CSS permiten esconder contenido superfluo poniendo la propiedad display a none.

Esto puede acarrear 2 posibles problemas:

- Falta de coherencia entre la versión de escritorio y la de móvil.
- El contenido existe aunque no se muestre y ha sido descargado malgastando ancho de banda.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

**imágenes de fondo.** Se pueden cambiar las imágenes de fondo mediante css dependiendo del dispositivo. Un ejemplo sería el siguiente.

```
1      .logo {  
2          width: 960px;  
3          height: 120px;  
4          background-image: url(imatges/logo_gran.png)  
5      }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

#### **‘Mobile first’ o ‘Desktop first’**

Una decisión importante es escoger el dispositivo por defecto.

No es necesario crear tres conjuntos diferenciados de estilos, uno para cada tipo de dispositivo. Se empieza por un diseño por defecto y especificado sin media queries. Después las media queries específicas, sobrescriben este estilo por defecto y reformatear la página para el resto de dispositivos.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

#### ‘Mobile first’ o ‘Desktop first’

**Desktop first.** Es la estrategia más utilizada y consiste en hacer la página para que quede bien en un monitor y después, poner media queries para que quede bien en el resto de dispositivos. Hay diseñadores que suelen utilizar esta estrategia porque es más fácil quitar que añadir contenido.

La mayoría de servidores proporcionan estadísticas sobre las visitas de las plataformas y dispositivos que visitan los sitios web. A partir de estos datos también podemos tomar esta decisión.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

### ‘Mobile first’ o ‘Desktop first’

### Desktop first. Ejemplo:

```
1  /*Regles d'inicialització */
2
3  /* Aquí posaríem els estils per a equips d'escriptori i els estils bàsics
   per a tots els dispositius */
4
5
6  body {
7      /* Aquí, propietats de body */
8  }
9
10 /* pantalles de mida intermitja (tauletes) */
11
12 @media (min-width: 481px) and (max-width:768px) {
13     body {
14         /* propietats específiques per a tauletes */
15     }
16 }
17
18 /* pantalles petites */
19
20 @media (max-width:480px) {
21     body {
22         /* propietats específiques per a telèfons */
23     }
24 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

#### ‘Mobile first’ o ‘Desktop first’

**Mobile first.** Una ventaja de esta estrategia es que el estilo por defecto será más simple y liviano. Normalmente se recomienda esta estrategia, ya que lo más habitual es que los usuarios utilicen sus móviles, pero siempre se ha de verificar con las estadísticas de los proveedores.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.2 Consulta de medios

### Estrategias de diseño:

#### ‘Mobile first’ o ‘Desktop first’

#### Mobile first. Ejemplo

```
1  /*Regles d'inicialització */
2
3  /* Aquí posaríem els estils per a telèfons mòbils i els estils bàsics per
   a tots els dispositius */
4
5  body {
6      /* Aquí, propietats de body */
7  }
8
9  /* pantalles de mida intermitja (tauletes) */
10
11  @media (min-width: 481px) and (max-width:768px) {
12      body {
13          /* propietats específiques per a tauletes */
14      }
15  }
16
17  /* pantalles grans */
18
19  @media (min-width:769px) {
20      body {
21          /* propietats específiques per a equips d'escriptori */
22      }
23  }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.3 Cuadrículas flexibles

Para lograr un **buen diseño adaptable**, es necesario **dejar** de diseñar **estructuras de anchuras fijas especificadas en píxeles**.

Con el poder de las ***media queries***, puede surgir la tentación de diseñar anchuras fijas variables usando ***media queries***, especificando diferentes anchuras en píxeles según el dispositivo. Por ejemplo, 375px para teléfonos móviles, 768px para tabletas y 1000px para equipos de escritorio.

Sin embargo, esto sería cometer el mismo error de nuevo. Existen teléfonos móviles con muchas resoluciones distintas, y es casi imposible encontrar una anchura fija que funcione para todos. Lo mismo ocurre con las tabletas: hay pantallas con muchas anchuras diferentes.

La solución consiste en crear páginas con **anchuras flexibles**.

**Las cuadrículas flexibles juegan un papel clave en el diseño web adaptable.** No son más que un diseño fluido con *media queries* que permite modificar la disposición de algunos elementos y personalizar el diseño según el dispositivo.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.3 Cuadrículas flexibles

### Conversión de diseño de anchura fija a una cuadrícula flexible

Un caso común es tener que convertir un diseño de anchura fija en uno adaptable. En un diseño desde cero, para tener una estructura de cuatro columnas iguales, se puede empezar con una distribución de anchuras de columna del 25%.

Pero supongamos que encontramos un diseño de anchura fija de 960 píxeles. Lo primero que se debe hacer es identificar cuál es el elemento contenedor principal del diseño, que normalmente será un `div`, el elemento semántico `main` o incluso el elemento `body`. Luego, es necesario encontrar la regla CSS que proporciona la anchura fija, que será algo similar a esto:

```
width: 960px;
```

Cambiar esta anchura fija:

```
width: 100%; /*També podem posar-li auto, tan ample com el seu  
contenedor*/
```

**El siguiente paso es convertir todas las anchuras de columnas de píxeles a porcentajes**, teniendo en cuenta la relación en píxeles entre la anchura del elemento convertido y la anchura de su contenedor.

Así, en un diseño de anchura fija de 960 píxeles, tenemos dos columnas:

- Una barra lateral con una anchura de 200 píxeles.
- Una columna principal de contenido con una anchura de 760 píxeles.

El CSS (simplificado) será similar al siguiente:

```
1 .barra_lateral {  
2     float: left;  
3     width: 192px;  
4 }  
5  
6 .principal {  
7     float: left;  
8     width: 768px;  
9 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.3 Cuadrículas flexibles

### Conversión de diseño de anchura fija a una cuadrícula flexible

**Evidentemente, en el CSS habrá muchas más propiedades relacionadas con bordes, colores de fondo y otros aspectos, pero ahora nos centramos en la anchura de los elementos.**

Lo primero que debemos hacer es convertir la anchura fija de la barra lateral. A partir de su anchura, 200 píxeles, se divide por la anchura de su contenedor, 960px. El resultado (0,20) multiplicado por 100 nos da la anchura, en porcentaje, de la barra lateral (20%). De la misma manera, la anchura fija de 760 píxeles de la zona principal de contenido nos dará, al dividirla por 960, una anchura del 80%.

El diseño fluido quedará de la siguiente manera:

```
1 .barra_lateral {  
2   float: left;  
3   width: 20%;  
4 }  
5  
6 .principal {  
7   float: left;  
8   width: 80%;  
9 }
```

En caso de obtener resultados con decimales, no se debe redondear, ya que esto podría exceder ligeramente la anchura disponible (la suma de las anchuras de las columnas puede superar el 100%) y provocar que algunas columnas "caigan" del diseño a la fila siguiente. Los navegadores pueden trabajar perfectamente con anchuras relativas expresadas en porcentajes de dos o más decimales.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.4 Contenidos multimedia adaptativos

Aunque las cuadrículas flexibles se adaptan sin problema a todo tipo de dispositivos, los contenidos multimedia como imágenes y vídeos incrustados en estas cuadrículas pueden generar problemas, ya que sus dimensiones fijas pueden desbordar los límites de las columnas en las que se encuentran. Para evitar este tipo de problemas, existen algunas propiedades de CSS que pueden ser útiles:

- **Propiedad max-width.** Una regla importante para lograr que las imágenes y vídeos sean adaptativos es utilizar la propiedad `max-width` con un valor de `100%`, de modo que las imágenes se redimensionen y queden siempre ajustadas dentro de su contenedor.

```
1  img, video {  
2    max-width: 100%;  
3    height: auto;  
4  }
```

Con esta regla, las imágenes de la página web se escalan sin problemas para ajustarse a la anchura y altura de la columna en la que están ubicadas. Sin embargo, al usar `max-width` en lugar de `width`, se evita que crezcan por encima de su tamaño original.

- **Eliminar los atributos height y width de todos los elementos img.** Es decir, transformar el HTML eliminando todas las especificaciones explícitas de altura y anchura.

```
1  
```

```
1  
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.4 Contenidos multimedia adaptativos

**Utilizar la propiedad background-size** para lograr que las imágenes de fondo se redimensionen y escalen. Para ello, se pueden usar los tres valores principales de esta propiedad.

**TAULA 1.4.** Valors de la propietat 'background-size'

| Valor     | Descripció   |
|-----------|--|
| contain   | La imatge de fons s'escala per intentar encaixar l'àrea de contingut, mantenint la seva proporció.   |
| 100% 100% | La imatge de fons s'estira fins a cobrir tota l'àrea de contingut. Perd la seva proporció si aquesta àrea és de proporcions diferents.                                 |
| cover     | La imatge s'escala per cobrir tota l'àrea de contingut, però sense perdre la seva proporció. Si l'àrea de contingut té una altra proporció, la imatge queda retallada. |

VER EJEMPLO

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.4 Contenidos multimedia adaptativos

### Especificar diferentes imágenes según el dispositivo

Si en dispositivos pequeños es necesario redimensionar las imágenes grandes, ¿por qué desperdiciar ancho de banda y tiempo descargando esas imágenes tan grandes en teléfonos móviles?

Con las *media queries*, se pueden mostrar diferentes imágenes según el dispositivo.

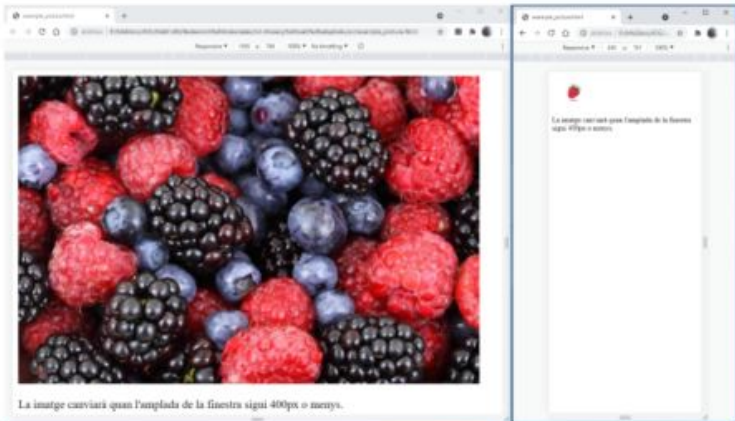
```
1  /* Per a dispositius de menys de 400px */
2  body {
3      background-image: url('imatge_petita.jpg');
4  }
5
6  /* Per a pantalles de més de 400px */
7  @media only screen and (min-width: 400px) {
8      body {
9          background-image: url('imatge_gran.jpg');
10     }
11 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 1.2.4 Contenidos multimedia adaptativos

### Especificar diferentes imágenes según el dispositivo

Otra opción es utilizar el elemento HTML `picture`, que permite especificar diferentes fuentes de imágenes y cuenta con un atributo `media` para indicar el dispositivo al que corresponde cada fuente.



```
1 <body>
2
3 <picture>
4   <source srcset="https://i.ibb.co/10Mk3dJ/frambuesa-petita.png"
5     media="(max-width: 400px)">
6   <source srcset="https://i.ibb.co/QNKr348/berries-2277-640.jpg"
7     >
8   
10 </picture>
11
12 <p>La imatge canviarà quan l'amplada de la finestra sigui de 400
13   px o menys.</p>
14
15 </body>
```

Código: <https://codepen.io/iocdawm9/pen/NWjepNR>.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2 Modelos de diseño CSS

### Especificar diferentes imágenes según el dispositivo

Existen tres modelos de diseño CSS que se utilizan habitualmente:

- las cuadrículas flexibles CSS.
- el flexbox CSS.
- el grid CSS.

A partir de estos elementos, se establecen patrones de diseño adaptativo para poder visualizar la web en cualquier dispositivo.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1 Cuadrículas flexibles CSS

En el diseño adaptable, las cuadrículas flexibles son un mecanismo para estructurar de forma adaptativa el diseño del sitio web. Esto constituye la base del diseño adaptable del sitio.

Las cuadrículas adaptativas son muy similares al diseño fluido. Utilizan tamaños relativos, pero añaden consultas de medios (*media queries*) cuando es necesario para personalizar aún más la estructura para algún dispositivo específico.

Existen muchos sistemas de cuadrículas disponibles en la red, creados tanto por diseñadores web a título personal como por grandes compañías, como Twitter. Todos estos sistemas emplean un enfoque muy similar a la hora de estructurar el HTML de la cuadrícula, basándose en una serie de `div` anidados de tres tipos de elementos diferentes:

- **Contenedores.** Un contenedor está compuesto por una o más filas. Ayuda a establecer la anchura de la cuadrícula y normalmente utiliza la propiedad `max-width` para evitar que el contenido se expanda innecesariamente en las pantallas más grandes. Los contenedores también suelen estar centrados en la ventana del navegador.
- **Filas.** Son otro elemento `div` ubicado dentro del contenedor. Incluyen en su interior elementos que representan las columnas de la fila en cuestión.
- **Columnas.** Una columna es un elemento de tipo bloque dentro de una fila. Normalmente es un `div`, pero también pueden ser elementos semánticos como `header`, `footer`, `section`, `article`. Cada fila tiene una o más columnas, con una anchura determinada.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Para crear una cuadrícula flexible, lo primero que se debe hacer es una **división inicial del espacio disponible en un número de columnas base**. Esta será la estructura base de la cuadrícula a partir de la cual se deriva la distribución del diseño.

Las cuadrículas flexibles suelen utilizar una estructura inicial de **doce columnas** para dividir el ancho total de la pantalla. Esta cifra no es arbitraria; el hecho de que el 12 tenga **muchos divisores** (2, 3, 4, 6) facilita la posterior distribución del espacio en bloques del mismo tamaño.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Veamos cómo funciona esta cuadrícula con un ejemplo concreto. Se trata de un sitio web con una cabecera, un bloque lateral con botones de acceso a otras secciones, una barra lateral derecha con publicidad u otros elementos, y un pie de página.



La división inicial en doce columnas ya ofrece una estructura base en la que encajar los diferentes elementos del diseño (ver figura 2.2). Estos elementos también definirán las filas de la cuadrícula.

Para implementar esta cuadrícula, primero es necesario asegurarse de que el *padding* y el *border* se incluyan en el ancho y alto total de los elementos. Esto facilitará mucho los cálculos de las dimensiones de la cuadrícula. Recuerda que la propiedad `box-sizing` permite lograr esto.

```
1 * {  
2   box-sizing: border-box;  
3 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Después, es necesario crear clases para poder generar, dentro de las filas de la cuadrícula, columnas de diferentes anchuras.

Para que la cuadrícula sea flexible, se necesitarán **porcentajes** para **dimensionar las distintas anchuras**.

Creamos una clase `col-X` para cada posible anchura: así tendremos `col-1`, `col-2`, ..., `col-12` para poder tener elementos de anchura de 1, 2, ..., hasta 12 columnas. Con doce columnas y un ancho total de la pantalla que representa el 100%, cada columna tendrá un ancho de  $100\% / 12 = 8.33\%$ .

```
1 .col-1 {width: 8.33%;}
2 .col-2 {width: 16.66%;}
3 .col-3 {width: 25%;}
4 .col-4 {width: 33.33%;}
5 .col-5 {width: 41.66%;}
6 .col-6 {width: 50%;}
7 .col-7 {width: 58.33%;}
8 .col-8 {width: 66.66%;}
9 .col-9 {width: 75%;}
10 .col-10 {width: 83.33%;}
11 .col-11 {width: 91.66%;}
12 .col-12 {width: 100%;}
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Haremos que estas columnas, dentro de una fila, floten (*float*) hacia la izquierda, es decir, se posicionen inmediatamente en la posición izquierda de su elemento adyacente.

También añadiremos un poco de espacio interior o relleno (*padding*) de 15 píxeles para el contenido de los elementos.

```
1 [class*="col-"] {  
2   float: left;  
3   padding: 15px;  
4 }
```

Para crear el concepto de fila, podemos utilizar un `<div>` con la clase `fila` que contendrá las columnas, que también serán elementos `<div>`, pero de las correspondientes clases `col-x` creadas anteriormente.

```
1 <div class="fila">  
2   <div class="col-3">...</div> <!-- Columna 25% d'amplada -->  
3   <div class="col-9">...</div> <!-- Columna 75% d'amplada -->  
4 </div>
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

El número de columnas en una fila siempre debe ser 12; de lo contrario, quedarían espacios vacíos en el diseño. Además de `divs`, también se pueden utilizar etiquetas semánticas como `header`, `nav`, `section`, `article`, `aside` o `footer`, que en la práctica se comportan igual que un `div`.

```
1 <div class="fila">
2   <nav class="col-3">...</div> <!-- Menú de navegació a una
      columna de 25% d'amplada -->
3   <section class="col-9">...</div> <!-- Secció de contingut a
      una segona columna de 75% d'amplada -->
4 </div>
```

Así, todas las columnas de una fila siempre flotan a la izquierda. Esto puede provocar que elementos posteriores se posicionen en los huecos que dejan las columnas al flotar.

Para evitarlo, es necesario añadir después de cada fila un `clear: both` para asegurarse de que ningún elemento posterior a las columnas de una fila "suba" a ocupar los huecos de las columnas flotantes. Aquí puede ser útil el pseudoelemento `::after`, que permite definir una regla para el espacio final del elemento con la clase `.fila`.

```
1 .fila::after {
2   content: "";
3   clear: both;
4   display: table;
5 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

```
1 .fila::after {  
2   content: "";  
3   clear: both;  
4   display: table;  
5 }
```

Finalmente le damos un poco de forma a la cuadrícula

```
1 html {  
2   font-family: "Lucida Sans", sans-serif;  
3 }  
4  
5 header, footer {  
6   background-color: #9933cc;  
7   color: #ffffff;  
8   padding: 15px;  
9 }  
10  
11 nav ul {  
12   list-style-type: none;  
13   margin: 0;  
14   padding: 0;  
15 }  
16  
17 nav li {  
18   padding: 8px;  
19   margin-bottom: 7px;  
20   background-color: #33b5e5;  
21   color: #ffffff;  
22   box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);  
23 }  
24  
25 nav li:hover {  
26   background-color: #0099cc;  
27 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

```
1 * {
2     box-sizing: border-box;
3 }
4
5 .fila::after {
6     content: "";
7     clear: both;
8     display: table;
9 }
10
11 [class*="col-"] {
12     float: left;
13     padding: 15px;
14 }
15
16 .col-1 {
17     width: 8.33%;
18 }
19
20 .col-2 {
21     width: 16.66%;
22 }
23
24 .col-3 {
25     width: 25%;
26 }
27
```

```
.col-4 {
    width: 33.33%;
}

.col-5 {
    width: 41.66%;
}

.col-6 {
    width: 50%;
}

.col-7 {
    width: 58.33%;
}

.col-8 {
    width: 66.66%;
}

.col-9 {
    width: 75%;
}

.col-10 {
    width: 83.33%;
}

.col-11 {
    width: 91.66%;
}

.col-12 {
    width: 100%;
}

html {
    font-family: "Lucida Sans", sans-serif;
}

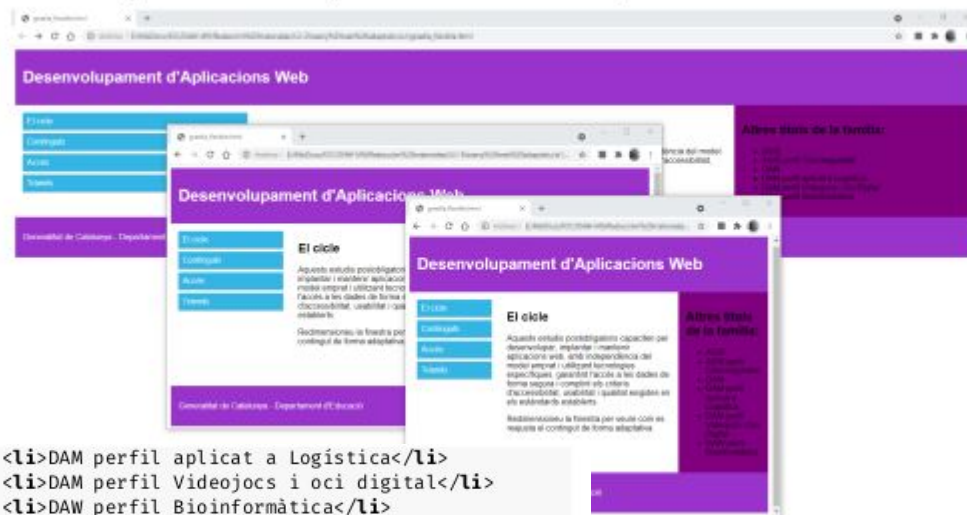
header, footer {
    background-color: #9933cc;
    color: #ffffff;
    padding: 15px;
}
```

```
72 }
73
74 nav ul {
75     list-style-type: none;
76     margin: 0;
77     padding: 0;
78 }
79
80 nav li {
81     padding: 8px;
82     margin-bottom: 7px;
83     background-color: #33b5e5;
84     color: #ffffff;
85     box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px
86         rgba(0, 0, 0, 0.24);
87 }
88
89 nav li:hover {
90     background-color: #0099cc;
91 }
92
93 aside {
94     background-color: purple;
95     height: 100%;
96 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

FIGURA 2.3. Visualització de la graella flexible amb diferents amplades de finestra



```
1 <body>
2 <div class="fila">
3   <header class="col-12">
4     <h1>Desenvolupament d'aplicacions web</h1>
5   </header>
6 </div>
7 <div class="fila">
8   <nav class="col-3">
9     <ul>
10       <li>El cicle</li>
11       <li>Continguts</li>
12       <li>Accés</li>
13       <li>Tràmits</li>
14     </ul>
15   </nav>
16   <section class="col-6">
17     <h2>El cicle</h2>
18     <p>Aquests estudis postobligatoris capaciten per
19       desenvolupar, implantar i mantenir aplicacions
20       web, amb
21       independència del model emprat i utilitzant
22       tecnologies
23       específiques, garantint l'accés a les dades de
24       forma segura
25       i complint els criteris d'accessibilitat,
26       usabilitat i qualitat
27       exigits en els estàndards establerts.</p>
28     <p>Redimensioneu la finestra per veure com es
29       reajusta el contingut de forma adaptativa.</p>
30   </section>
31   <aside class="col-3">
32     <h3>Altres títols de la família:</h3>
33     <ul>
34       <li>ASIX</li>
35       <li>ASIX perfil Ciberseguretat</li>
36       <li>DAM</li>
```

```
33 <li>DAM perfil aplicat a Logística</li>
34 <li>DAM perfil Videojocs i oci digital</li>
35 <li>DAW perfil Bioinformàtica</li>
36 </ul>
37 </aside>
38 </div>
39 <div class="fila">
40   <footer class="col-12">
41     <p>Generalitat de Catalunya - Departament d'Educació</p>
42   </footer>
43 </div>
44 </body>
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

De momento hemos creado una cuadrícula fluida que se adapta al ancho de la ventana donde se visualiza. Pero aún no es adaptable.

Es necesario personalizar la estructura con *media queries* para que se adapte a las características del dispositivo. Para ello, se deben añadir los puntos de corte (*breakpoints*) adecuados para establecer los tres tipos básicos de dispositivos: escritorio, tableta y teléfono móvil.

Primero, añadimos la *media query* para teléfonos móviles. Para estos dispositivos, establecemos que todas las columnas tengan un ancho del 100%, es decir, que la estructura se simplifique.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Con esta *media query* para teléfonos móviles, usamos una estrategia **Desktop First**, para tener una versión por defecto de la cuadrícula para equipos de escritorio. El CSS con las nuevas clases sería similar al siguiente:

```
1  /* Versió equip d'escriptori */
2
3  .col-1 {width: 8.33%;}
4  .col-2 {width: 16.66%;}
5  .col-3 {width: 25%;}
6  .col-4 {width: 33.33%;}
7  .col-5 {width: 41.66%;}
8  .col-6 {width: 50%;}
9  .col-7 {width: 58.33%;}
10 .col-8 {width: 66.66%;}
11 .col-9 {width: 75%;}
12 .col-10 {width: 83.33%;}
13 .col-11 {width: 91.66%;}
14 .col-12 {width: 100%;}
15
16 @media only screen and (max-width: 768px) {
17   /* Per a telèfons mòbils */
18   [class*="col-"] {
19     width: 100%;
20   }
21 }
```

Así, establecemos que para cualquier dispositivo con un ancho inferior a 768px, todas las columnas de la cuadrícula (las clases `col-1`, `col-2`, ... hasta `col-12`) tendrán un ancho del 100%.

Esto ya supone una diferencia importante respecto al diseño fluido, ya que se ha modificado la distribución de los elementos para algunos dispositivos específicos.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Sin embargo, hay algunas ventajas en diseñar primero para dispositivos móviles. Por este motivo, modificamos el CSS y usamos la estrategia *Mobile First*, poniendo al principio de la hoja de estilos las reglas para teléfonos móviles y, en la *media query*, las reglas CSS específicas para equipos de escritorio.

```
1  /* Per a telèfons mòbils */
2  [class*="col-"] {
3    width: 100%;
4  }
5
6  @media only screen and (min-width: 768px) {
7    /* Versió equip d'escriptori */
8    .col-1 {width: 8.33%;}
9    .col-2 {width: 16.66%;}
10   .col-3 {width: 25%;}
11   .col-4 {width: 33.33%;}
12   .col-5 {width: 41.66%;}
13   .col-6 {width: 50%;}
14   .col-7 {width: 58.33%;}
15   .col-8 {width: 66.66%;}
16   .col-9 {width: 75%;}
17   .col-10 {width: 83.33%;}
18   .col-11 {width: 91.66%;}
19   .col-12 {width: 100%;}
20 }
```

Finalmente, añadimos otra *media query* al archivo de estilos para las tabletas. Establecemos un nuevo punto de corte (*breakpoint*) para adaptar la distribución de los elementos del diseño según el tipo de dispositivo.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Para la versión para tabletas, mantenemos el menú lateral y la zona de contenido, pero eliminamos la sección lateral derecha (el elemento `aside`) para dar mayor legibilidad al contenido.

En este caso, la versión para tabletas no es tan sencilla de implementar como la de teléfono móvil. La solución consiste en crear unas nuevas clases específicas para este dispositivo y, aprovechando que HTML permite asignar múltiples clases a un elemento, asignamos a cada elemento del diseño las clases tanto para equipos de escritorio como para tabletas. La versión para móvil es la más sencilla, ya que todas las columnas tienen el mismo ancho y se pueden usar las columnas definidas para la versión de escritorio.

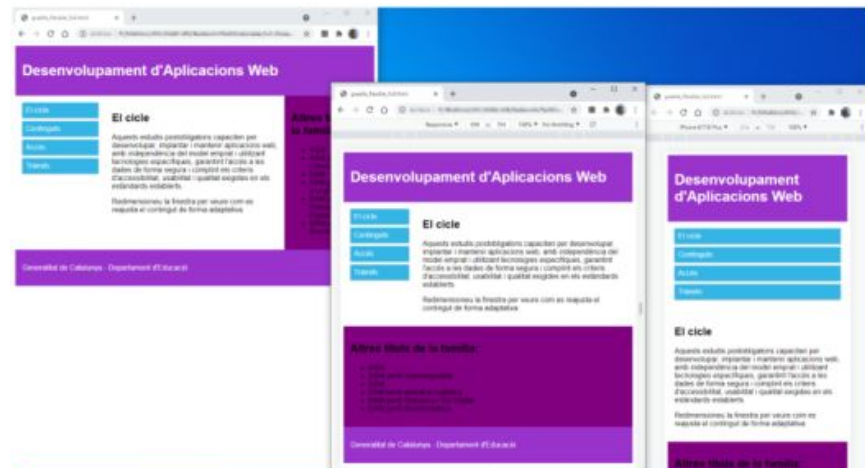
### Version movil

```
6 @media only screen and (min-width: 576px) {
7   /* Versió tauletes */
8   .col-s-1 {width: 8.33%;}
9   .col-s-2 {width: 16.66%;}
10  .col-s-3 {width: 25%;}
11  .col-s-4 {width: 33.33%;}
12  .col-s-5 {width: 41.66%;}
13  .col-s-6 {width: 50%;}
14  .col-s-7 {width: 58.33%;}
15  .col-s-8 {width: 66.66%;}
16  .col-s-9 {width: 75%;}
17  .col-s-10 {width: 83.33%;}
18  .col-s-11 {width: 91.66%;}
19  .col-s-12 {width: 100%;}
20 }
```

Les noves files de la graella flexible tenen ara l'aspecte següent:

```
1 <div class="fila">
2   <div class="col-3 col-s-3">...</div>
3   <div class="col-6 col-s-9">...</div>
4   <div class="col-3 col-s-12">...</div>
5 </div>
```

### Versión Pantalla



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.1.1 Implementación

Para la versión para tabletas, mantenemos el menú lateral y la zona de contenido, pero eliminamos la sección lateral derecha (el elemento **aside**) para dar mayor legibilidad al contenido.

En este caso, la versión para tabletas no es tan sencilla de implementar como la de teléfono móvil. La solución consiste en crear unas nuevas clases específicas para este dispositivo y, aprovechando que HTML permite asignar múltiples clases a un elemento, asignamos a cada elemento del diseño las clases tanto para equipos de escritorio como para tabletas. La versión para móvil es la más sencilla, ya que todas las columnas tienen el mismo ancho y se pueden usar las columnas definidas para la versión de escritorio.

### Version movil

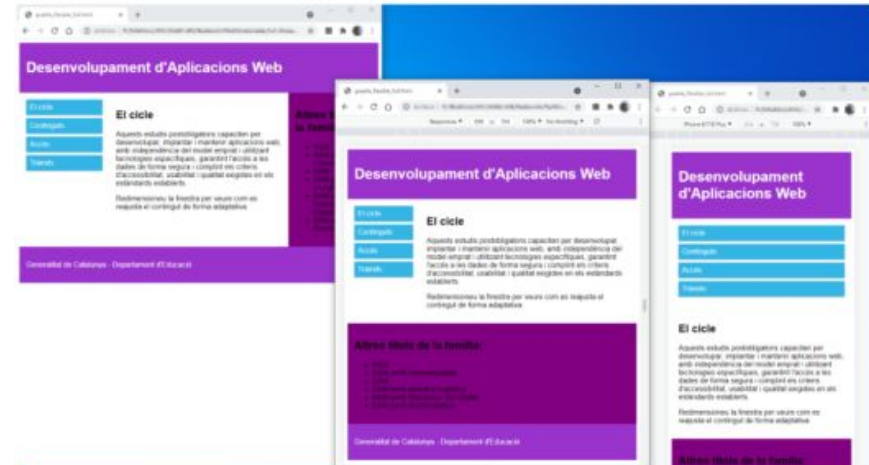
```
6 @media only screen and (min-width: 576px) {
7   /* Versió tauletes */
8   .col-s-1 {width: 8.33%;}
9   .col-s-2 {width: 16.66%;}
10  .col-s-3 {width: 25%;}
11  .col-s-4 {width: 33.33%;}
12  .col-s-5 {width: 41.66%;}
13  .col-s-6 {width: 50%;}
14  .col-s-7 {width: 58.33%;}
15  .col-s-8 {width: 66.66%;}
16  .col-s-9 {width: 75%;}
17  .col-s-10 {width: 83.33%;}
18  .col-s-11 {width: 91.66%;}
19  .col-s-12 {width: 100%;}
20 }
```

Les noves files de la graella flexible tenen ara l'aspecte següent:

```
1 <div class="fila">
2   <div class="col-3 col-s-3">...</div>
3   <div class="col-6 col-s-9">...</div>
4   <div class="col-3 col-s-12">...</div>
5 </div>
```

<https://codepen.io/iocdawm9/pen/gOWQdRM>

### Versión Pantalla



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

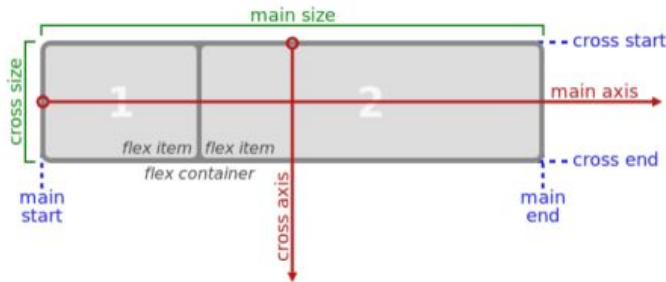
### Introducción a Flexbox

- **¿Qué es Flexbox?**
  - Flexbox es un módulo de diseño CSS3 que mejora la capacidad de creación de diseños adaptativos.
  - Permite un **posicionamiento flexible** de los elementos en la página, facilitando la adaptación a diferentes tamaños de pantalla.
- **Ventajas de Flexbox:**
  - Simplifica el diseño de **interfaces responsivas**.
  - Reemplaza la necesidad de usar **float** y **inline-block**.
  - Facilita el alineamiento y distribución del espacio entre los elementos.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Modelo de Flexbox:



- Compuesto por un **contenedor padre** (flex container) y **elementos hijos** (flex items).
  - El **flex container** es el elemento padre que contiene todos los flex items. Indica la posición de sus hijos. Se debe indicar que un elemento es un flex container con el valor 'flex' en la propiedad display.
  - El **flex item** es el elemento que se encuentra dentro del flex container. Se comporta automáticamente según lo definido en su elemento padre.
- **Ejes del Flexbox:**
  - El **main axis** es el eje principal en sentido horizontal en el cual se van posicionando los flex items. Su dirección por defecto es de izquierda a derecha (horizontal), pero se puede cambiar con la propiedad flex-direction.
  - El **cross axis** es el eje transversal, perpendicular al eje principal. Su dirección por defecto es de arriba a abajo, en sentido vertical, pero de la misma manera que el main axis, se puede modificar con la propiedad flex-direction.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Modelo de Caja Flexible

EJ//

```
<div class="flex-container">
  <div class="flex-item">Elemento 1</div>
  <div class="flex-item">Elemento 2</div>
  <div class="flex-item">Elemento 3</div>
</div>
```

```
.flex-container {
  display: flex;
}
.flex-item {
  padding: 10px;
  border: 1px solid #ccc;
}
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

**display:** Define el contenedor como un flex container.

```
.flex-container {  
  display: flex;  
}
```



**inline-flex:** Define el contenedor como un flex container, pero el contenedor en sí se comporta como un elemento en línea. Esto significa que el contenedor se mantiene dentro del flujo del documento como si fuera un elemento en línea, pero los elementos dentro de él se distribuyen utilizando Flexbox.

```
.flex-container {  
  display: inline-flex;  
}
```



**Cuándo usarlo:** Es útil cuando se necesita que el contenedor sea tratado como un elemento en línea, pero se desea que los elementos dentro de él se dispongan como un contenedor flexible, por ejemplo, para alinear botones dentro de una línea de texto.

**flex-direction:** Especifica la dirección del eje principal.

- Valores: **row**, **row-reverse**, **column**, **column-reverse**.

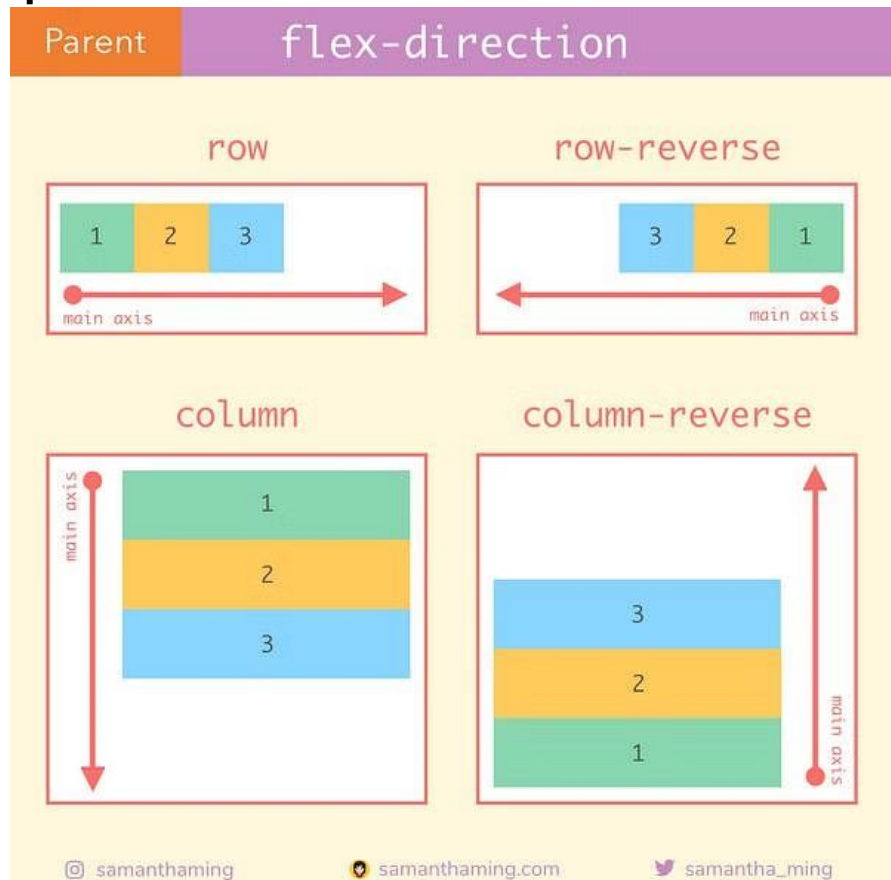
```
.flex-container {  
  flex-direction: row-reverse;  
}
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

- **row:** Organiza los elementos horizontalmente de izquierda a derecha (dirección por defecto).
- **row-reverse:** Organiza los elementos horizontalmente, pero de derecha a izquierda.
- **column:** Organiza los elementos verticalmente, de arriba a abajo.
- **column-reverse:** Organiza los elementos verticalmente, pero de abajo a arriba.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

**Ejercicio:** Crea un contenedor con tres elementos y cambia el **flex-direction** para ver cómo afecta a la distribución de los elementos.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

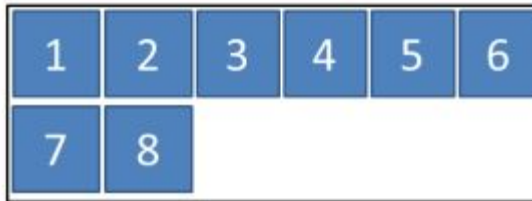
#### Propiedades del Contenedor Flex (II)

- **flex-wrap**: Permite distribuir los elementos en una o varias líneas.
  - **nowrap**: Todos los elementos permanecen en una sola línea, sin importar si hay suficiente espacio o no.
  - **wrap**: Los elementos se dividen en varias líneas si es necesario, permitiendo que se ajusten según el tamaño del contenedor.
  - **wrap-reverse**: Los elementos también se dividen en varias líneas si es necesario, pero el orden de las líneas se invierte (la última línea se coloca primero).

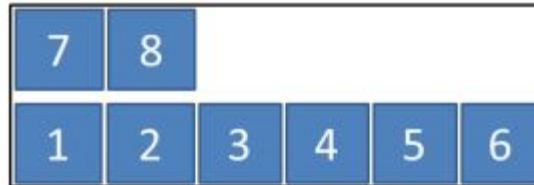
Se pueden ver ejemplos en la siguiente url <https://codepen.io/team/css-tricks/pen/1ea1ef35d942d0041b0467b4d39888d3>.



flex-wrap: nowrap



flex-wrap: wrap



flex-wrap: wrap-reverse

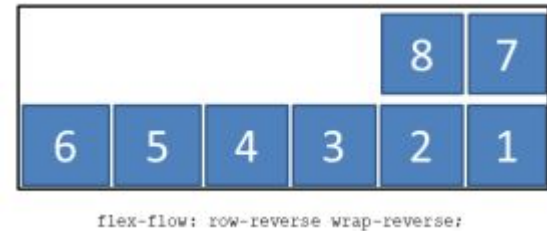
# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

- **flex-flow:** Es una combinación abreviada de flex-direction y flex-wrap. Su valor por defecto es row no-wrap.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

##### **justify-content:**

La propiedad **justify-content** define la alineación de los elementos flexibles a lo largo del main-axis horizontal. Ayuda a distribuir el espacio libre que sobra cuando todos los elementos 'flex' de una línea son inflexibles o son flexibles pero han alcanzado su tamaño máximo. También ayuda a controlar los elementos cuando han desbordado la línea.

- **flex-start:** Los elementos se alinean al inicio del contenedor a lo largo del eje principal.
- **center:** Los elementos se alinean al centro del contenedor a lo largo del eje principal.
- **flex-end:** Los elementos se alinean al final del contenedor a lo largo del eje principal.
- **space-between:** Los elementos se distribuyen uniformemente con el mayor espacio posible entre ellos.
- **space-around:** Los elementos tienen un espacio igual alrededor de cada uno.
- **space-evenly:** Los elementos tienen un espacio igual entre ellos y entre los bordes del contenedor.

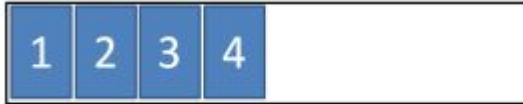
# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

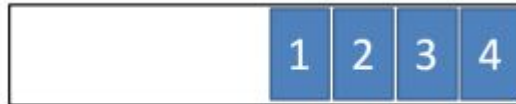
### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

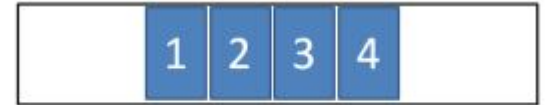
**justify-content:**



`justify-content: flex-start;`



`justify-content: flex-end;`



`justify-content: center;`



`justify-content: space-around;`



`justify-content: space-between;`



`justify-content: space-evenly;`

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

##### **align-items:**

La propiedad **align-items** define el comportamiento por defecto de los elementos flexibles respecto al cross-axis (vertical). Es algo similar al tipo de alineación vertical de los elementos 'flex' dentro del contenedor.

- **flex-start:** Los elementos se alinean al inicio del eje transversal (cross axis).
- **center:** Los elementos se alinean al centro del eje transversal.
- **flex-end:** Los elementos se alinean al final del eje transversal.
- **stretch:** Los elementos se estiran para llenar el contenedor si no se especifica un tamaño.



# 5. Patrones de diseño web adaptativo. Frameworks web

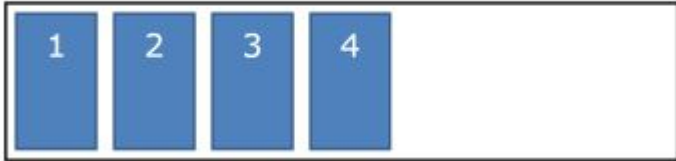
## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

**justify-content:**

**FIGURA 2.22.** Visualització align-items: stretch;



**FIGURA 2.24.** Visualització align-items: flex-end;



**FIGURA 2.26.** Visualització align-items: baseline;



**FIGURA 2.23.** Visualització align-items: flex-start;



**FIGURA 2.25.** Visualització align-items: center;



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

##### **align-content:**

La propiedad **align-content** permite controlar la alineación de las líneas a lo largo del cross-axis cuando los elementos no utilizan todo el espacio disponible. Solo tiene sentido en contenedores multilínea, con la propiedad **flex-wrap** con el valor **wrap** o **wrap-reverse**.

- **flex-start:** Las líneas de elementos se alinean al inicio del contenedor (cuando hay varias líneas).
- **center:** Las líneas de elementos se alinean al centro del contenedor.
- **flex-end:** Las líneas de elementos se alinean al final del contenedor.
- **space-between:** Las líneas se distribuyen uniformemente con el mayor espacio posible entre ellas.
- **space-around:** Las líneas tienen un espacio igual alrededor de cada una.
- **stretch:** Las líneas se estiran para llenar el contenedor.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

**align-content:**

FIGURA 2.27. Visualització align-content: space-between;

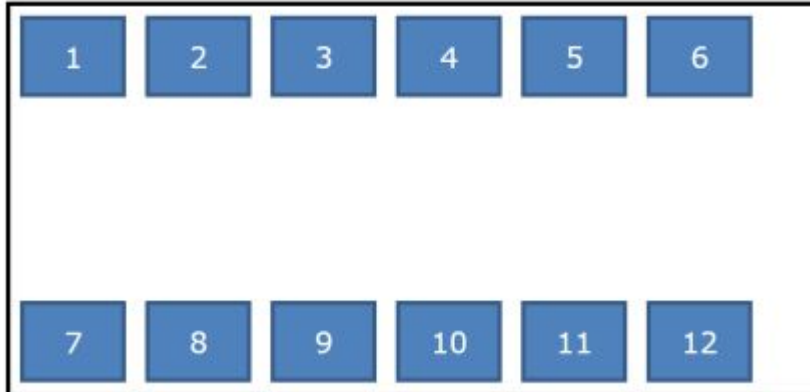


FIGURA 2.28. Visualització align-content: space-around;

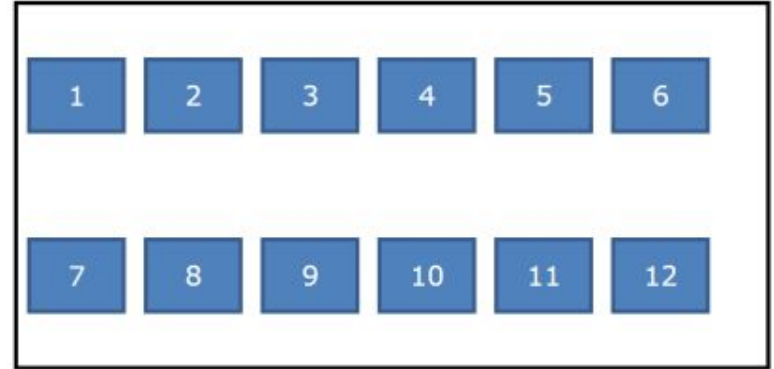
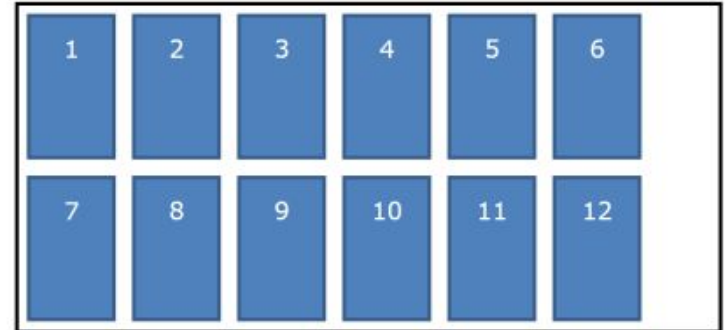


FIGURA 2.29. Visualització align-content: stretch;



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### Propiedades del Contenedor Flex (II)

**align-content:**

FIGURA 2.30. Visualització align-content: center;

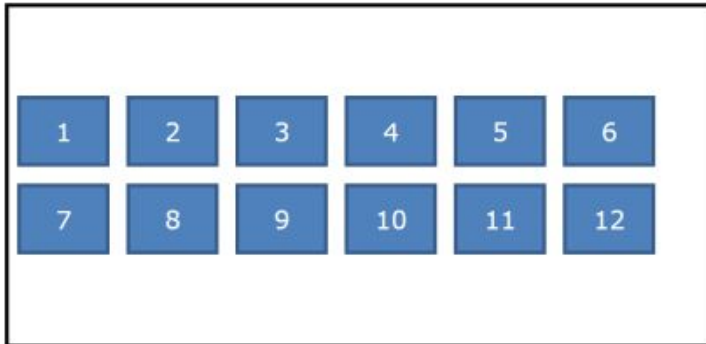


FIGURA 2.31. Visualització align-content: flex-start;

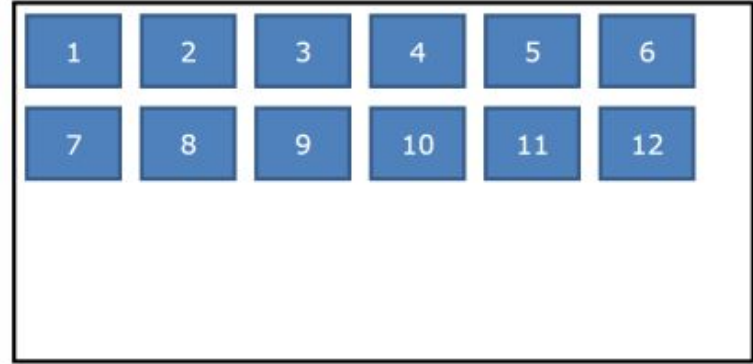
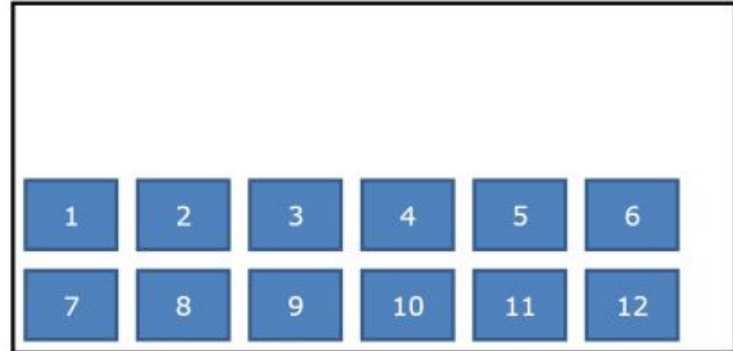


FIGURA 2.32. Visualització align-content: flex-end;



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

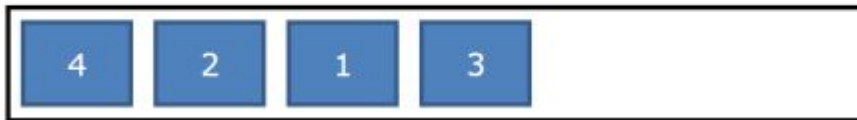
### Elementos 'flex'

Los elementos que son hijos directos de un contenedor 'flex' automáticamente se convierten en elementos flexibles.

### Propiedad 'order'

Permite controlar el orden de los flex items que contiene el flex container. El orden por defecto es el que determina el código fuente. El valor por defecto es 0.

FIGURA 2.33. Visualització de canvi d'ordre



### Exemple de la propietat 'order' dels elements '

```
1 .flex-container {  
2   display: flex;  
3   align-items: stretch;  
4   background-color: white;  
5   border: black solid 5px;  
6 }  
7  
8 .flex-container > div {  
9   color: #fff;  
10  font-family: Verdana;  
11  background-color: #4F81BD;  
12  border: #385DBA solid 5px;  
13  width: 100px;  
14  margin: 10px;  
15  text-align: center;  
16  line-height: 75px;  
17  font-size: 30px;  
18 }
```

```
1 <div class="flex-container">  
2   <div style="order: 3">1</div>  
3   <div style="order: 2">2</div>  
4   <div style="order: 4">3</div>  
5   <div style="order: 1">4</div>  
6 </div>
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

#### Propiedad 'flex-grow'

Determina cuanto puede crecer un flex item en relación con el resto de flex items. El valor por defecto es 0.

```
1 <div class="flex-container">
2   <div style="flex-grow: 1">1</div>
3   <div style="flex-grow: 2">2</div>
4   <div style="flex-grow: 8">3</div>
5 </div>
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

#### Propiedad 'flex-shrink'

Indica cuanto se puede encoger un flex item. El valor por defecto es 1.

```
1  <div class="flex-container">
2    <div>1</div>
3    <div>2</div>
4    <div>3</div>
5    <div>4</div>
6    <div style="flex-shrink: 0">5</div>
7    <div>6</div>
8    <div>7</div>
9    <div>8</div>
10   <div>9</div>
11   <div>10</div>
12 </div>
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

#### Propiedad 'flex-basis'

Especifica la medida por defecto de un elemento flex antes que el espacio sea repartido. Puede ser una longitud (20%, 5rem, etc) o una palabra clave como **auto** o **content** que fijará la medida del elemento en función del contenido. Esta funcionalidad aún está en fase experimental.

```
1 <div class="flex-container">
2   <div>1</div>
3   <div style="flex-basis:20rem">2</div>
4   <div style="flex-basis:200px">3</div>
5 </div>
```





# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

#### Propiedad 'flex'

Versión abreviada de las propiedades flex-grow, flex-shrink y flex-basis.

Su valor por defecto es 0 1 auto.

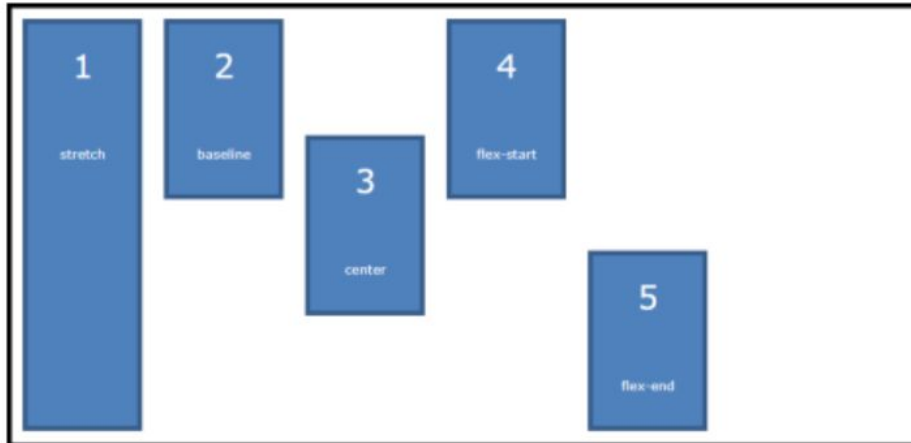
# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

#### Propiedad 'align-self'

Permite sobrecribir la alineación por defecto ( o la especificada por align-items) para un elemento flex concreto.



```
1 <div class="flex-container">
2   <div style="align-self: stretch">1 <br/> <span style="font-size:0.4em">
   stretch</span></div>
3   <div style="align-self: baseline">2<br/> <span style="font-size:0.4em">
   baseline</span></div>
4   <div style="align-self: center">3<br/> <span style="font-size:0.4em">
   center</span></div>
5   <div style="align-self: flex-start">4<br/> <span style="font-size:0.4em">
   >flex-start</span></div>
6   <div style="align-self: flex-end">5<br/> <span style="font-size:0.4em">
   flex-end</span></div>
7 </div>
```

```
1 .flex-container {
2   display: flex;
3   align-items: stretch;
4   height: 400px;
5   background-color: white;
6   border: black solid 5px;
7 }
8
9 .flex-container > div {
10  color: #fff;
11  font-family: Verdana;
12  background-color: #4F81BD;
13  border: #385D8A solid 5px;
14  width: 100px;
15  margin: 10px;
16  text-align: center;
17  line-height: 75px;
18  font-size: 30px;
19 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Elementos 'flex'

Ejercicio:

<https://flexboxfroggy.com/#ca>

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.2 CSS Flexbox

### Propiedades del Contenedor Flex

#### 1. display

- flex, inline-flex

#### 2. flex-direction

- row, row-reverse, column, column-reverse

#### 3. flex-wrap

- nowrap, wrap, wrap-reverse

#### 4. justify-content

- flex-start, center, flex-end, space-between, space-around, space-evenly

#### 5. align-items

- flex-start, center, flex-end, stretch

#### 6. align-content

- flex-start, center, flex-end, space-between, space-around, stretch

#### 7. flex-grow

- Valor numérico (por defecto: 0)

#### 8. flex-shrink

- Valor numérico (por defecto: 1)

#### 9. flex-basis

- Tamaño (por ejemplo: **200px**)

#### 10. order

- Valor numérico (por defecto: 0)

#### 11. align-self

- auto, flex-start, center, flex-end, stretch

#### 12. flex-flow

- Combina **flex-direction** y **flex-wrap**

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3 CSS Grid

Proporciona un sistema de diseño basado en cuadrículas, con filas y columnas, que simplifica el diseño adaptativo sin tener que utilizar floats ni posicionamiento.

Es posterior a CSS Flexbox, y aporta algunas funcionalidades que simplifican las estructuras y el comportamiento de los elementos del diseño.

Los dos modelos se complementan. A veces puede utilizarse uno, el otro o una combinación de los dos.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3 CSS Grid

Para comenzar a trabajar con CSS Grid:

- Se ha de definir un elemento contenedor con la declaración CSS *display:grid*.
- Establecer las medidas de las filas y columnas con las propiedades *grid-template-columns* y *grid-template-rows*.
- Colocar los elementos hijo dentro de la parrilla con la propiedad *grid-area* y/o las propiedades *grid-column* y *grid-row*.

El orden de los elementos del código HTML **no importa**. Esto hace que sea muy fácil modificar el diseño según la resolución con media queries.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.1 Terminología básica.

Conceptos básicos:

Las líneas verticales de los elementos de la parrilla se llaman **columnas** y las líneas horizontales **filas**.

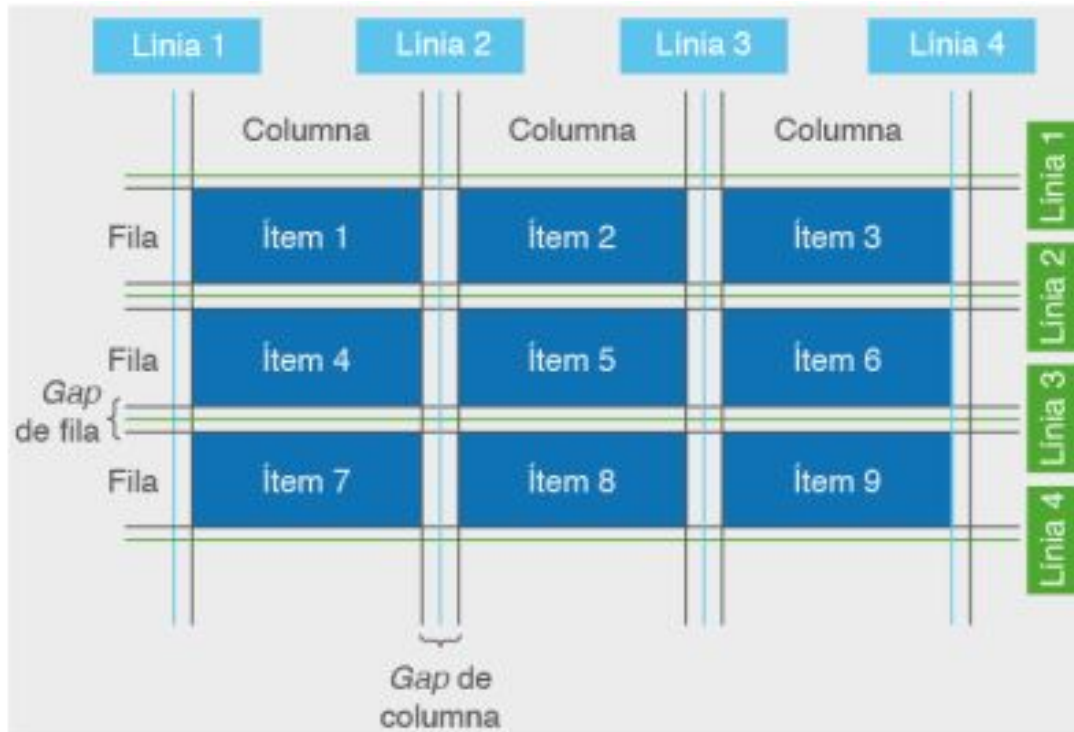
Los espacios entre cada columna o fila **gaps**. Se pueden especificar con las propiedades del contenedor *grid-row-gap*, *grid-column-gap* y *grid-gap* (versión abreviada de las anteriores).

Las líneas entre las columnas se les llama **líneas de columna**. Las líneas entre las filas **líneas de fila**.

Para ubicar los elementos dentro de la parrilla se utilizan estos conceptos.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.1 Terminología básica.





## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

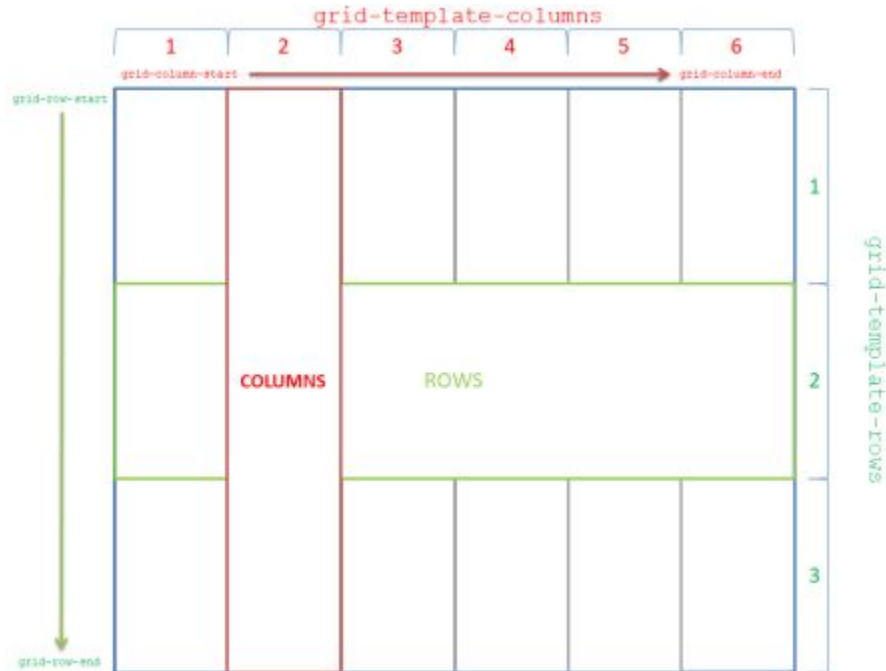
Es el elemento al que se le aplica la regla CSS *display:grid*. Es el padre directo de todos los elementos de la parrilla.

Este es un elemento de bloque y ocupa toda la amplitud disponible. No permite que el contenido se posicione a su alrededor.

Para crear un elemento en línea del tipo grid se puede utilizar *display:inline-grid*.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

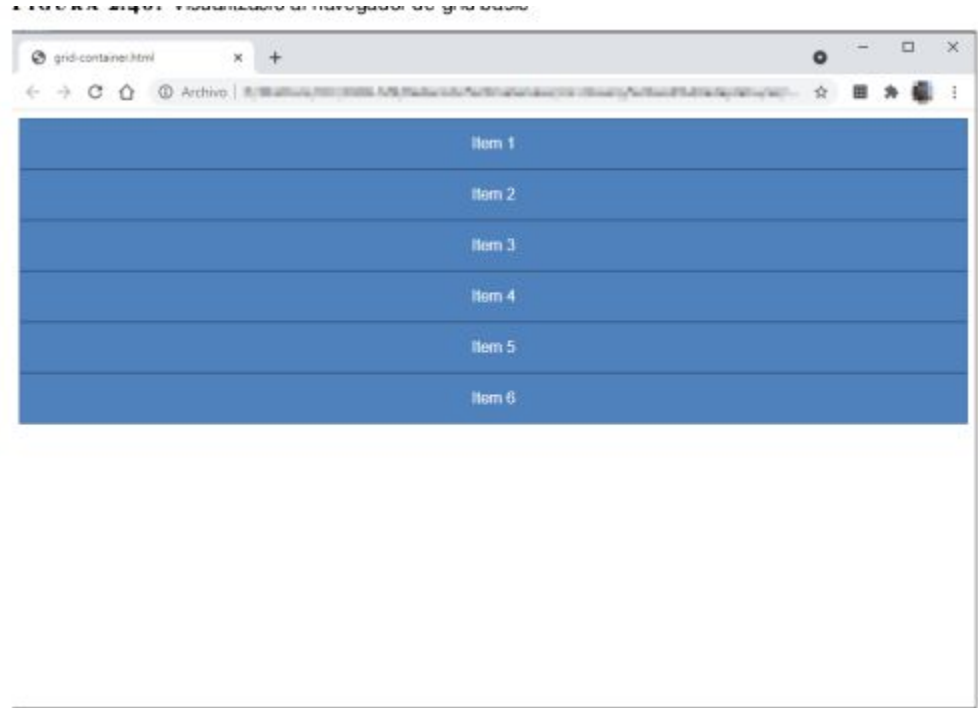


# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

```
1 .contenedor-grid {  
2   display: grid;  
3 }  
4  
5 .item {  
6   background-color: #4F81BD;  
7   border: #385D8A solid 1px;  
8   padding: 1rem;  
9   font-family: arial;  
10  color: white;  
11  text-align: center;  
12 }
```

```
1 <div class="contenedor-grid">  
2   <div class="item">Item 1</div>  
3   <div class="item">Item 2</div>  
4   <div class="item">Item 3</div>  
5   <div class="item">Item 4</div>  
6   <div class="item">Item 5</div>  
7   <div class="item">Item 6</div>  
8 </div>
```



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

#### Definición de la parrilla.

Para que cambie la disposición de la parrilla anterior se han de definir las propiedades *grid-template-columns* y *grid-template-rows*. Son los que definen el número y la amplitud de las columnas o filas.

Sus valores son una lista separada por espacios donde cada valor define la amplitud de la respectiva columna y la altura de la fila. Para que la amplitud de todas las filas sea la misma usaremos *auto*.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

### Definición de la parrilla.

```
1 .contenedor-grid {  
2     display: grid;  
3     grid-template-columns: auto auto auto auto auto auto;  
4 }
```

Este código generará una parrilla de una sola fila y seis columnas con la misma amplitud , y totalmente adaptiva.



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

#### Definición de la parrila.

Para hacer un diseño más personalizado, la propiedad *grid-template-columns* se utiliza la unidad de medida *fr*.

**1 fr** es igual a una fracción del espacio real disponible para cada columna, teniendo en cuenta los posibles espaciados entre columnas.

```
1 .contenedor-grid {  
2     display: grid;  
3     grid-template-columns: 1fr 1fr 1fr 1fr 1fr 1fr;  
4 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

### Definición de la parrilla.

Si por el contrario, se quieren definir diferentes medidas para cada una de las columnas también se puede hacer lo siguiente.

```
1 .contenedor-grid {  
2     display: grid;  
3     grid-template-columns: 2fr 2fr 1fr 1fr 1fr 1fr;  
4 }
```

La propiedad grid-template-columns también acepta anchos fijos.

```
1 .contenedor-grid {  
2     display: grid;  
3     grid-template-columns: 20% 2fr 2fr 75px 1fr 1fr;  
4 }
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

### Definición de la parrila.

Para parrillas grande se puede utilizar la notación repeat()

```
1 .contenedor-grid {  
2     display: grid;  
3     grid-template-columns: repeat(6, 1fr);  
4 }
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

### Ejemplos de definición de anchura y altura de columnas y filas.

Parrila con una primera columna de 30px, 4de 1fr y una columna final de 10px.

```
1 .contenedor-grid {  
2   display: grid;  
3   grid-template-columns: 30px repeat(4, 1fr) 10px;  
4 }
```

También se puede usar la notación de repetición para crear un patrón de iteración de columnas o filas.

```
1 .contenedor-grid {  
2   display: grid;  
3   grid-template-columns: repeat(4, 1fr 2fr);  
4 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.2 Contenedor de parrilla.

Ejemplos de definición de anchura y altura de columnas y filas.

La propiedad *grid-template-rows* funciona de forma equivalente con el número de filas y su altura.

Para definir una parrilla de 3x3 con nueve celdas.

```
1 .contenedor-grid {  
2   display: grid;  
3   grid-template-columns: repeat(3, 1fr);  
4   grid-template-rows: repeat(3, 1fr);  
5 }  
6  
7 .item {  
8   background-color: #4FB8BD;  
9   border: #385D8A solid 1px;  
10  padding: 1rem;  
11  font-family: arial;  
12  color: white;  
13  text-align: center;  
14 }
```

```
1 <div class="contenedor-grid">  
2   <div class="item">Item 1</div>  
3   <div class="item">Item 2</div>  
4   <div class="item">Item 3</div>  
5   <div class="item">Item 4</div>  
6   <div class="item">Item 5</div>  
7   <div class="item">Item 6</div>  
8   <div class="item">Item 7</div>  
9   <div class="item">Item 8</div>  
10  <div class="item">Item 9</div>  
11 </div>
```



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

#### Propiedad '*grid-template-areas*'.

Permite definir una plantilla haciendo referencia a los nombres de las áreas, que asignamos a la propiedad *grid-areas*. Si se repite el nombre de un área, el contenido se repite en todas las celdas.

Un punto significa una celda vacía.

Veamos un ejemplo de uso combinando *grid-template-areas*, *grid-template-columns* y *grid-template-rows* en una disposición típica de cabecera + navegación + zona principal + barra lateral + pie de página para demostrar la simplicidad y potencia del modelo **CSS Grid**.

```

1 body,
2     html {
3         margin: 0;
4         padding: 0;
5     }
6
7     * {
8         box-sizing: border-box;
9     }
10
11     .contenedor-grid {
12         display: grid;
13         grid-template-columns: repeat(4, 1fr);
14         grid-template-rows: 1fr 3fr 1fr;
15         grid-template-areas:
16             "capçalera capçalera capçalera capçalera"
17             "navegació principal principal lateral"
18             "peu peu peu peu";
19         height: 100vh; /*Volem que la graella ocupi tot el
20             camp de visió. */
21     }
22
23     header {
24         grid-area: capçalera;
25         background-color: orange;
26         border: black solid 1px;
27         font-family: arial;
28         color: white;
29     }
30
31     nav {
32         grid-area: navegacio;
33         background-color: lightseagreen;
34         border: black solid 1px;
35         font-family: arial;
36         color: white;
37     }
38
39     section {
40         grid-area: principal;

```

```

41         background-color: #4F81BD;
42         border: black solid 1px;
43         font-family: arial;
44         color: white;
45     }
46
47     aside {
48         grid-area: lateral;
49         background-color: lightcoral;
50         border: black solid 1px;
51         font-family: arial;
52         color: white;
53     }
54
55     footer {
56         grid-area: peu;
57         background-color: darkgreen;
58         border: black solid 1px;
59         font-family: arial;
60         color: white;
61     }

```

orks web

```

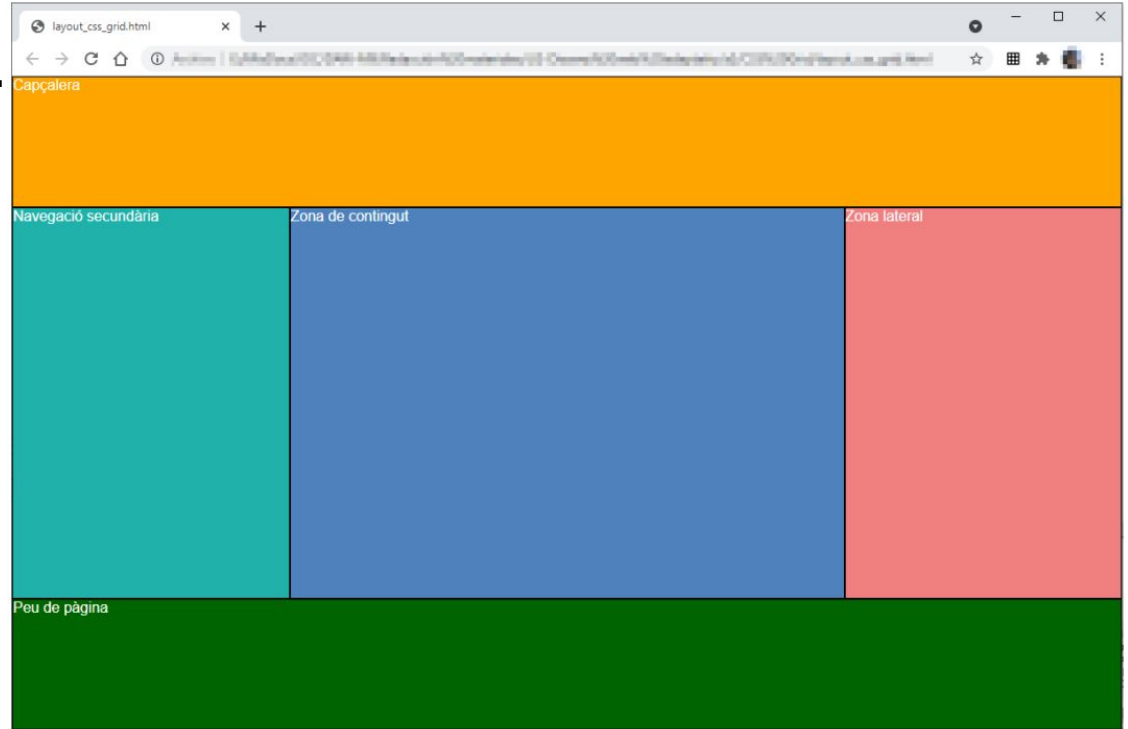
1 <div class="contenedor-grid">
2     <header>Capçalera</header>
3     <nav>Navegació secundària</nav>
4     <section>
5         <article>
6             Zona de contingut
7         </article>
8     </section>
9     <aside>
10         Zona lateral
11     </aside>
12     <footer>
13         Peu de pàgina
14     </footer>
15 </div>

```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.2 Contenedor de parrilla.

Propiedad '*grid-template-areas*'.



<https://codepen.io/iocdawm9/pen/PojYomq>

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

El contenedor de cuadrícula contienen los elementos de la cuadrícula. Su comportamiento por defecto es un elemento por cada intersección fila-columna. Este comportamiento se puede modificar por CSS.

#### Propiedad 'grid-column'

Permite indicar la columna donde se colocará un elemento. Es una propiedad abreviada de las propiedades *grid-column-start* y *grid-column-end*. Esta propiedad abreviada es muy fácil y práctica.

*grid-column*: <línea\_columna\_inici> / <línea\_columna\_fi> |

<línea\_columna\_inici> / span <valor>;

Se puede dar el número de columna inicial y el final o la línea inicial y a cuantas columnas se expande.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

#### Propiedad 'grid-column'

Estas dos expresiones son equivalentes:

```
1 .item1 {  
2   grid-column: 1 / 4; /* Fes que l'element1 comenci a la columna  
   1 i acabi abans de la columna 4. */  
3 }
```

```
1 .item1 {  
2   grid-column: 1 / span 3; /* Fes que l'element1 comenci a la  
   columna 1 i ocupi 3 columnes.*/  
3 }
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.3 Elementos de la parrilla.

### Propiedad 'grid-column'

Dado el siguiente código:

```
1  .contenedor-grid {
2      display: grid;
3      grid-template-columns: repeat(5, 1fr);
4      grid-template-rows: repeat(3, 1fr);
5  }
6
7  .item {
8      background-color: #4F81BD;
9      border: #385D8A solid 1px;
10     padding: 1rem;
11     font-family: arial;
12     color: white;
13     text-align: center;
14 }
15
16 .item1 {
17     grid-column: 1 / 4;
18 }
```

```
1  <div class="contenedor-grid">
2      <div class="item item1">Ítem 1</div>
3      <div class="item">Ítem 2</div>
4      <div class="item">Ítem 3</div>
5      <div class="item">Ítem 4</div>
6      <div class="item">Ítem 5</div>
7      <div class="item">Ítem 6</div>
8      <div class="item">Ítem 7</div>
9      <div class="item">Ítem 8</div>
10     <div class="item">Ítem 9</div>
11 </div>
```

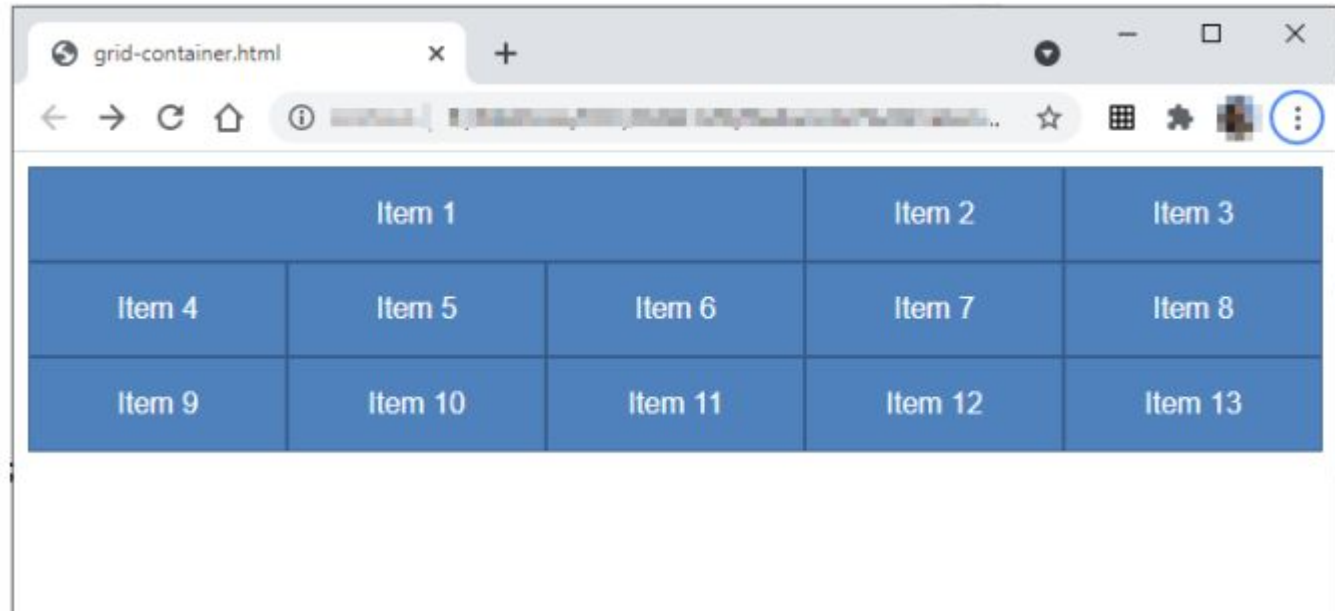


## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

Propiedad 'grid-column'

Resultado:



A screenshot of a web browser window displaying a grid layout. The browser's address bar shows 'grid-container.html'. The grid consists of 13 blue rectangular items arranged in 3 rows and 5 columns. The items are labeled 'Item 1' through 'Item 13'. 'Item 1' spans the first two columns of the first row. 'Item 2' and 'Item 3' span the third and fourth columns of the first row. The second row contains 'Item 4', 'Item 5', 'Item 6', 'Item 7', and 'Item 8'. The third row contains 'Item 9', 'Item 10', 'Item 11', 'Item 12', and 'Item 13'.

|        |         |         |         |         |
|--------|---------|---------|---------|---------|
| Item 1 |         | Item 2  |         | Item 3  |
| Item 4 | Item 5  | Item 6  | Item 7  | Item 8  |
| Item 9 | Item 10 | Item 11 | Item 12 | Item 13 |

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

#### Propiedad '*grid-row*'

Analoga a *grid-column*, pero hace referencia a las filas que ocupa un elemento de cuadrícula. También es la propiedad abreviada de *grid-row-start* y *grid-row-end*.

Tiene una sintaxis equivalente a *grid-column*:

`grid-row: <línia_fila_inici> / <línia_fila_fi> | <línia_fila_inici>`

`/ span <valor>;`

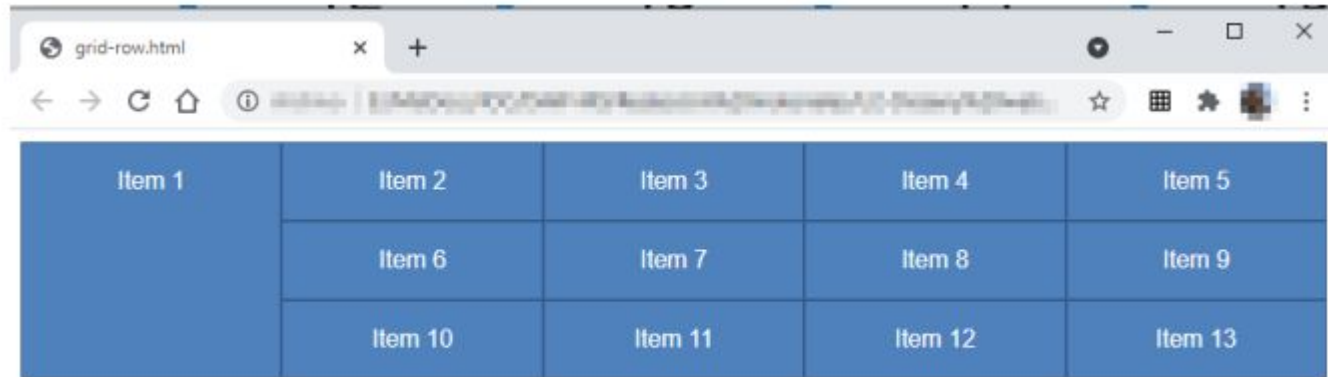
```
1 .item1 {  
2   grid-row: 1 / 4; /* Fes que l'element1 comenci a la fila 1 i  
   acabi abans de la fila 4. */  
3 }
```

```
1 .item1 {  
2   grid-row: 1 / span 3; /* Fes que l'element1 comenci a la fila  
   1 i ocupi 3 files.*/  
3 }
```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

Propiedad '*grid-row*'



A screenshot of a web browser window displaying a grid layout. The browser's address bar shows 'grid-row.html'. The grid consists of 13 items arranged in 3 rows and 5 columns. The first column has a rowspan of 3, meaning 'Item 1' spans all three rows. The other columns have 1 item each per row. The items are labeled 'Item 1' through 'Item 13'.

|        |         |         |         |         |
|--------|---------|---------|---------|---------|
| Item 1 | Item 2  | Item 3  | Item 4  | Item 5  |
| Item 1 | Item 6  | Item 7  | Item 8  | Item 9  |
| Item 1 | Item 10 | Item 11 | Item 12 | Item 13 |

<https://codepen.io/iocdawm9/pen/JjJPJEB>

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

#### Propiedad '*grid-area*'

Se puede utilizar junto a *grid-template-areas* para asignar los elementos de la cuadrícula a áreas del contenedor etiquetadas con nombres. También se puede utilizar como versión abreviada de *grid-row-start*, *grid-column-start*, *grid-row-end* y *grid-column-end*.

Permite definir el área de un elemento de la cuadrícula indicando sus coordenadas fila/columna de su parte superior izquierda y la inferior derecha.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.3.3 Elementos de la parrilla.

### Propiedad '*grid-area*'

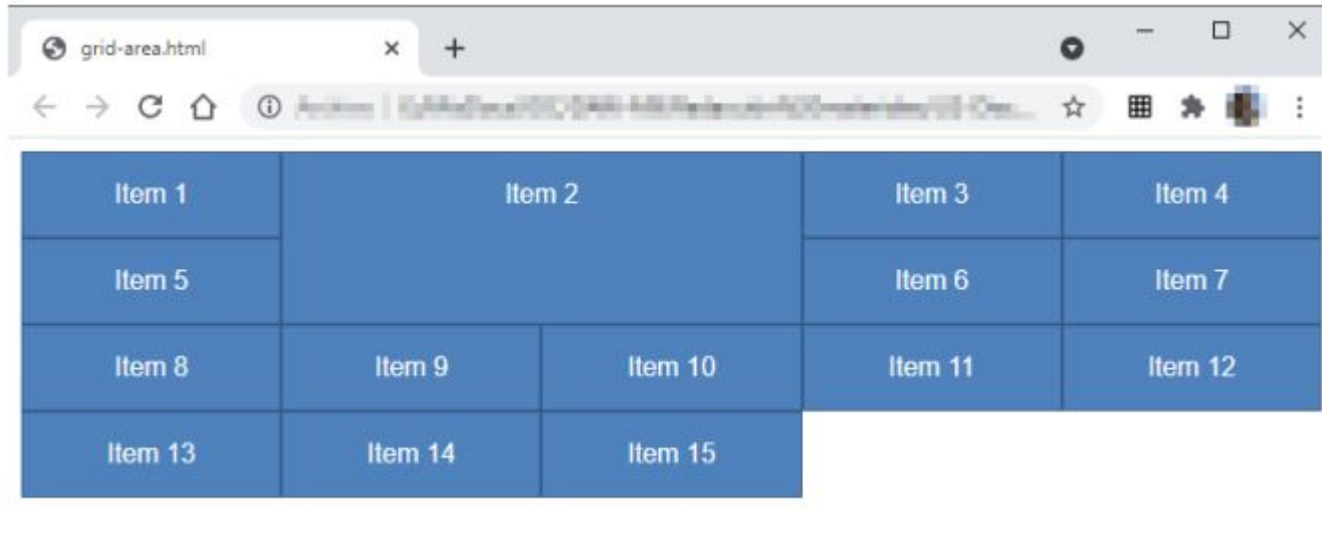
```
1 .contenedor-grid {
2     display: grid;
3     grid-template-columns: repeat(5, 1fr);
4     grid-template-rows: repeat(3, 1fr);
5 }
6
7 .item {
8     background-color: #4F81BD;
9     border: #385D8A solid 1px;
10    padding: 1rem;
11    font-family: arial;
12    color: white;
13    text-align: center;
14 }
15 .item2{
16     grid-area: 1 / 2 / 3 / 4; /*Fem que l'item2 comenci a la
17     línia de fila 1 línia de columna 2 i acabi a la línia de fila 3 i línia de columna 4.*/
18 }
```

```
1 <div class="contenedor-grid">
2     <div class="item">Ítem 1</div>
3     <div class="item item2">Ítem 2</div>
4     <div class="item">Ítem 3</div>
5
6     <div class="item">Ítem 4</div>
7     <div class="item">Ítem 5</div>
8     <div class="item">Ítem 6</div>
9     <div class="item">Ítem 7</div>
10    <div class="item">Ítem 8</div>
11    <div class="item">Ítem 9</div>
12    <div class="item">Ítem 10</div>
13    <div class="item">Ítem 11</div>
14    <div class="item">Ítem 12</div>
15    <div class="item">Ítem 13</div>
16    <div class="item">Ítem 14</div>
17    <div class="item">Ítem 15</div>
18 </div>
```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.3.3 Elementos de la parrilla.

Propiedad '*grid-area*'



<https://codepen.io/iocdawm9/pen/GREKEyR> <https://cssgridgarden.com/#ca>

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4 Patrones de diseño web adaptativo.

La mayoría de los diseños se pueden clasificar en:

- **Mayormente fluido.** (Mostly Fluid)
- **Caída de columnas.** (Column Drop)
- **Variador de estructura..** (Layout Shifter)
- **Pequeños retoques.** (Tiny Tweaks)
- **Fuera de la ventana.** (Off canvas)

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.1 Mayoritariamente fluido.

Es uno de los patrones más habituales , por su sencillez y efectividad.

Es una estructura multicolumna que en las pantallas más grandes deja márgenes laterales de medidas superiores, para evitar anchos exagerados.

A medida que las pantallas se hacen más estrechas, se basa en cuadrículas fluidas y imágenes adaptativas para ir esclanado y apliando las columnas verticales.





## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.1 Mayoritariamente fluido.

Su implementación es **sencilla**, casi no se utilizan ***media queris*** ya que solamente hay una variación sencilla en la distribución de columnas y la fijación de la anchura para pantallas más grandes.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.1 Mayoritariamente fluido.

### Ejemplo de implementación de diseño mayoritariame

```
1
2  .container {
3    display: -webkit-flex;
4    display: flex;
5    -webkit-flex-flow: row wrap;
6    flex-flow: row wrap;
7  }
8  /* A telèfons mòbils, apilem columnes. */
9  .c1, .c2, .c3 {
10   width: 100%;
11 }
12 /* Per a dispositius de més de 600px d'amplada, la 2a i 3a columna es
   reparteixen l'amplada de pantalla. */
13 @media (min-width: 600px) {
14
15   .c1 {
16     width: 100%;
```

```
17   }
18   .c2 {
19     width: 50%;
20   }
21   .c3 {
22     width: 50%;
23   }
24 }
25
26 /* A pantalles més grans, fixem l'amplada del contenidor principal
   ja no continuarà creixent, l'amplada de les columnes no la
   toquem. */
27 @media (min-width: 800px) {
28   .container {
29     width: 800px;
30     margin-left: auto;
31     margin-right: auto;
32   }
33 }
```

```
1  <body>
2    <div class="container" role="main">
3      <div class="c1">
4      </div>
5      <div class="c2">
6      </div>
7      <div class="c3">
8      </div>
9    </div>
10 </body>
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.1 Mayoritariamente fluido.

### Ejemplo de implementación de diseño mayoritariame

```
1
2     .container {
3         display: -webkit-flex;
4         display: flex;
5         -webkit-flex-flow: row wrap;
6         flex-flow: row wrap;
7     }
8     /* A telèfons mòbils, apilem columnes. */
9     .c1, .c2, .c3 {
10         width: 100%;
11     }
12     /* Per a dispositius de més de 600px d'amplada, la 2a i 3a columna es
13         reparteixen l'amplada de pantalla. */
14     @media (min-width: 600px) {
15         .c1 {
16             width: 100%;
```

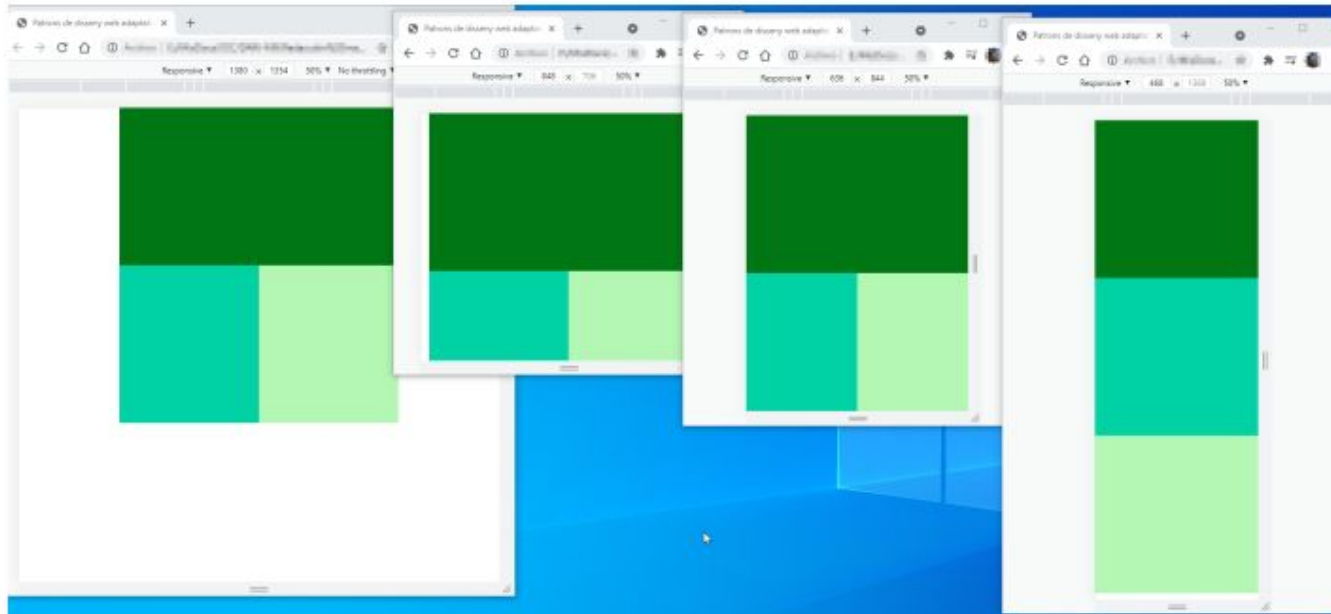
```
17         }
18         .c2 {
19             width: 50%;
20         }
21         .c3 {
22             width: 50%;
23         }
24     }
25
26     /* A pantalles més grans, fixem l'amplada del contenidor principal
27         ja no continuarà creixent, l'amplada de les columnes no la
28         toquem. */
29     @media (min-width: 800px) {
30         .container {
31             width: 800px;
32             margin-left: auto;
33             margin-right: auto;
34         }
35     }
```

```
1     <body>
2         <div class="container" role="main">
3             <div class="c1">
4             </div>
5             <div class="c2">
6             </div>
7             <div class="c3">
8             </div>
9         </div>
10    </body>
```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.1 Mayoritariamente fluido.

**Ejemplo de implementación de diseño mayoritariamente fluido.**



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.2 Caída de columnas.

Es un patrón bastante popular. Comienza con una estructura multicolumna y acaba con una columna única que apila todas las columnas que formaban parte de la estructura inicial, que van cayendo a medida que la amplitud del dispositivo se va haciendo más estrecha.

A diferencia del patrón mayoritariamente fluido, la medida total de los elementos de este diseño se mantiene constante. La adaptación de las diversas medidas de pantalla se basa en apilar columnas.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.2 Caída de columnas.

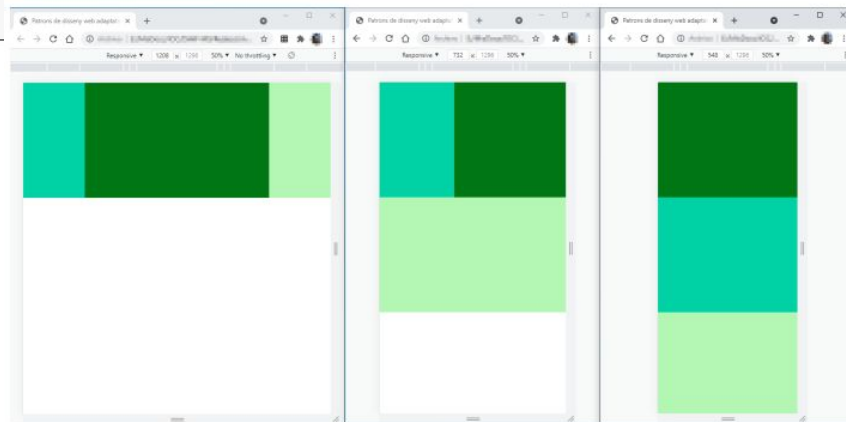
Este patrón contempla altera el orden de las columnas según el dispositivo.

Exemple d'implementació del patró de disseny de caiguda de columnes

```
1  .container {
2    display: -webkit-flex;
3    display: flex;
4    -webkit-flex-flow: row wrap;
5    flex-flow: row wrap;
6  }
7
8  /* A telèfons mòbils, apilem columnes. */
9  .c1,
10 .c2,
11 .c3 {
12   width: 100%;
13 }
14
15 /* Per a dispositius de més de 600px d'amplada, la primera columna
16 passa a ser la segona i hi ha una primera fila de dues
17 columnes. */
18 @media (min-width: 600px) {
19   .c1 {
20     width: 60%;
21     -webkit-order: 2;
22     order: 2;
23   }
24   .c2 {
25     width: 40%;
26     -webkit-order: 1;
27     order: 1;
28   }
29   .c3 {
30     width: 100%;
31     -webkit-order: 3;
32     order: 3;
33   }
34 }
35
36 /* A equips d'escriptori, tenim estructura multicolumna de tres
37 columnes que ocupen l'amplada de pantalla. */
38 @media (min-width: 800px) {
39   .c2 {
40     width: 20%;
41   }
42   .c3 {
43     width: 20%;
44   }
45 }
```

Amb el mateix codi HTML que en l'exemple anterior...

```
1  <body>
2    <div class="container" role="main">
3      <div class="c1">
4      </div>
5      <div class="c2">
6      </div>
7      <div class="c3">
8      </div>
9    </div>
10 </body>
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.2 Caída de columnas.

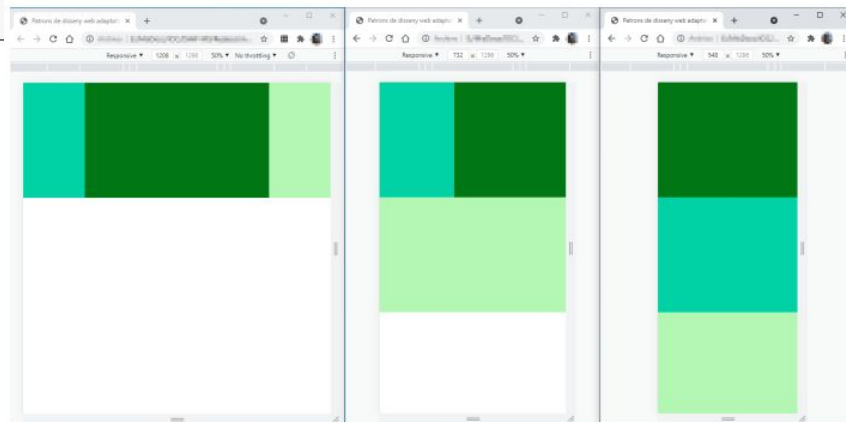
Este patrón contempla altera el orden de las columnas según el dispositivo.

Exemple d'implementació del patró de disseny de caiguda de columnes

```
1  .container {
2    display: -webkit-flex;
3    display: flex;
4    -webkit-flex-flow: row wrap;
5    flex-flow: row wrap;
6  }
7
8  /* A telèfons mòbils, apilem columnes. */
9  .c1,
10 .c2,
11 .c3 {
12   width: 100%;
13 }
14
15 /* Per a dispositius de més de 600px d'amplada, la primera columna
16   passa a ser la segona i hi ha una primera fila de dues
17   columnes. */
18 @media (min-width: 600px) {
19   .c1 {
20     width: 60%;
21     -webkit-order: 2;
22     order: 2;
23   }
24   .c2 {
25     width: 40%;
26     -webkit-order: 1;
27     order: 1;
28   }
29   .c3 {
30     width: 100%;
31     -webkit-order: 3;
32     order: 3;
33   }
34 }
35
36 /* A equips d'escriptori, tenim estructura multicolumna de tres
37   columnes que ocupen l'amplada de pantalla. */
38 @media (min-width: 800px) {
39   .c2 {
40     width: 20%;
41   }
42   .c3 {
43     width: 20%;
44   }
45 }
```

Amb el mateix codi HTML que en l'exemple anterior...

```
1  <body>
2    <div class="container" role="main">
3      <div class="c1">
4      </div>
5      <div class="c2">
6      </div>
7      <div class="c3">
8      </div>
9    </div>
10 </body>
```

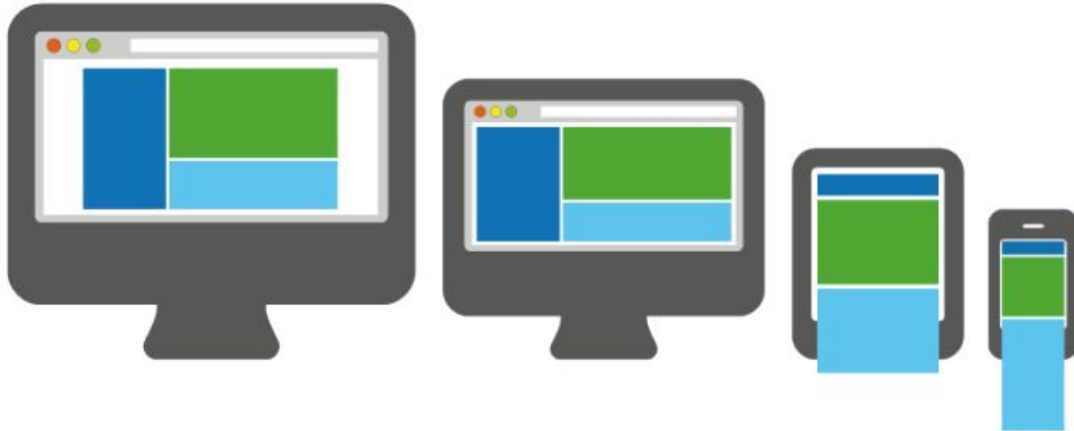


## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.3 Variador de estructura.

El variador de estructura maximiza la adaptación a las diferentes medidas de pantalla. Utiliza diferentes estructuras para los tres tipos generales de pantalla.

Es un poco más laborioso, ya que tiene más puntos de interrupción con media queries más complejas. Esto hace que no sea tan popular como los anteriores.



En este patrón es más importante la reubicación del contenido en vez de su apilamiento.



# 5. Patrones de diseño web adaptativo. Frameworks web

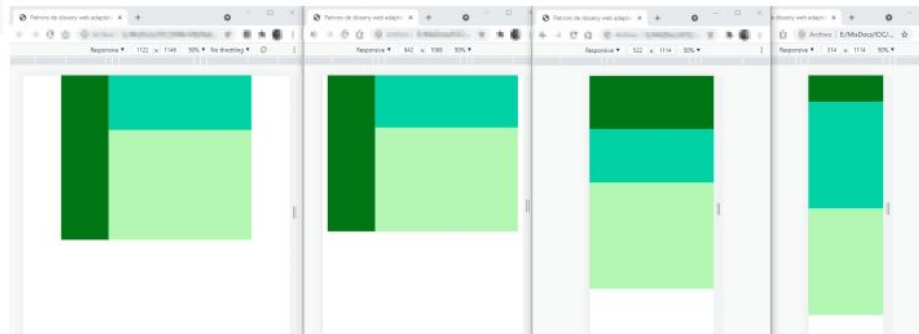
## 2.4.3 Variador de estructura.

Exemple d'implementació del patró variador d'estructura

```
1 .container {
2   display: -webkit-flex;
3   display: flex;
4   -webkit-flex-flow: row wrap;
5   flex-flow: row wrap;
6 }
7
8 .c1,
9 .c2,
10 .c3,
11 .c4 {
12   width: 100%;
13 }
14
15 .c1 {
16   min-height: 10vh;
17 }
18
19 .c2 {
20   min-height: 40vh;
21 }
22
23 .c3 {
24   min-height: 40vh;
25 }
26
27 @media (min-width: 400px) {
28
29   /*En dispositius petits, però no gaire, la primera columna és
30   una mica més gran.*/
31   .c1 {
32     min-height: 20vh;
33   }
34   .c2 {
35     min-height: 20vh;
36   }
37   .c3 {
38     min-height: 40vh;
39   }
40 }
41
42
```

```
43 @media (min-width: 600px) {
44
45   /*En tauletes es canvia la disposició redistribuint amplituds
46   entre c1 i c4.*/
47   .c1 {
48     width: 25%;
49   }
50   .c4 {
51     width: 75%;
52   }
53 }
54
55 @media (min-width: 800px) {
56
57   /* A partir de 800px, el contenidor principal té una amplitud
58   fixa de 800px.*/
59   .container {
60     width: 800px;
61     margin-left: auto;
62     margin-right: auto;
63   }
64 }
65
```

```
1 <body>
2   <div class="container" role="main">
3     <div class="c1"></div>
4     <div class="c4">
5       <div class="c2"></div>
6       <div class="c3"></div>
7     </div>
8   </div>
9 </body>
```



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.4 Pequeños retoques.

Realiza pequeños cambios en el diseño, como:

- ajustes de la medida de la fuente.
- espaciados.
- paddings.
- medidas de las imágenes.
- desplazamiento de contenido poco significativo.

Es adecuado para diseños de una sola columna, como los sitios web lineales con una única página o artículo con mucho texto.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.4 Pequeños retoques.

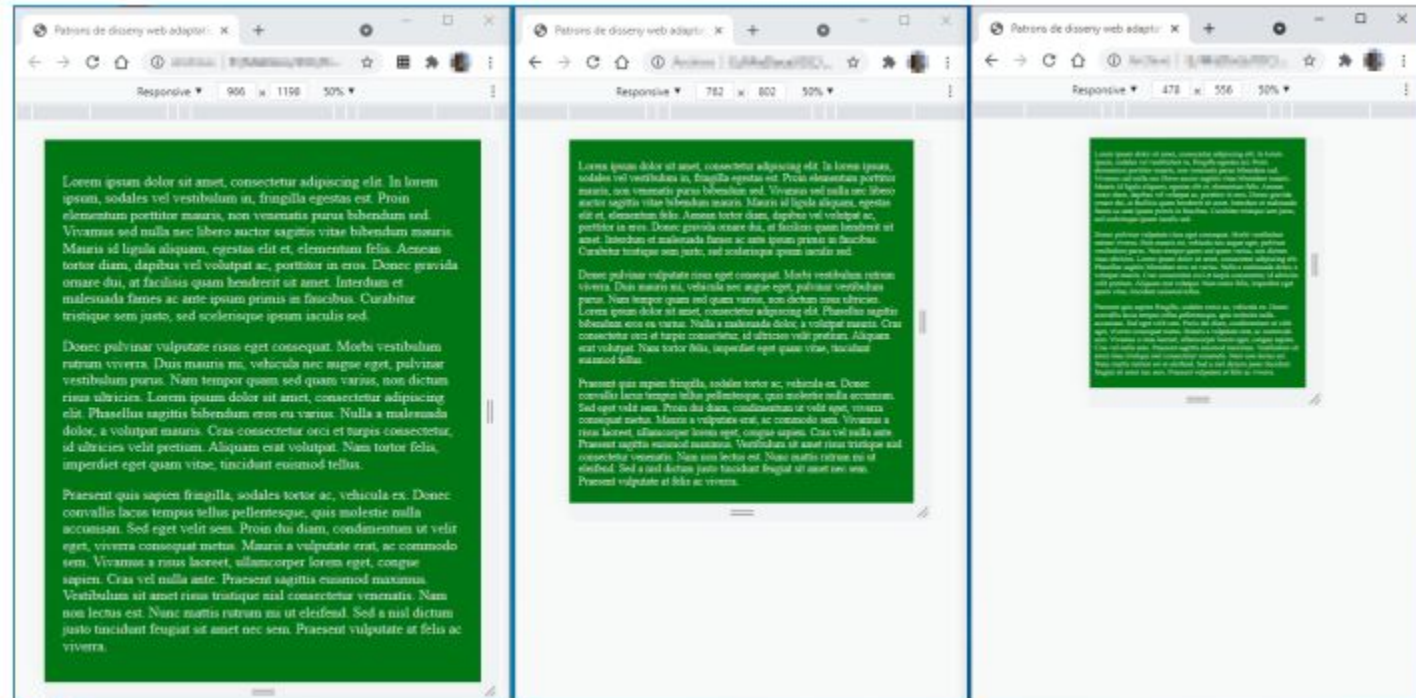
### Exemple d'implementació del patró de petits retocs

```
1      /*S'ajusta el padding i la font-size segons resolucions.*/
2      .c1 {
3          padding: 10px;
4          width: 100%;
5      }
6
7      @media (min-width: 500px) {
8          .c1 {
9              padding: 20px;
10             font-size: 1.5em;
11         }
12     }
13
14     @media (min-width: 800px) {
15         .c1 {
16             padding: 40px;
17             font-size: 2em;
18         }
19     }
```

```
1      <body>
2      <div class="container" role="main">
3
4          <div class="c1">
5              <p>
6                  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
7                  In lorem ipsum, sodales vel vestibulum in,
8                  fringilla egestas est. Proin elementum porttitor mauris,
9                  non venenatis purus bibendum sed. Vivamus sed
10                 nulla nec libero auctor sagittis vitae bibendum mauris.
11                 Mauris id ligula aliquam, egestas elit et,
12                 elementum felis. Aenean tortor diam, dapibus vel volutpat
13                 ac, porttitor in eros. Donec gravida ornare
14                 dui, at facilisis quam hendrerit sit amet. Interdum et
15                 malesuada fames ac ante ipsum primis in faucibus.
16                 Curabitur tristique sem justo, sed scelerisque ipsum
17                 iaculis sed.
18             </p>
19
20             <p>
21                 Donec pulvinar vulputate risus eget consequat. Morbi
22                 vestibulum rutrum viverra. Duis mauris mi, vehicula
23                 nec augue eget, pulvinar vestibulum purus. Nam tempor quam
24                 sed quam varius, non dictum risus ultricies.
25                 Lorem ipsum dolor sit amet, consectetur adipiscing elit.
26                 Phasellus sagittis bibendum eros eu varius.
27                 Nulla a malesuada dolor, a volutpat mauris. Cras
28                 consectetur orci et turpis consectetur, id ultricies
29                 velit pretium. Aliquam erat volutpat. Nam tortor felis,
30                 imperdiet eget quam vitae, tincidunt euismod
31                 tellus.
32             </p>
33
34             <p>
35                 Praesent quis sapien fringilla, sodales tortor ac,
36                 vehicula ex. Donec convallis lacus tempus tellus
37                 pellentesque, quis molestie nulla accumsan. Sed eget velit
38                 sem. Proin dui diam, condimentum ut velit
39                 eget, viverra consequat metus. Mauris a vulputate erat, ac
40                 commodo sem. Vivamus a risus laoreet,
41                 ullamcorper lorem eget, congue sapien. Cras vel nulla ante
42                 . Praesent sagittis euismod maximus.
43                 Vestibulum sit amet risus tristique nisl consectetur
44                 venenatis. Nam non lectus est. Nunc mattis rutrum
45                 mi ut eleifend. Sed a nisl dictum justo tincidunt feugiat
46                 sit amet nec sem. Praesent vulputate at felis
47                 ac viverra.
48             </p>
49         </div>
50     </div>
51 </body>
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.4 Pequeños retoques.



## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.5 Fuera de la ventana.

El patrón ***fuera de la ventana*** tiene una implementación más compleja. Implica utilizar tecnologías adicionales como Javascript. En vez de apilar contenido verticalmente, escondemos el contenido menos relevante fuera de la pantalla y solo se enseña cuando la medida de la pantalla es suficientemente grande.

En pantallas pequeñas, el acceso al contenido se hace clicando un elemento determinado, como por ejemplo el menú de hamburguesa.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 2.4.5 Fuera de la ventana.

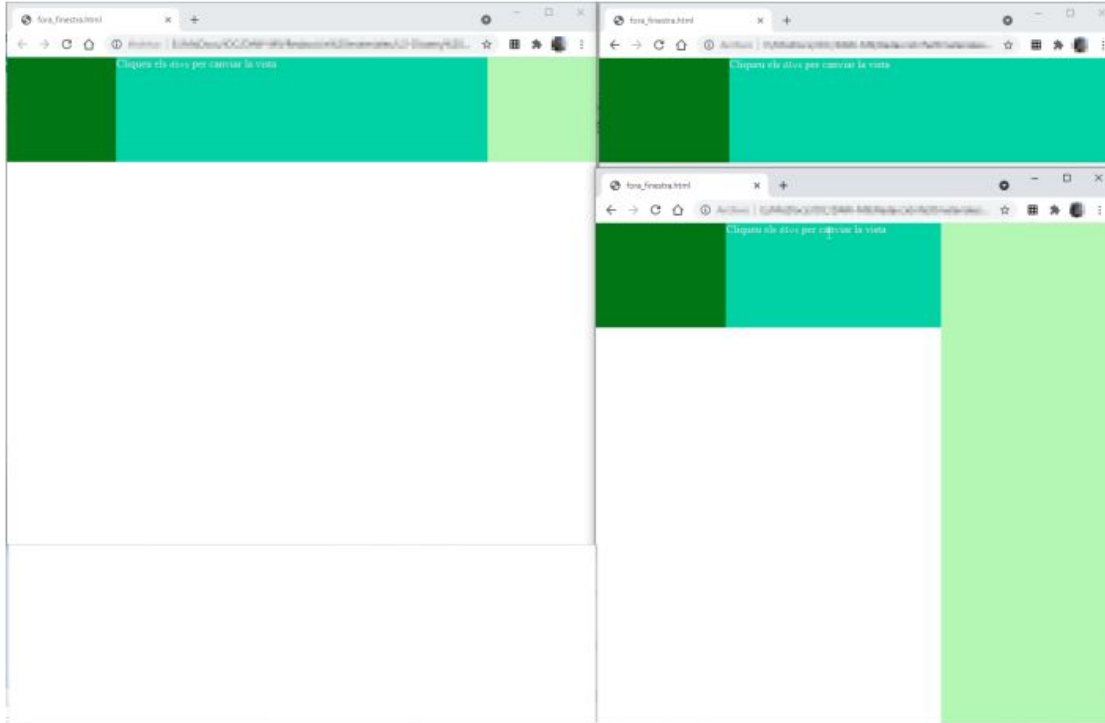
```
1 body {
2   overflow-x: hidden;
3 }
4
5 .container {
6   display: block;
7 }
8
9 .c1,
10 .c3 {
11   position: absolute;
12   width: 250px;
13   height: 100%;
14
15   /*
16    Millora de rendiment a les versions més recents de Chrome
17    */
18   -webkit-backface-visibility: hidden;
19   backface-visibility: hidden;
20
21   -webkit-transition: -webkit-transform 0.4s ease-out;
22   transition: transform 0.4s ease-out;
23
24   z-index: 1;
25 }
26
27 .c1 {
28   -webkit-transform: translate(-250px, 0);
29   transform: translate(-250px, 0);
30 }
31
32 .c2 {
33   width: 100%;
34   position: absolute;
35 }
36
37 .c3 {
38   left: 100%;
39 }
```

```
40
41 /*classes open per amagar/mostrar c1 i c3*/
42
43 .c1.open {
44   -webkit-transform: translate(0, 0);
45   transform: translate(0, 0);
46 }
47
48 .c3.open {
49   -webkit-transform: translate(-250px, 0);
50   transform: translate(-250px, 0);
51 }
52
53 @media (min-width: 500px) {
54
55   /* Farem servir flexbox amb pantalles de més de 500px d'
56    amplada. */
57
58   .container {
59     display: -webkit-flex;
60     display: flex;
61     -webkit-flex-flow: row nowrap;
62     flex-flow: row nowrap;
63   }
64
65   .c1 {
66     position: relative;
67     -webkit-transition: none 0s ease-out;
68     transition: none 0s ease-out;
69     -webkit-transform: translate(0, 0);
70     transform: translate(0, 0);
71   }
72
73   .c2 {
74     position: static;
75   }
76
77   @media (min-width: 800px) {
78     body {
79       overflow-x: auto;
80     }
81
82     .c3 {
83       position: relative;
84       left: auto;
85       -webkit-transition: none 0s ease-out;
86       transition: none 0s ease-out;
87       -webkit-transform: translate(0, 0);
88       transform: translate(0, 0);
89     }
90   }
```

```
1 <body>
2   <div class="container" role="main">
3     <div class="c1" id="c1">
4     </div>
5     <div class="c2" id="c2">
6       Cliqueu els divs per canviar la vista
7     </div>
8     <div class="c3" id="c3">
9     </div>
10  </div>
11
12  <script type="text/javascript">
13    var position = 0;
14    var mainPanel = document.getElementById("c2");
15    var leftDrawer = document.getElementById("c1");
16    var rightDrawer = document.getElementById("c3");
17
18    function toggle(evt) {
19      position++;
20      if (position % 3 == 0) {
21        leftDrawer.classList.remove("open");
22        rightDrawer.classList.remove("open");
23      } else if (position % 3 == 1) {
24        leftDrawer.classList.add("open");
25        rightDrawer.classList.remove("open");
26      } else {
27        leftDrawer.classList.remove("open");
28        rightDrawer.classList.add("open");
29      }
30    }
31
32    mainPanel.addEventListener("click", toggle);
33    leftDrawer.addEventListener("click", toggle);
34    rightDrawer.addEventListener("click", toggle);
35
36  </script>
37 </body>
```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 2.4.5 Fuera de la ventana.



En esta página hay un ejemplo:  
<https://codepen.io/iocdawm9/pen/VwbNpJg>

## 5. Patrones de diseño web adaptativo. Frameworks web

### **3. Entornos de trabajo CSS.**

El diseño adaptativo comporta bastante faena adicional en los proyectos web. Obliga al diseñador a tener en cuenta todas las variaciones del sitio web según las diferentes resoluciones de los dispositivos con los que se visualiza.

Esta tarea se puede aliviar si se utiliza un entorno de trabajo o framework adaptativo.



# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

Características:

- Sistema de cuadrícula, con todas las estructuras para organizar el contenido y diseñar la disposición de los elementos de la web.
- Elementos tipográficos.
- Compatibilidad con los diferentes navegadores.
- Clases para ayudar con el posicionamiento de elementos.
- Clases de utilidades (utility classes), que permiten de forma rápida asignar formatos complejos a los elementos con tan solo una clase CSS.
- Elementos de navegación (menús desplegables, por ejemplo).
- Elementos multimedia, comentarios, tooltips y botones.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

Características:

- Elementos de interfaz de uso general:
  - Menús desplegable (*dropdowns*).
  - Botones y grupos de botones.
  - Formularios.
  - Breadcrumbs, o migas de pan.
  - Paginación.
  - Etiquetas.
  - Badges, insignias o placas.
  - Encabezados.
  - Miniaturas o thumbnails.
  - Alertas.
  - Barras de progreso.
  - Listas agrupadas.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.1. Introducción a los 'frameworks' CSS.

Un entorno de trabajo o framework es un conjunto predefinido de conceptos, módulos y criterios estandarizados que facilitan el desarrollo de sitios y aplicaciones web.

Proporcionan funcionalidades genéricas con módulos ya escritos y componentes específicos creados de forma estandarizada.

Es un entorno de programación reutilizable que permite a los diseñadores web y desarrolladores construir fácilmente los proyectos, con poca codificación y sin tener que preocuparse de los detalles a bajo nivel.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

Trabajar con frameworks facilita mucho la faena, y proporciona implícitamente todas estas funcionalidades a las webs.

- Código limpio y consistente.
- Compatibilidad con los diferentes navegadores.
- Diseño adaptativo basado en cuadrícula.
- Buenas prácticas de codificación.
- Prototipado rápido y fácil.
- Facilidad de mantenimiento y actualización.
- Posibilidad de reutilización.
- Escalabilidad de los diseños.
- Documentación completa.
- Accesibilidad.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.1. Introducción a los 'frameworks' CSS.

Hay muchos frameworks CSS. En Internet se pueden encontrar muchas comparativas y clasificaciones en función de sus características, funcionalidades, complejidad, etc.

Destacan por encima de todos Bootstrap ([getbootstrap.com](https://getbootstrap.com)), Foundation([get.foundation](https://get.foundation)) y Bulma ([bulma.io](https://bulma.io)).

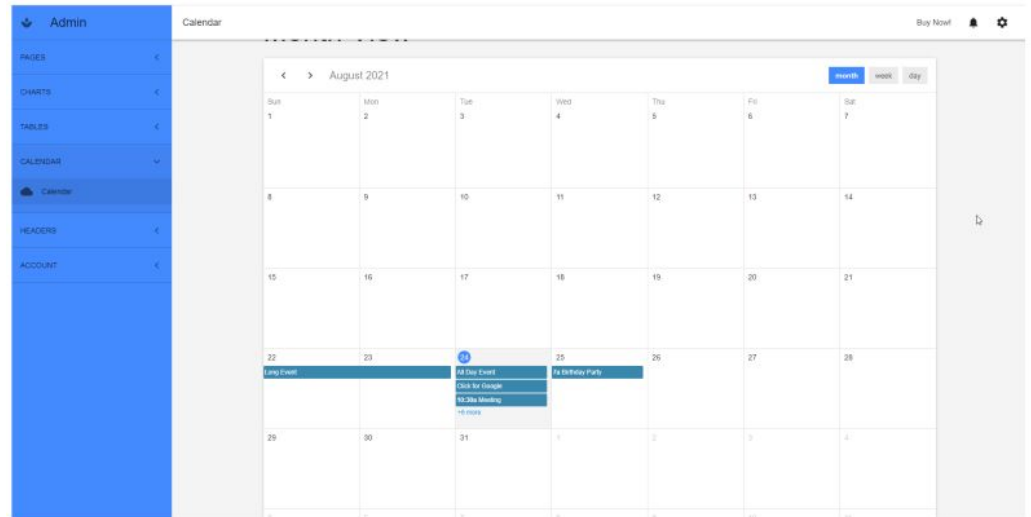
# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

De los otros entornos de trabajo podemos destacar:

**Materialize.** Basado en la filosofía de diseño material de Google, ayuda a los desarrolladores a construir y diseñar sitios web multiplataforma con el estilo típico de las interfaces de las aplicaciones web de Google.

web: [materializecss.com](https://materializecss.com)



# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

De los otros entornos de trabajo podemos destacar:

**Tailwind CSS.** Se caracteriza por defender el CSS basado en utilidades respecto al CSS semántico. Es un framework muy personalizable y configurable. Presenta cierta dificultad inicial de aprendizaje

web: [tailwindcss.com](https://tailwindcss.com)

### Multi-Column Layouts

Full-width three-column FIG PREVIEW

[Get the code →](#)

 workflow

 Dashboard

 Calendar

 Teams

 Directory

 Announcements

 Office Map

 Whitney Francis  
[View profile](#)

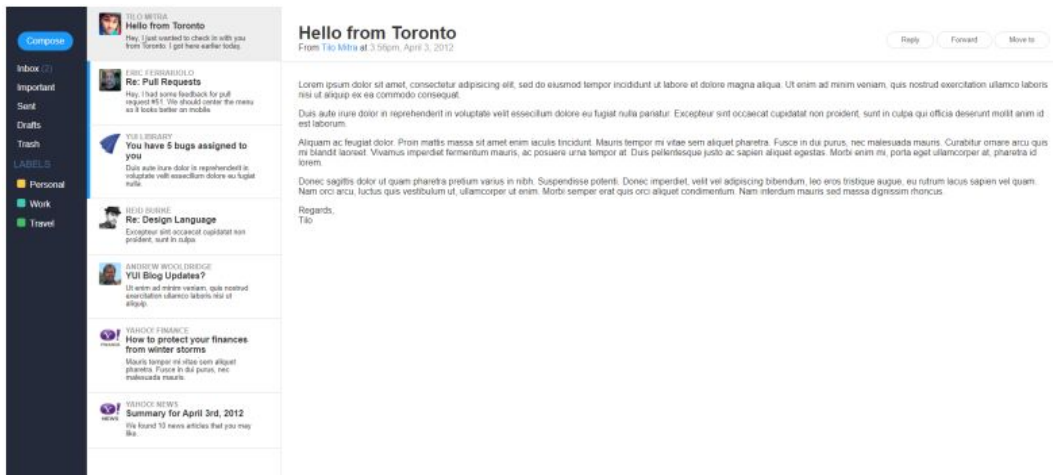
# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.1. Introducción a los 'frameworks' CSS.

De los otros entornos de trabajo podemos destacar:

**Pure CSS.** Desarrollado por Yahoo. Dispone de un conjunto reducido de módulos adaptativos CSS que se pueden integrar fácilmente en cualquier proyecto web, como por ejemplo el cliente de correo. Es sencillo (no tiene librerías Javascript, tan solo CSS), muy configurable y ligero.

web: [purecss.io](http://purecss.io)





## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2. El 'framework' CSS Bootstrap.

Fue desarrollado para Twitter, para mejorar las herramientas internas, y posteriormente se licenció en código abierto.

Consiguió mucha aceptación y una incipiente comunidad de usuarios. A día de hoy es el framework más popular en el mundo del diseño web y es utilizado por desarrolladores de todo el mundo.

Bootstrap adopta el paradigma *mobile-first* a la hora de construir webs adaptativas.

Proporciona un conjunto de componentes, módulos, funciones javascript y media queries que ayuda a los desarrolladores a construir webs con mucha facilidad.

La versión actual a día de hoy es 5.3.3

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2. El 'framework' CSS Bootstrap.

Para empezar a utilizar bootstrap hay 2 opciones:

1. Incluir Bootstrap 5 desde un CDN. Solamente se ha de incluir el siguiente código dentro del <head> del documento, antes de cualquier otra hoja de estilos.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOMLAsjC" crossorigin="anonymous">
```

También se ha de referenciar la funcionalidad Javascript, requerida por muchos componentes de Bootstrap. Esta referencia se pone al final de la página, justo antes de la etiqueta </body>

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYIzcLA8NI+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>
```

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2. El 'framework' CSS Bootstrap.

Para empezar a utilizar bootstrap hay 2 opciones:

2. Descargar Bootstrap 5 desde [getBootstrap.com](https://getbootstrap.com) e incluílo dentro del sitio web. La forma de referenciar es similar a la anterior pero haciendo referencia a los ficheros .css y .js del sistema local de archivos.

Descargamos la última versión de bootstrap y referenciamos los archivos de bootstrap del siguiente modo

Una vez hecho eso se ha de añadir la ruta donde está el css y el js.

```
<link rel="stylesheet" href="css/bootstrap.min.css">
```

```
<script src="js/bootstrap.bundle.min.js"></script>
```

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2.1. Sistema de cuadrícula.

Es una cuadrícula adaptativa de doce columnas. Se pueden utilizar las columnas de forma individual o expandirlas para crear grupos de medidas superiores.

Las columnas se organizan según la medida de la pantalla.

La forma de reorganización depende de las clases escogidas para cada resolución.

### Clases de cuadrícula.

**.col-.** Dispositivo *Extra Small*, resoluciones inferiores a 576px.

**.col-sm.** Dispositivos Small, resoluciones iguales o superiores a 576px.

**.col-md.** Dispositivos Medium, resoluciones iguales o superiores a 768px.

**.col-lg.** Dispositivos Large, resoluciones iguales o superiores a 992px.

**.col-xl.** Dispositivos EXtra Large, resoluciones iguales o superiores a 1200px.

**.col-xxl.** Dispositivos EXtra eXtra Large, resoluciones iguales o superiores a 1400px.

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2.1. Sistema de cuadrícula.

Combinando estas clases se pueden crear estructuras flexibles y dinámicas. Es importante tener en cuenta que, tal como están definidas las clases, para establecer la misma anchura para sm y md tan solo se ha de especificar sm.

### Funcionamiento de la cuadrícula.

Se basa en las siguientes pautas:

- Las filas han de incluir un contenedor, que centra y encuadra horizontalmente el contenido. Se puede utilizar la clase **.container** para tener un contenedor de amplitud fija pero adaptativa (la amplitud varía en función de la resolución del dispositivo). Con la clase **.container-fluid** se obtiene un contenedor con **width:100%** para todos los viewports y dispositivos, un contenedor fluido.

En Bootstrap 5 se incorpora el concepto de contenedor adaptativo, que consiste en utilizar las nuevas clases **.container-sm**, **.container-md**, **.container-lg**, **.container-xl**, **.container-xxl**. Estas clases tienen una amplitud del 100% hasta el punto de corte especificado en el sufijo y para resoluciones superiores adaptan una amplitud fija inferior a la anchura disponible.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.1. Sistema de cuadrícula.

| Classe           | Extra Small | Small   | Medium  | Large   | X-Large  | XX-Large |
|------------------|-------------|---------|---------|---------|----------|----------|
|                  | <576px      | >=576px | >=768px | >=992px | >=1200px | >=1400px |
| .container       | 100%        | 540px   | 720px   | 960px   | 1140px   | 1320px   |
| .container-sm    | 100%        | 540px   | 720px   | 960px   | 1140px   | 1320px   |
| .container-md    | 100%        | 100%    | 720px   | 960px   | 1140px   | 1320px   |
| .container-lg    | 100%        | 100%    | 100%    | 960px   | 1140px   | 1320px   |
| .container-xl    | 100%        | 100%    | 100%    | 100%    | 1140px   | 1320px   |
| .container-xxl   | 100%        | 100%    | 100%    | 100%    | 100%     | 1320px   |
| .container-fluid | 100%        | 100%    | 100%    | 100%    | 100%     | 100%     |

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.1. Sistema de cuadrícula.

- Las filas permiten crear grupos horizontales de columnas.
- El contenido se ha de ubicar dentro de las columnas y tan solo pueden haber columnas como descendientes directos de las filas.
- Las clases predefinidas como **.row** y **col-sm-2** ayuda a hacer facilmente y rapidamente estructuras de cuadrícula.
- Cada columna tiene un relleno (padding) horizontal llamado (gutter) para controlar el espacio entre columnas. Este espacio se compensa en las filas con márgenes negativos para garantizar que el contenido de las columnas se visualice alineado correctamente a la izquierda.
- Las columnas se crean especificando el número de las doce columnas disponibles que se quiere abarcar. Para tener 3 columnas iguales, se ha de indicar **.col-sm-4**.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.1. Sistema de cuadrícula.

- Los anchos de las columnas siempre son en porcentaje, fluidas y dimensionadas en relación con su elemento padre.
- Si no se especifica la amplitud de la columna, automáticamente se distribuyen de forma equitativa la amplitud de las columnas de la fila. Si tenemos 3 columnas con **.col-sm**, automáticamente cada una se ajusta a una amplitud del 33,33% desde el punto de corte sm.



# 5. Patrones de diseño web ad

## 3.2.1. Sistema de cuadrícula.

Ejemplo:

[https://docs.google.com/document/d/1PpVTK\\_R09\\_jCacs9KcbtmgBumUU0bY4DSuZS83HBwTw/edit?usp=sharing](https://docs.google.com/document/d/1PpVTK_R09_jCacs9KcbtmgBumUU0bY4DSuZS83HBwTw/edit?usp=sharing)

Responsive ▾ 992 x 1176 75% ▾ No throttling ▾

### Exemples de graelles Bootstrap

Estructures bàsiques amb graelles per iniciar-se al sistema de graella de Bootstrap

Redimensioneu la finestra del navegador per veure les diferències entre cada estructura

#### Els sis nivells de graella

Tenim sis nivells al sistema de graella, un per cada rang de dispositius suportats. Cada nivell comença a una mida mínima del viewport i automàticament s'aplica per a tot dispositiu superior si no és sobreescrit.

Començeu visualitzant aquest primer exemple a resolució petita, totes les columnes excepte les de la primera fila estaran apilades, a mida que anem fent més gran la finestra, s'aniran distribuint en tres columnes cada una de les files.

|            |           |           |
|------------|-----------|-----------|
| .col-4     | .col-4    | .col-4    |
| .col-sm-4  | .col-sm-4 | .col-sm-4 |
| .col-md-4  | .col-md-4 | .col-md-4 |
| .col-lg-4  | .col-lg-4 | .col-lg-4 |
| .col-xl-4  |           |           |
| .col-xl-4  |           |           |
| .col-xl-4  |           |           |
| .col-xxl-4 |           |           |
| .col-xxl-4 |           |           |
| .col-xxl-4 |           |           |

Tres columnes iguals

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.2. Navegación.

En bootstrap una barra de navegación se puede expandir o colapsar en función del ancho de la pantalla.

Par utilizar una barra de navegación de Bootstrap:

- Se utiliza la clase `.navbar` como envoltorio con las clases `.navbar-expand-sm/-md/-lg/-xl/-xxl` para indicar a los dispositivos donde poner la barra expandida.
- Los contenidos de una `navbar` son fluidos por defecto. Para limitar su amplitud, se ha de modificar el contenedor de la página.
- El espaciado y alineación de las barras de navegación se controla con las utilidades de Bootstrap `spacin` y `flex`.
- Las barras de navegación de Bootstrap son adaptativas por defecto.

Ver ejemplo [codepen.io/iocdawm9/pen/JjJowyw](https://codepen.io/iocdawm9/pen/JjJowyw)

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.2. Navegación.

Es interesante observar el comportamiento de la barra de navegación a resoluciones inferiores a 992px, se colapsa y adopta el estilo de *hamburguer*. Este comportamiento lo provoca la clase *.navbar-expand-lg*.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.3. Botones.

Bootstrap proporciona muchos estilos predefinidos de botones. Muchos botones tienen semánticas predefinidas, como notificar el éxito de una acción, hacer una advertencia, avisar de un peligro etc.

Para utilizarlo hay *classes btn*, que se pueden combinar con los diferentes tipos: *btn-primary*, *btn-success*, *btn-dark*, etc.

También hay clases para hacer los mismos botones pero tan solo con la orilla de color. *btn-outline-primary*, *btn-outline-success*, etc.

Todas estas clases están pensadas para utilizarlas con el elemento HTML button, pero también se pueden utilizar con elementos de tipo *input*.

Ver ejemplo en: [codepen.io/iocdawm9/pen/BaZyvqp](https://codepen.io/iocdawm9/pen/BaZyvqp)

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.4. Formularios.

Los controles de un formulario reciben automáticamente algunas características globales de estilo de Bootstrap. Todos los elementos `<inputs>` textuales, *textareas* y los elementos *select* con la clase *.form-control* tienen una amplitud del 100%.

Es importante utilizar los atributos apropiados *type* de todos los elementos *input* (*email* para direcciones de correo electrónico o *number* para información numérica), así se pueden aprovechar las funcionalidades de control de entrada de texto como la verificación de correo, la selección de número y otras.

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.4. Formularios.

En un formulario se pueden utilizar las classes de cuadrícula de Bootstrap para controlar la disposición de los diferentes elementos del formulario.

[codepen.io/iocdawm9/pen/eYRmxMq](https://codepen.io/iocdawm9/pen/eYRmxMq)

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2.5. Tipografías.

Bootstrap utiliza una lista de fuentes nativas que selecciona la mejor *font-family* para cada sistema operativo y dispositivo.

En cuanto a las escalas, utiliza la *font-size* por defecto del elemento root del navegador (normalmente 16px).

Todos los elementos tienen *margin-top: 0* y *margin-bottom: 1rm*

Define todas las cabeceras hasta el nivel 6.

| Heading   | Example                      |
|-----------|------------------------------|
| <h1></h1> | <b>h1. Bootstrap heading</b> |
| <h2></h2> | <b>h2. Bootstrap heading</b> |
| <h3></h3> | <b>h3. Bootstrap heading</b> |
| <h4></h4> | <b>h4. Bootstrap heading</b> |
| <h5></h5> | <b>h5. Bootstrap heading</b> |
| <h6></h6> | <b>h6. Bootstrap heading</b> |

# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2.5. Tipografías.

También hay disponibles las clases .h1 hasta el .h6 por si se ha de utilizar este estilo en elementos diferentes como un <p>.

La siguiente sintaxis genera el mismo resultado:

```
1 <h1>Encapçalament h1</h1>
```

```
1 <p class="h1">Encapçalament h1</p>
```



# 5. Patrones de diseño web adaptativo. Frameworks web

## 3.2.5. Tipografías.

A partir de los encabezados estándares o de HTML, Bootstrap incorpora el **display headings**, que son más grandes y con una tipografía más ligera que los encabezados tradicionales.

```
1 <h1 class="display-1">Display 1</h1>
2 <h1 class="display-2">Display 2</h1>
3 <h1 class="display-3">Display 3</h1>
4 <h1 class="display-4">Display 4</h1>
5 <h1 class="display-5">Display 5</h1>
6 <h1 class="display-6">Display 6</h1>
```

Ejemplo:

[codepen.io/iocdawm9/pen/NWgqwZP](https://codepen.io/iocdawm9/pen/NWgqwZP)

Display 1

Display 2

Display 3

Display 4

Display 5

Display 6

## 5. Patrones de diseño web adaptativo. Frameworks web

### 3.2.6. Plantilla básica con Bootstrap.

A partir de un cuadrícula flexible CSS, creamos una plantilla básica implementada con clases de Bootstrap.

Esta plantilla básica se implementa con clases de Bootstrap.

Modificaremos un poco la disposición del ejemplo original y pondremos la barra de navegación en la cabecera, dejando más espacio disponible para el contenido.

Utilizaremos las clases del sistema de cuadrícula de Bootstrap para facilitar en todo momento la legibilidad del contenido, A resoluciones inferiores a 768px la barra de navegación se colapsará y la barra lateral se apilará, dejando que la zona de contenido ocupe todo el ancho disponible.

Código: [codepen.io/iocdawm9/pen/gORaOgz](https://codepen.io/iocdawm9/pen/gORaOgz)