

**UD2**

# Hojas de estilo

Carlos Millán  
IES Conselleria



The background is an abstract geometric pattern composed of numerous triangles of varying sizes. The colors transition from a warm orange on the left to a cool blue on the right, with a pale yellow/white area at the top. The word "INTRODUCCIÓN" is centered in the middle of the image.

# **INTRODUCCIÓN**

# HOJAS DE ESTILO CSS3

- Las hojas de estilo en cascada son un conjunto de reglas para indicarle al navegador como se ha de mostrar la página.
- Están reguladas por el W3C

# HOJAS DE ESTILO CSS3

## → Ventajas:

- Separa contenido de visualización
- Variantes de visualización de una web
- Unificación y reutilización
- Estándar W3C

# INCLUIR CSS3 EN HTML

## → Estructura de un documento CSS

```
/* Comentari del document CSS */  
  
h1 {  
    color: #FFFFFF;  
    background-color: #999999;  
}  
  
p {  
    font-family: Georgia, Helvetica;  
}
```

# INCLUIR CSS3 EN HTML

→ Incrustados en el HTML (o en la etiqueta)

```
<head>
...
<style type="text/css" media="screen">
  body { background: url(foo.gif) red; color: black; }
  p em { background-color: yellow; color: black; }
  .nota { margin-left: 5em; margin-right: 5em; }
</style>
</head>
```

# INCLUIR CSS3 EN HTML

→ Enlazado a documento CSS externo:

- <link>

```
<html>
<head>
  <link rel="stylesheet" href="./css/estils.css" />
  ...
</head>
...
</html>
```

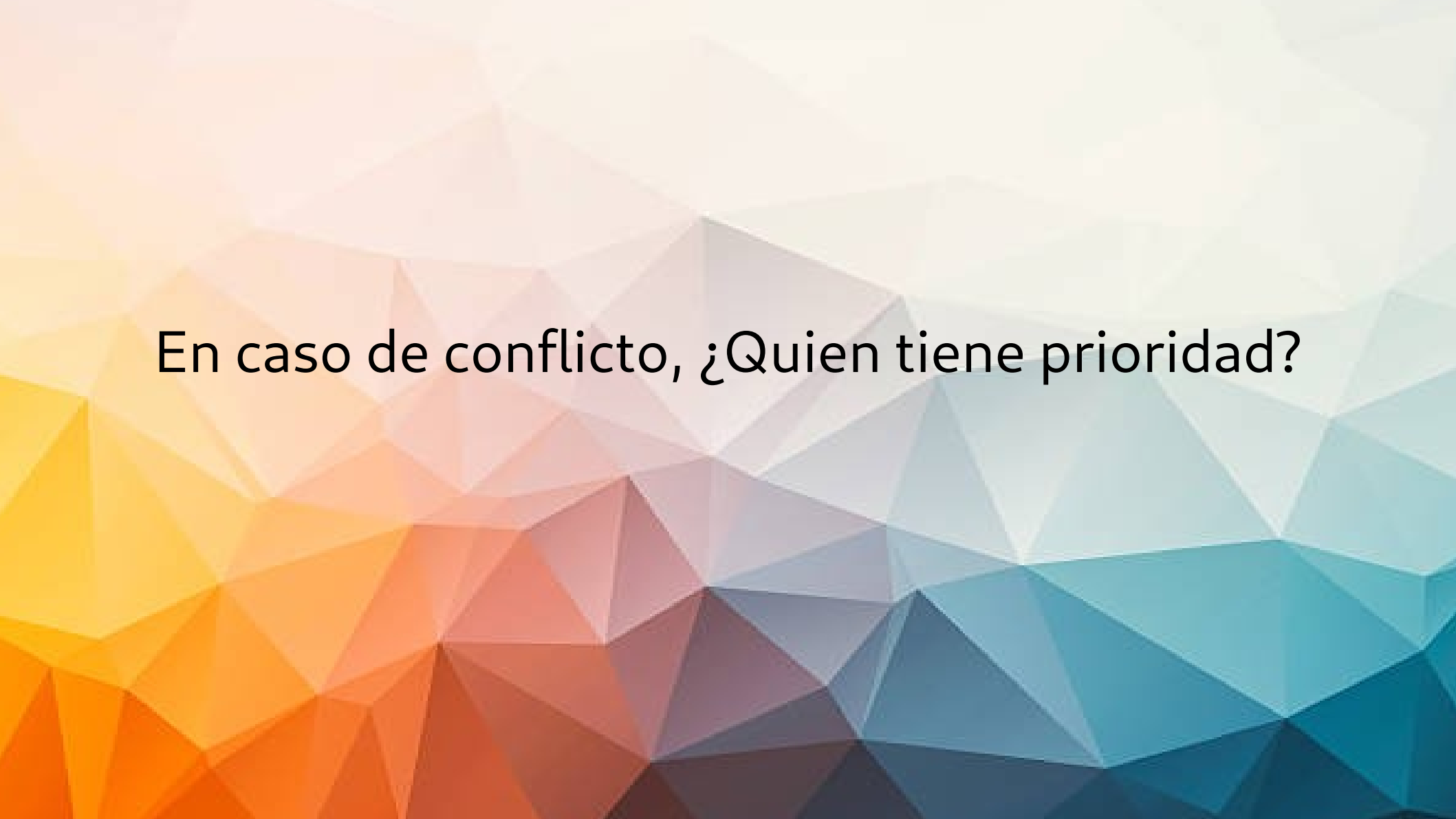
- @import

```
<style type="text/css" media="screen, projection">
  @import url(http://elmeudomini.com/css/estils.css);
  @import url(/css/estils2.css);
  p { background-color: yellow; color: black; }
</style>
```

# ESTILOS EN CASCADA

- Un documento HTML puede utilizar más de un documento CSS para definir su visualización.
- De hecho, se combinan los estilos inline, los de los la etiqueta `<style>`, los enlazados por las etiquetas `<link>...`



The background is an abstract geometric pattern composed of numerous overlapping triangles. The color gradient transitions from a warm orange on the left side to a cool blue on the right side, with a neutral white and light grey area at the top. The text is centered horizontally and vertically within the image.

En caso de conflicto, ¿Quién tiene prioridad?

# ESTILOS EN CASCADA

- El último estilo definido es el que tiene prioridad.
- Estilos del navegador < Estilos en documentos  
CSS < Estilos inline <http://codepen.io/ioc-daw-m09/pen/pyrpER>

# HERENCIA EN LOS ESTILOS

- La herencia es la capacidad de los elementos del documento HTML de heredar propiedades heredables\* de sus antecesores.
- Todos los elementos HTML están dentro de otra etiqueta (menos `<html>`) y de ella heredan todos sus estilos.

<http://codepen.io/ioc-daw-m09/pen/yOopMb>

\* <https://www.w3.org/TR/css-2010/#properties>

# FORMATO DE UNA REGLA CSS

```
selector { propiedad: valor; propiedad: valor; ... }
```

- Donde selector hace referencia al elemento al que se aplica el estilo.
- Las diferentes propiedades se separan con «;»

```
h1 {font-size:10px; color:blue; text-align:center;}
```

# SELECTORES

- \*
- NombreElementoHTML
- .nombreClase
- #idElemento
- selector[atributo]

<http://codepen.io/ioc-daw-m09/pen/pyrBaP>

# SELECTORES

- `[atribut]` Selecciona el elemento si tiene el atributo
- `[atribut="valor"]` Selecciona el elemento si el atributo es valor
- `[atribut^="text"]` Selecciona el elemento si el atributo empieza por text
- `[atribut$="text"]` Selecciona el elemento si el atributo termina por text
- `[atribut*="text"]` Selecciona el elemento si el atributo contiene text
- `[atribut~="valor"]` Selecciona el elemento si tiene un atributo separado por espacios y una de las palabras es text.
- `[atribut|="valor"]` Selecciona el elemento si tiene un atributo separado por guiones y una de las palabras es text.

[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sel\\_attribute\\_value2](https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_attribute_value2)

[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sel\\_attribute\\_hyphen](https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_attribute_hyphen)

# ACTIVIDAD

- Diseña un selector css que seleccione únicamente los enlaces a aules.edu y los ponga del color verde predominante en dicha web.

# COMBINACIÓN DE SELECTORES

- **A + B** Selectores adyacentes. Permite aplicar un estilo que se encuentre a continuación (B) de un elemento específico (A).
- **A ~ B** Selector de hermanos. Permite aplicar un estilo que sea hermano (B) de un elemento específico (A).
- **A > B** Selector de hijos. Permite aplicar un estilo que sea el primer hijo (B) de un elemento específico (A)
- **A B** Selectores descendentes. Permite aplicar un estilo que se encuentre dentro (B) de un elemento específico (A).



# ACTIVIDAD

- Genera el CSS necesario para obtener el siguiente resultado a partir del HTML proporcionado (selectores.html)

teal	teal	teal	teal
red		red	red
blue	blue	blue	
orange	orange	orange	orange
purple	purple	purple	purple
olive	olive	olive	olive
fuchsia	fuchsia	fuchsia	fuchsia
green	green		green

# PSEUDOCCLASES

→ Las pseudoclasses permiten seleccionar un elemento cuando cumple una determinada condición:

- |                                 |  |
|---------------------------------|--|
| - <code>selector:link</code>    | Enlace no visitado                     |
| - <code>selector:visited</code> | Enlace visitado                        |
| - <code>selector:active</code>  | Elemento activado (se clicca sobre él) |
| - <code>selector:hover</code>   | El ratón pasa sobre ellos              |
| - <code>selector:focus</code>   | Recibe el foco                         |

<https://codepen.io/ioc-daw-m09/pen/MyEWvW>

# PSEUDOCCLASES

- <code>selector:root</code>	Se encuentra a la raíz del documento.
- <code>selector:nth-child (num)</code>	Hijo enésimo de su padre.
- <code>selector:nth-last-child (num)</code>	Hijo enésimo de su padre (desde el final)
- <code>selector:nth-of-type (num)</code>	Hijo enésimo de ese tipo
- <code>selector:nth-last-of-type (num)</code>	Hijo enésimo de ese tipo (desde el final)
- <code>selector:first-child</code>	Primer hijo
- <code>selector:last-child</code>	Último hijo
- <code>selector:first-of-type</code>	Primer elemento de ese tipo
- <code>selector:last-of-type</code>	Último elemento de ese tipo
- <code>selector:only-of-type</code>	Único hijo
- <code>selector:empty</code>	Elemento vacío
- <code>selector:lang(es)</code>	Idioma

# ACTIVIDAD

- Diseña un selector css que se pueda utilizar para pintar de un color diferente las filas impares de una tabla.

# PSEUDOELEMENTOS

→ Permiten seleccionar solo parte del selector

- |                                       |  |
|---------------------------------------|--|
| - <code>selector::after</code>        | Añade texto después o antes del contenido. Hay que incluir la propiedad <code>content</code> . |
| - <code>selector::before</code>       |  |
| - <code>selector::first-letter</code> | Primera letra  |
| - <code>selector::first-line</code>   | Primera línea  |
| - <code>selector::selection</code>    | Selección  |

<http://codepen.io/ioc-daw-m09/pen/YqExdd?editors=1100>

# UNIDADES DE MEDIDA CSS

→ Cuando se trata de poner medidas en las propiedades CSS hay dos opciones:

- **Medidas absolutas:** cm, mm, in, pt
- **Medidas relativas:**
  - Respecto tamaño de otros elementos: em, rem, px, ex, ch, %
  - Respecto a la ventana (viewport): vw, vh, vmin, vmax

# ACTIVIDAD

- Diseña un hoja de estilos css para que en un documento, todos los párrafos empiecen por una letra capital, haya un selector copyright que añada el símbolo automáticamente antes del texto contenido y que el tamaño de la letra se adapte al tamaño de la pantalla.

# PROPIEDADES CSS

- Existen gran variedad de propiedades CSS para los elementos.
- No todos los navegadores soportan todos las propiedades, y algunos las soportan de forma experimental







# PROPIEDADES CSS

- **Fuente:** `color, font-size, font-family, font-weight, font-style`
- **Párrafos:** `line-height, text-decoration, text-align, text-indent, text-transform, text-shadow, text-overflow, text-wrap, list-style`
- **Fondo :** `background-color, background, background-image, background-origin, background-repeat, background-position, background-attachment`
- **Tablas:** `border-spacing, border-collapse, caption-side, empty-cells`

[https://developer.mozilla.org/es/docs/Web/CSS/Referencia\\_CSS](https://developer.mozilla.org/es/docs/Web/CSS/Referencia_CSS)

# PROPIEDADES EXPERIMENTALES

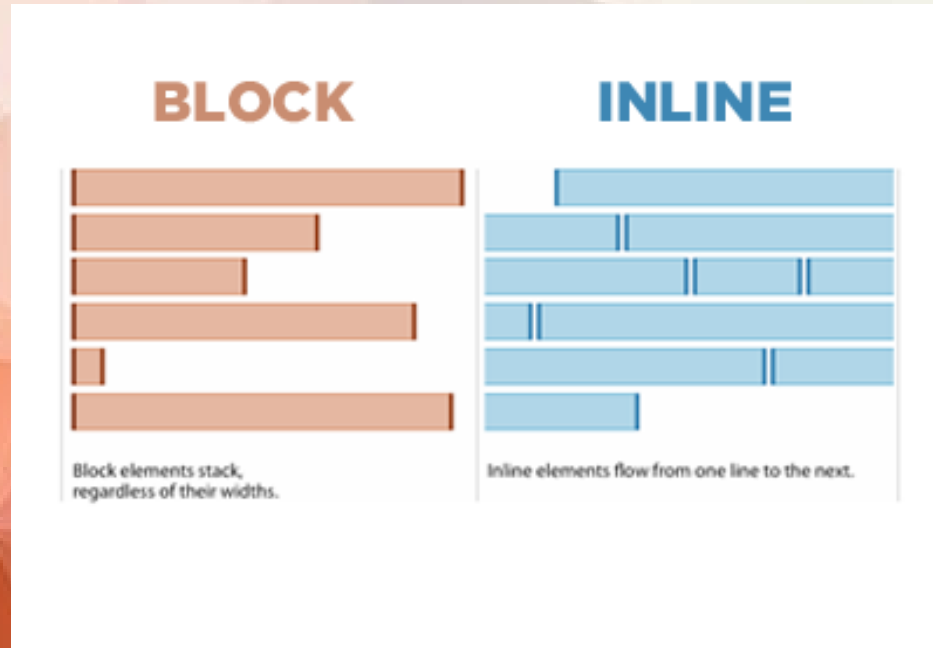
Prefijo	Familia de navegadores a los que aplica	
-webkit-	Chrome, Safari, Android, iOS	
-moz-	Firefox	
-o-	Opera	
-ms-	Microsoft Internet Explorer	

The background is an abstract geometric pattern composed of numerous overlapping triangles. The color gradient transitions from a warm orange on the left side to a cool blue on the right side, with a neutral white and light grey area at the top. The triangles vary in size and orientation, creating a complex, crystalline texture.

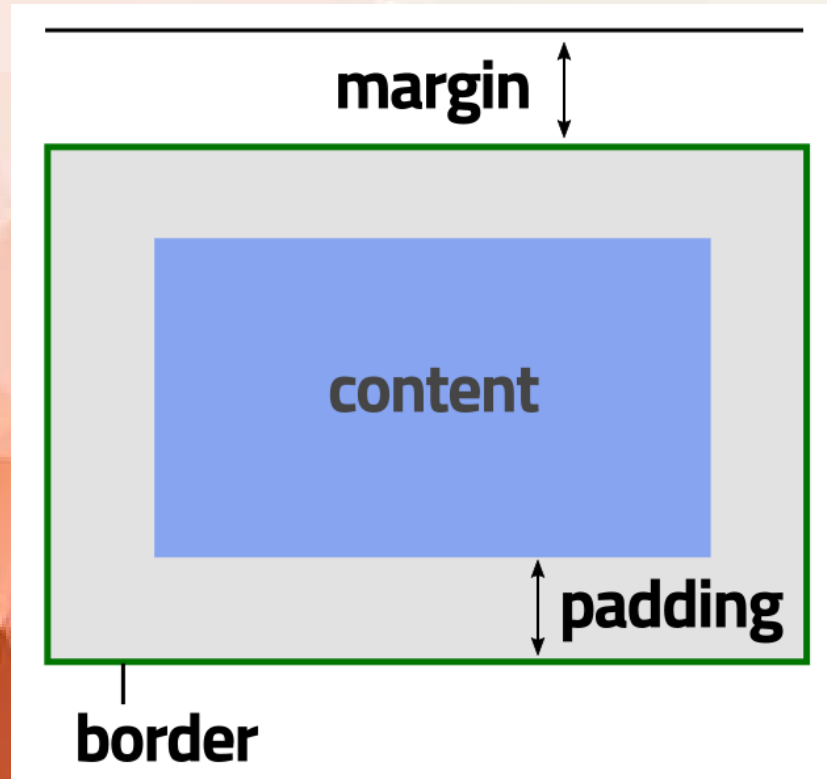
# **PRÁCTICA 2.1**

# EL MODELO DE CAJA

→ *Block vs Inline*



# PROPIEDADES DE UNA CAJA



# PROPIEDADES DE UNA CAJA

- **Márgen:** `margin-left`, `margin-right`, `margin-top`, `margin-bottom`, `margin`
- **Fondo:** `background-color`, `background-image`
- **Grosor del borde:** `border-top-width`, `border-right-width`, `border-bottom-width`, `border-left-width`, `border-width`
- **Color del borde:** `border-top-color`, `border-right-color`, `border-bottom-color`, `border-left-color`, `border-color`
- **Relleno:** `padding-left`, `padding-right`, `padding-top`, `padding-bottom`, `padding`
- **Redondeado de esquinas:** `border-radius`

Si tengo siguiente regla:

```
.caja {  
  width: 100px;  
  height: 100px;  
  padding: 50px;  
  border: 10px;  
}
```

¿Cuánto mide en realidad la caja?

# PROPIEDADES DE UNA CAJA

- A la hora de definir el tamaño de una caja, por defecto, no se tiene en cuenta el relleno ni la grosor del borde, si queremos que se tenga en cuenta hay que hay que marcar la propiedad `box-sizing: border-box;`



# ACTIVIDAD

→ Diseña las siguientes cajas:

Mensaje

# ACTIVIDAD

→ Diseña la siguiente caja

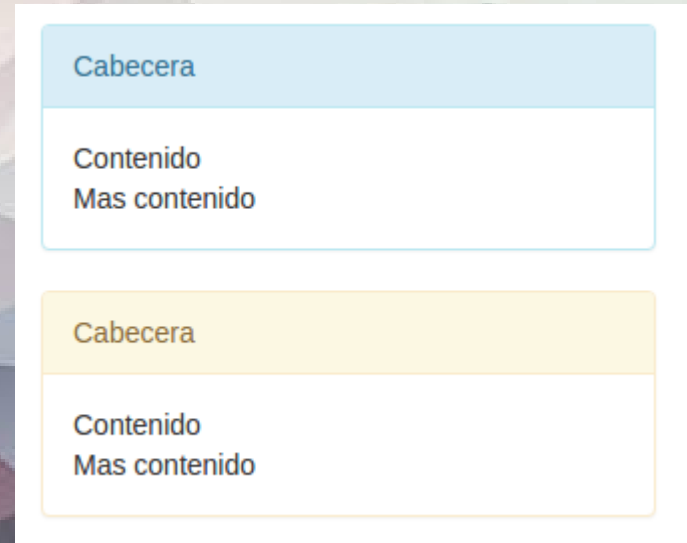
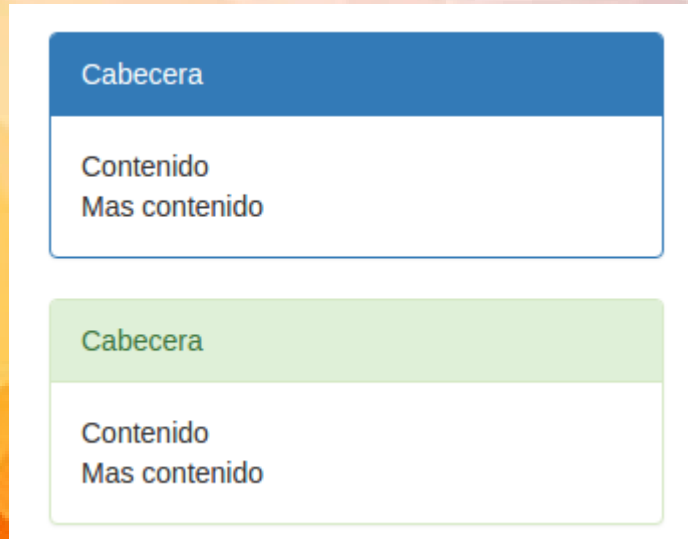
Cabecera

Contenido

Mas contenido

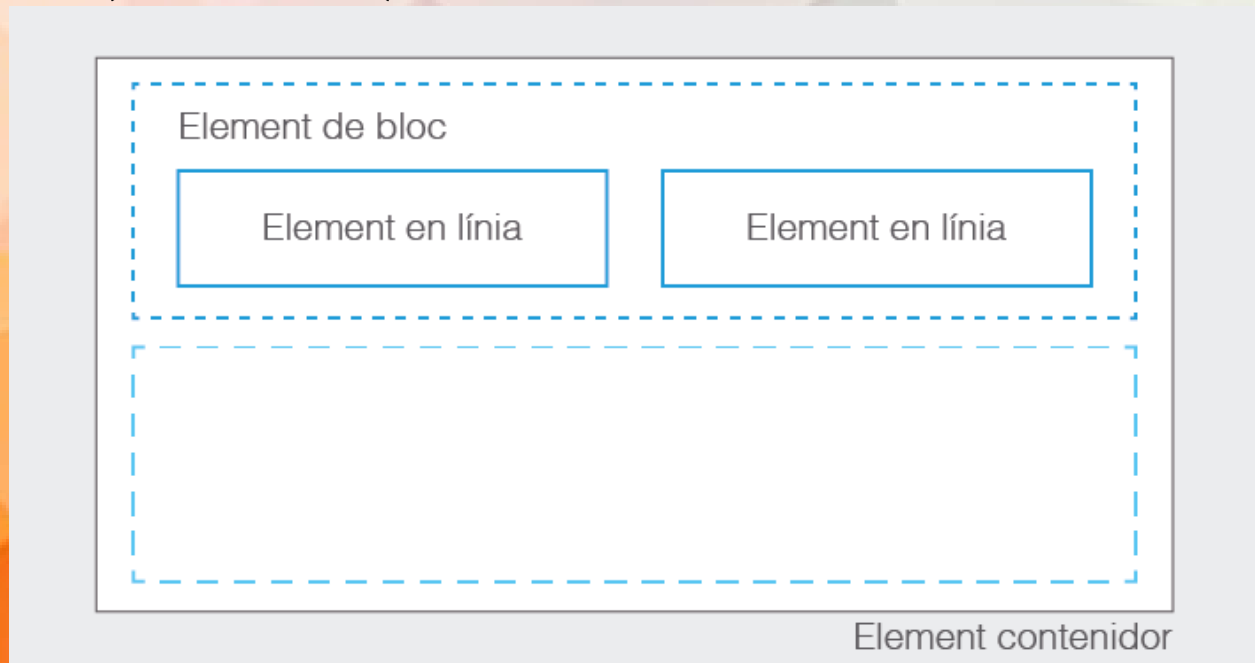
# ACTIVIDAD

→ ¿Cómo modificarías el CSS anterior para obtener variantes de varios colores?



# POSICIONAMIENTO ESTÁTICO

- Por defecto, el posicionamiento de una caja es **estático (fixed)**

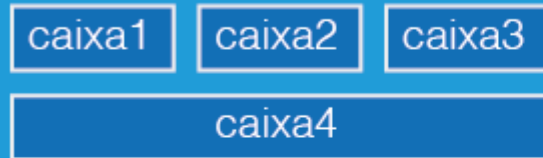


# POSICIONAMIENTO RELATIVO

- Con el posicionamiento relativo (`relative`), la caja se puede desplazar con las propiedades `top`, `bottom`, `right` y `left`

# POSICIONAMIENTO RELATIVO

Caixa contenedora: les caixes estan en posició estàtica



Caixa contenedora: les caixes estan en posició estàtica, excepte la caixa2, que està en posició relativa



# POSICIONAMIENTO ABSOLUTO

- En el posicionamiento absoluto (`absolute`) las cajas se posicionan con las propiedades `top`, `bottom`, `right` y `left`
- Se toma como referencia el contenedor o en su defecto la ventana
- Este posicionamiento «rompe» el flujo del diseño, lo que provoca que otros elementos se muevan

# POSICIONAMIENTO ABSOLUTO

Caixa contenedora: les caixes estan en posició estàtica

caixa1

caixa2

caixa3

caixa4

Caixa contenedora: les caixes estan en posició estàtica, excepte la caixa2, que està en posició absoluta

caixa1

caixa3

caixa2

caixa4

<http://codepen.io/ioc-daw-m09/pen/aNVEVy>



# POSICIONAMIENTO FIJO

- En el posicionamiento fijo (fixed), la posición de la caja permanece inmutable, independientemente del resto de elementos
- Las cajas se posicionan con las propiedades `top`, `bottom`, `right` y `left` respecto a la ventana
- También se «rompe» el flujo

# POSICIONAMIENTO FIJO

Caixa contenedora: les caixes estan en posició estàtica

caixa1

caixa2

caixa3

caixa4

Caixa contenedora: les caixes estan en posició estàtica, excepte la caixa2, que està en posició fixa

caixa1

caixa3

caixa2

caixa4

# POSICIONAMIENTO FLOTANTE

- El posicionamiento flotante (float) desplaza la caja a la izquierda (o la derecha) del contenedor, rompiendo el flujo de la página.
- Sin embargo, se establece un nuevo flujo entre todas las cajas flotantes, que se posicionan una junto a la otra.

# POSICIONAMIENTO FLOTANTE

Caixa contenidora: les caixes estan en posició estàtica, excepte la caixa1, que té un posicionament flotant a la dreta

caixa2

caixa1

caixa3

Caixa contenidora: les caixes estan en posició estàtica, excepte la caixa2, que té posició flotant a l'esquerra

caixa1

caixa2

caixa3

Caixa contenidora: totes les caixes estan en posició flotant a l'esquerra

caixa1

caixa2

caixa3

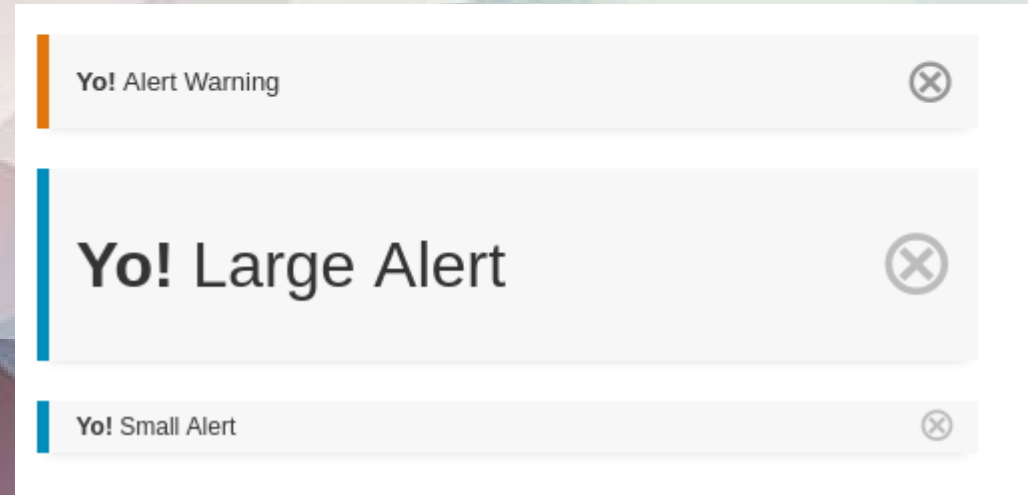
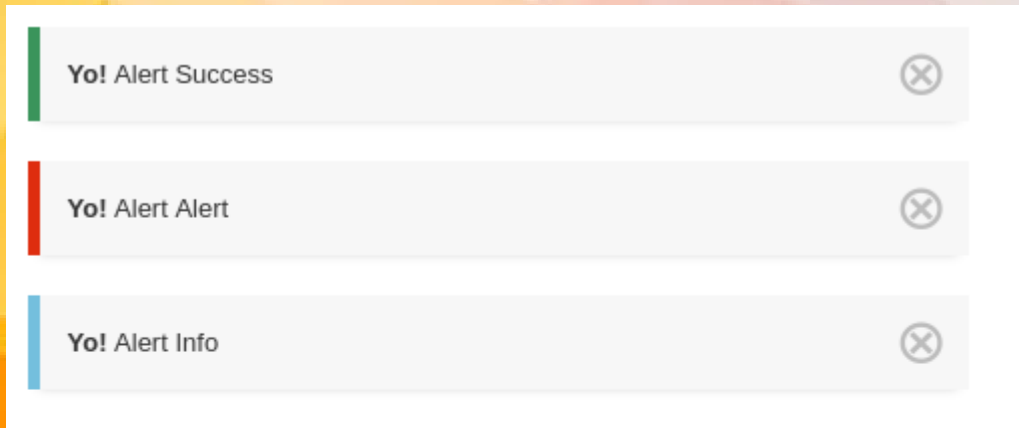
<http://codepen.io/ioc-daw-m09/pen/oxopQM>

# POSICIONAMIENTO FLOTANTE

- La propiedad `clear` permite definir si un elemento puede estar junto a elementos flotantes (o saltar a la siguiente línea).
- Se puede aplicar a elementos flotantes o no flotantes.

# ACTIVIDAD

- Diseña un juego de cajas informativas flotantes similares a las de las imágenes.



# VISUALIZACIÓN

- Cuando un navegador interpreta HTML, dibuja una caja (inline o block) para cada elemento, y aplica las propiedades que hemos visto hasta ahora para definir dimensiones, bordes, posición...
- CSS tiene una serie de propiedades para definir la manera en que se visualizarán las cajas

# VISUALIZACIÓN

- **display:** inline, block, none, list-item, run-in, inline-block, table, inline-table, table-row-group, table-header-group, table-footer-group, table-row, table-column-group, table-column, table-cell, table-caption, inherit
- **visibility:** visible, hidden, collapse, inherit
- **overflow:** visible, hidden, scroll, auto, inherit
- **z-index:** auto, numero, inherit



# VISUALIZACIÓN

## → Display vs Visibility

```
caixa2. display:none  
caixa7. visibility:hidden
```

caixa1

caixa3

caixa4

caixa5

caixa6

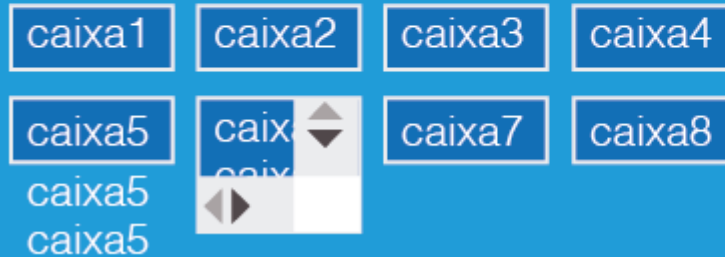
caixa8

<http://codepen.io/ioc-daw-m09/pen/ONQqMg>

# VISUALIZACIÓN

- La propiedad `overflow` permite definir como se comporta la caja con los contenidos que sobresalen.

```
caixa1. overflow:hidden  
caixa5. overflow:visible  
caixa6. overflow:scroll
```



# VISUALIZACIÓN

- La propiedad `z-index` define la profundidad de un elemento (cuanto más alto el valor, más arriba en el eje Z se posiciona). Es útil cuando varias cajas se solapan.

caixa1 caixa1 caixa1 caixa1 caixa1  
caixa1 caixa1 caixa1 caixa1 caixa1

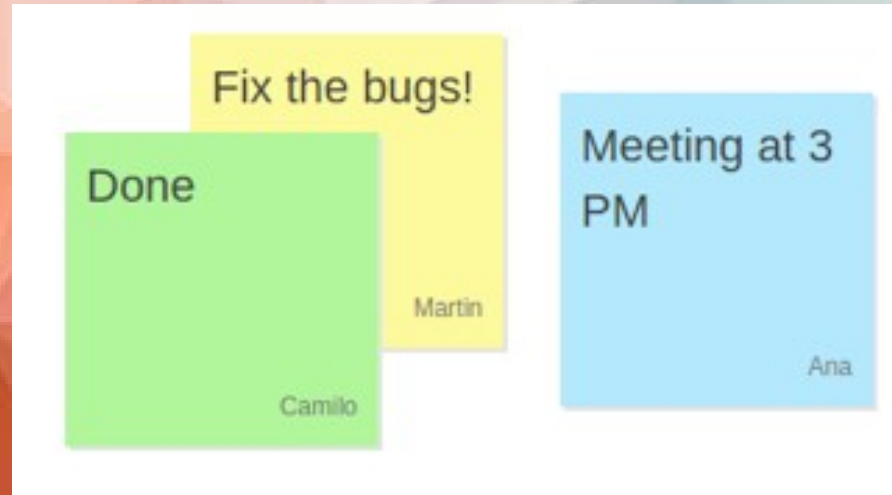
caixa1 caixa2 caixa2 caixa2 caixa2 caixa2  
caixa1 caixa2 caixa2 caixa2 caixa2 caixa2

caixa3 caixa3 caixa3 caixa3 caixa3  
caixa3 caixa3 caixa3 caixa3 caixa3  
caixa3 caixa3 caixa3 caixa3 caixa3  
caixa3 caixa3 caixa3 caixa3 caixa3  
caixa3 caixa3 caixa3 caixa3 caixa3  
caixa3 caixa3 caixa3 caixa3 caixa3

caixa2 caixa2 caixa2  
caixa2 caixa2 caixa2  
caixa2 caixa2 caixa2

# ACTIVIDAD

- Diseña una clase para simular las típicas notas adhesivas cuadradas para conseguir un efecto como el de la imagen.



The background is an abstract geometric pattern composed of numerous overlapping triangles. The color gradient transitions from a warm orange on the left side to a cool blue on the right side, with a neutral white and light grey area at the top. The triangles vary in size and opacity, creating a layered, crystalline effect.

# **PRÁCTICA 2.2**

# DISEÑO WEB ADAPTATIVO

- El creciente uso de dispositivos móviles ha provocado que el diseño de webs haya de tener en cuenta varios tamaños de pantalla en su diseño.
- Mobile First

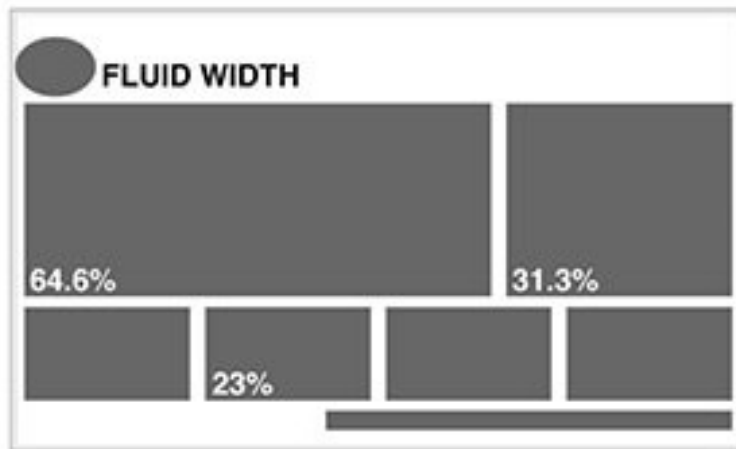
<https://www.apple.com/>

# DISEÑO WEB ADAPTATIVO

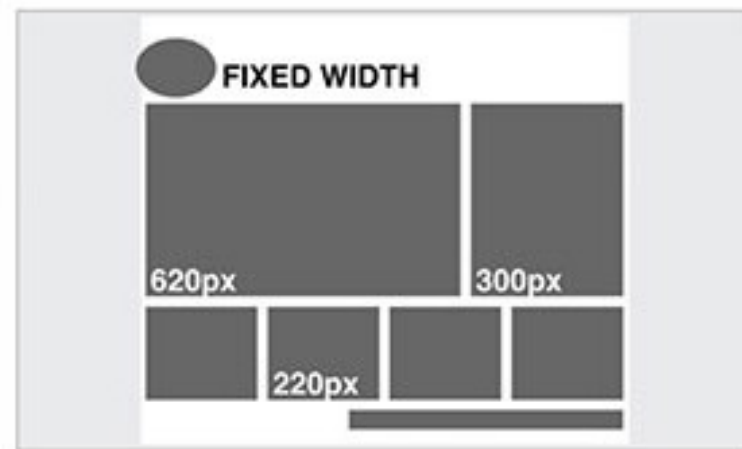
- Se define el diseño adaptativo o, en inglés, *Responsive Web Design*, como una técnica de diseño y desarrollo web que permite que un sitio web se vea correctamente en todos los tipos de dispositivos: ordenadores de sobremesa, portátiles, tabletas, teléfonos móviles, televisores, etc.

# LIQUID VS AWD VS RWD

- En un diseño líquido o fluido el contenido se ajusta horizontalmente a la medida de la pantalla sin necesidad de la barra de desplazamiento horizontal (*scroll*).



Fluid Width: Content spanning the entire page

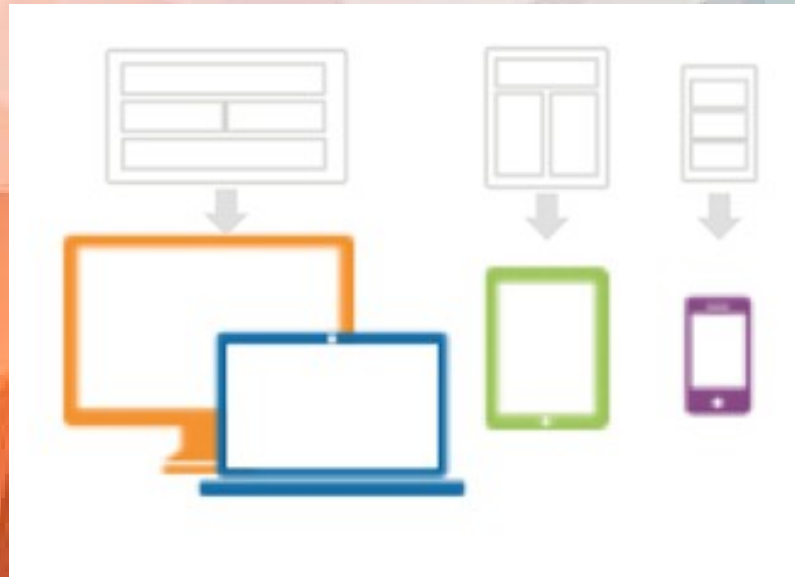


Fixed Width: Content remains centered



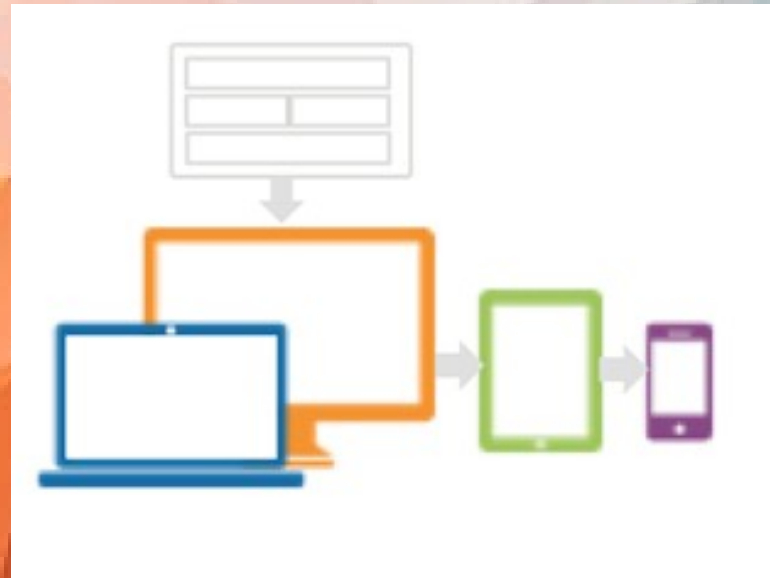
# LIQUID VS AWD VS RWD

- *Adaptative Web Design* o diseño web adaptativo se usan plantillas estáticas específicas para cada pantalla que se seleccionan en función del dispositivo.



# LIQUID VS AWD VS RWD

- *Responsive Web Design* o diseño web responsivo los contenidos se organizan en bloques que se reorganizan en función de la pantalla.



# MEDIA QUERIES

- Un *media query* es una consulta que da información sobre las características de la pantalla en la que se está visualizando la web
- Se puede hacer en el head del html

```
<link rel="stylesheet" media="screen and  
(width:780px)" href="estils780.css" />
```

- O en el CSS

```
@media screen and (width:780px) {  
    ...  
}
```

# MEDIA QUERIES

*IF THE SCREEN IS THIS SIZE*

`@media (max-width: 768px) {`

`.example-class {  
 margin: 0 auto !important;  
 padding: 0 !important;  
}`

*THEN THIS IS WHAT THIS CLASS  
SHOULD DO*

# MEDIA QUERIES

## → Valores:

- `screen` (valor por defecto): para presentación en pantallas de ordenadores
- `print` : para la salida por una impresora.
- `speech` : para sintetizadores de voz.
- `all` : para todos los dispositivos de salida.

# MEDIA QUERIES

→ Criterios:

Criterio	Valor	Mín./ Màx.	Descripción
width	Número	Sí	Para examinar la anchura de la zona de visualización del navegador.
height	Número	Sí	Para examinar la altura de la zona de visualización del navegador
device-width	Número	Sí	Para examinar la anchura física de la pantalla de difusión
device-height	Número	Sí	Para examinar la altura física de la pantalla de difusión.
orientation	<i>Landscape o portrait</i>	No	Para examinar si el usuario utiliza tableta táctil verticalmente, portrait u horizontalmente, landscape.
aspect-ratio	Ràtio	Sí	Para examinar el coeficiente ancho/alto.
device-aspect-ratio	Ràtio	Sí	Para examinar el coeficiente físico ancho/alto de la pantalla.

# MEDIA QUERIES

- Se pueden redactar consultas utilizando operadores lógicos como son `not`, `and` y `only`. Además, las listas separadas por comas (",") se comportan como el operador `or`.

# MEDIA QUERIES

## → Ejemplos:

- `@media (min-width: 700px) and (orientation: landscape) { ... }`
- `@media print (width: 1024px) or (width: 860px) { ... }`
- `@media screen and (not width: 780px) { ... }`



# ACTIVIDAD

- Diseña un CSS con media queries que cambien el color de fondo de la web en función de la anchura de visualización:
- Entre 1280-1024: verde
  - Entre 1024-480: naranja
  - Menos de 480: rojo

# VIEWPORT

- El **viewport** es el área visible del navegador. En los navegadores para móviles no se corresponde a la medida real de la pantalla en píxeles, sino al espacio que la pantalla está emulando que tiene
- Por ejemplo, si una pantalla móvil tiene un ancho 640px, las páginas pueden ser procesadas con un viewport de 980px, y después comprimidas para que entren en 640px.

# VIEWPORT

- Para evitar esta situación, se puede alterar el **viewport** del navegador a través de un elemento `meta` del `head`.

```
<head>  
...  
<meta name="viewport" content="width=device-width, initial-scale=1.0" />  
...  
</head>
```

# VIEWPORT

## → Parámetros:

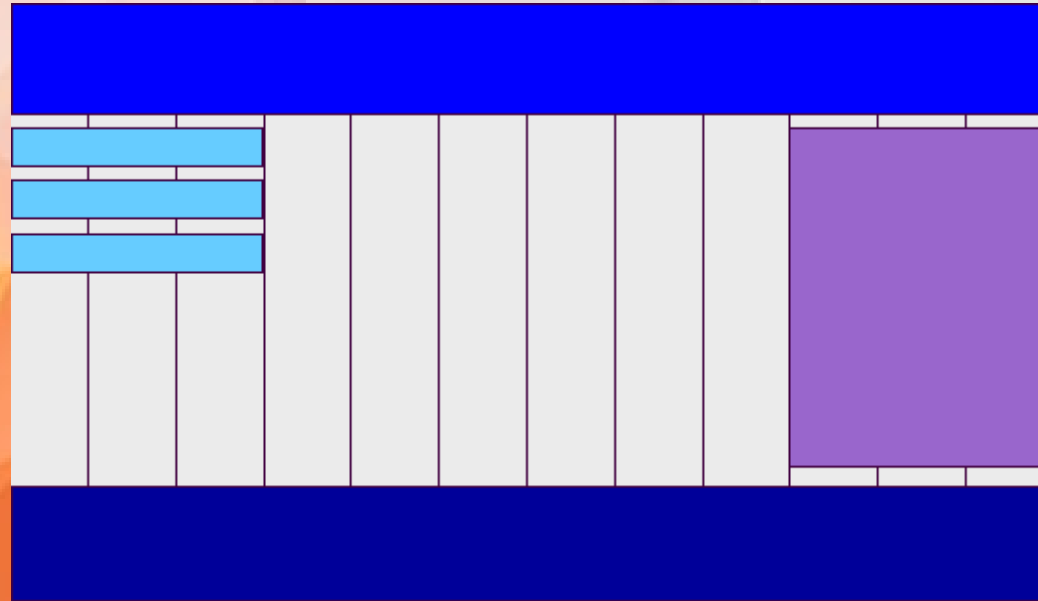
- `width` Anchura del viewport.
- `height` Altura del viewport.
- `initial-scale` Escala inicial del documento.
- `minimum-scale` Escala mínima configurable del documento.
- `maximum-scale` Escala máxima configurable del documento.
- `user-scalable` Si se permite o no al usuario hacer zoom.

# ACTIVIDAD

- Diseña una web con texto y una imagen y visualízala con el emulador de dispositivos de Chromium (Herramientas para desarrolladores). Prueba varios dispositivos.
- Añade el meta viewport al documento para que fuerze el tamaño de visualización al del dispositivo y deshabilite zoom.
- Modifica la fuente del documento para que su tamaño sea relativo al viewport.

# GRID

- El diseño basado en grid, divide la página en una serie de columnas donde los contenidos se organizan por filas.



# GRID

```
<header>
```

```
<div class="row">
```

```
<div class="col-9">
```

```
<div class="row">
```

```
<div class="col-3">
```

```
<nav>
```

```
<div class="col-9">
```

```
<article>
```

```
<div class="row">
```

```
<div class="col-3">
```

```
<div class="col-9">
```

```
<article>
```

```
<div class="row">
```

```
<div class="col-3">
```

```
<div class="col-9">
```

```
<article>
```

```
<div class="col-3">
```

```
<div class="row">
```

```
<div class="col-12">
```

```
<aside>
```

```
<footer>
```

# GRID

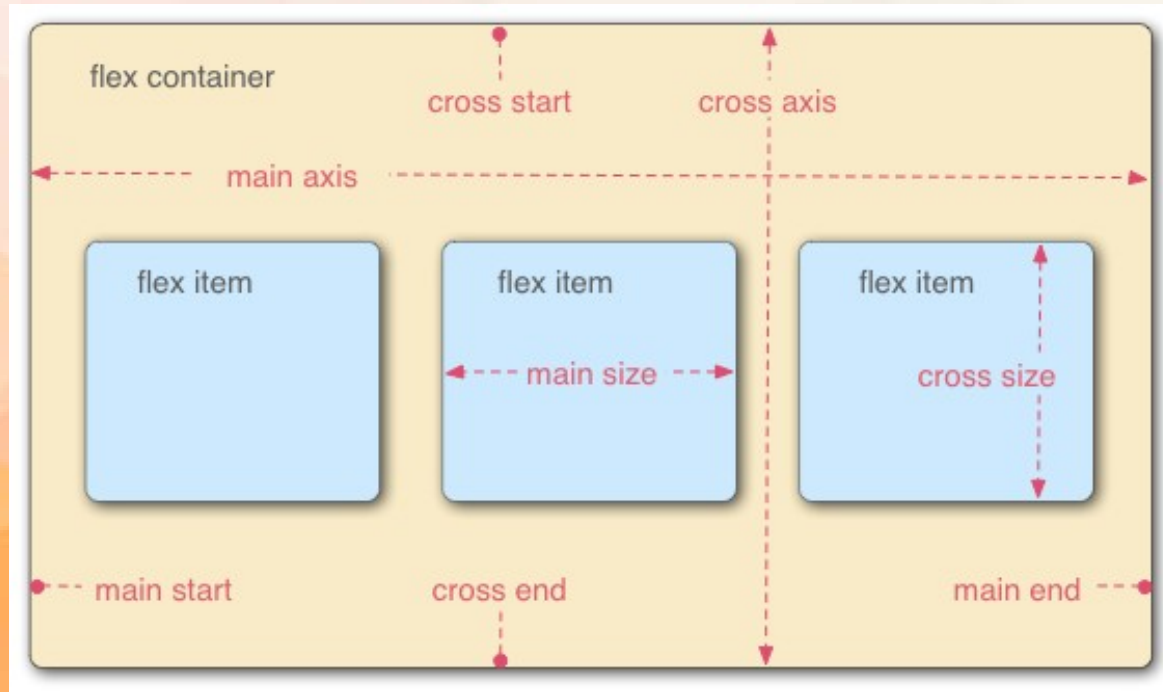
- Todos contenidos se ponen dentro de columnas.
- La página se estructura en filas.
- Este patrón de diseño tiene el inconveniente de la dificultad de cambiar la posición de los bloques cuando se redimensiona la pantalla.



# GRID FLEXIBLE

- Basado en el modelo de caja flexbox (`display: flex;`)
- Permite colocar los elementos de una página para que se comporten de forma predecible cuando el diseño de la página debe acomodarse a diferentes tamaños de pantalla y diferentes dispositivos
- No todos los navegadores implementan esta propiedad.

# GRID FLEXIBLE



```
display: flex;
```

# GRID FLEXIBLE

## → CSS Para el **contenedor** flex:

- `flex-direction` Especifica como se sitúan los elementos dentro del contenedor. Los valores que puede tomar son `row`, `row-reverse`, `column`, `column-reverse`.
- `flex-wrap` Especifica si el contenedor tiene una o varias líneas. Los valores que puede tomar son: `no-wrap`, `wrap`, `wrap-reverse`.
- `flex-flow` Propiedad abreviada de las propiedades anteriores.  
(`flex-flow: row wrap`)

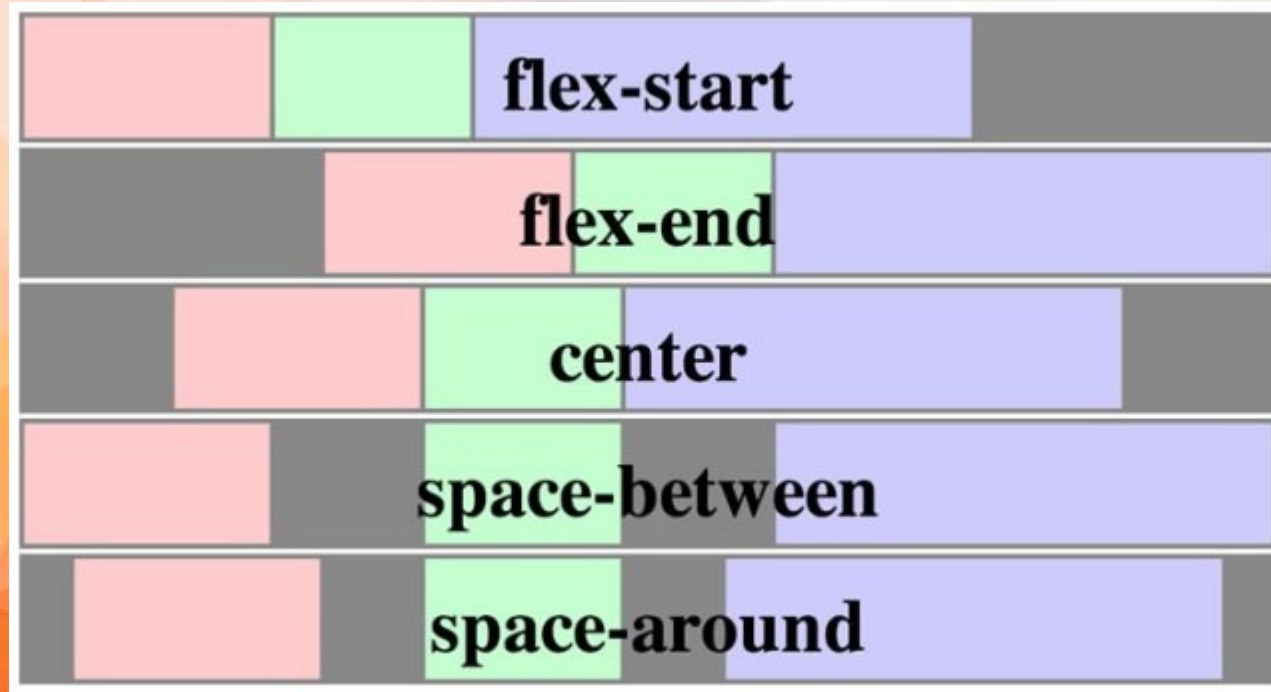
# GRID FLEXIBLE

## → CSS Para el **contenedor** flex:

- `justify-content` Permite alinear los elementos en el eje principal. Toma por valor `flex-start`, `flex-end`, `center`, `space-between`, `space-around`.
- `align-items` Permite alinear todos elementos en el eje transversal. Toma por valor: `auto`, `flex-start`, `flex-end`, `center`, `baseline`, `stretch`.
- `align-self` Permite alinear los elementos individuales en el eje transversal; así pues, sobrescribe la propiedad general.

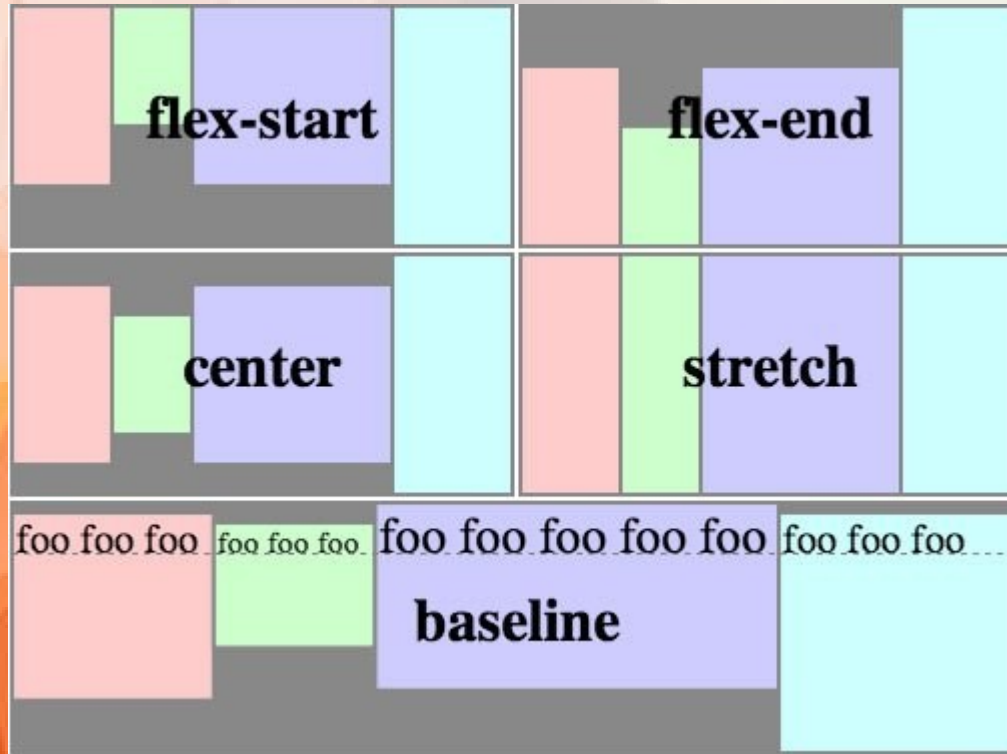
# GRID FLEXIBLE

→ Alineando respecto al eje principal (main)



# GRID FLEXIBLE

→ Alineando respecto al eje transversal (cross)



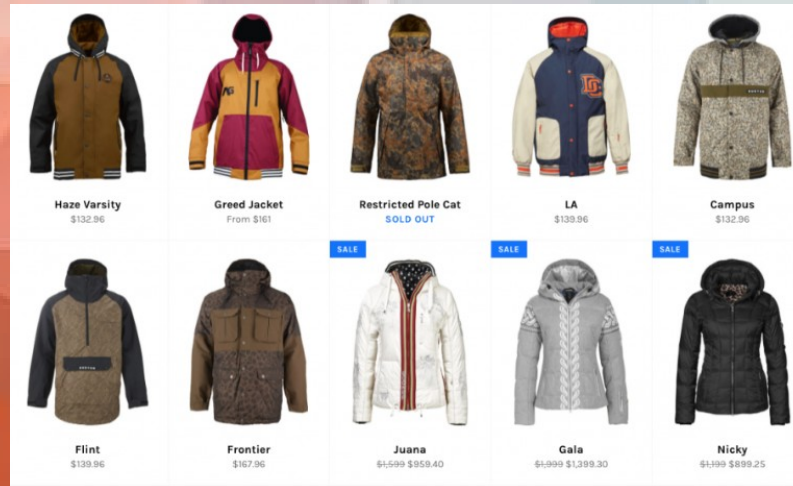
# GRID FLEXIBLE

## → CSS para los **contenidos**:

- `order` Para establecer la orden en que aparecen los elementos de una caja flexible. Por defecto es 0.
- `flex-grow` / `flex-shrink` Especifica el factor de crecimiento/decrecimiento, es decir, cuánto crece/decrece un elemento en relación con los otros cuando hay espacio disponible en el contenedor. Por defecto es 0 (1)
- `flex-basis` Toma el mismo valor que la propiedad `width`. Especifica la medida inicial del elemento antes de distribuir el espacio libre con las propiedades anteriores. Por defecto es `main-size` (auto).
- `flex: flex-grow flex-shrink flex-basis|`

# ACTIVIDAD

- Crea un listado de productos con una caja flex. Los productos se autoajustarán al tamaño, dejando siempre un espacio proporcional entre ellos.

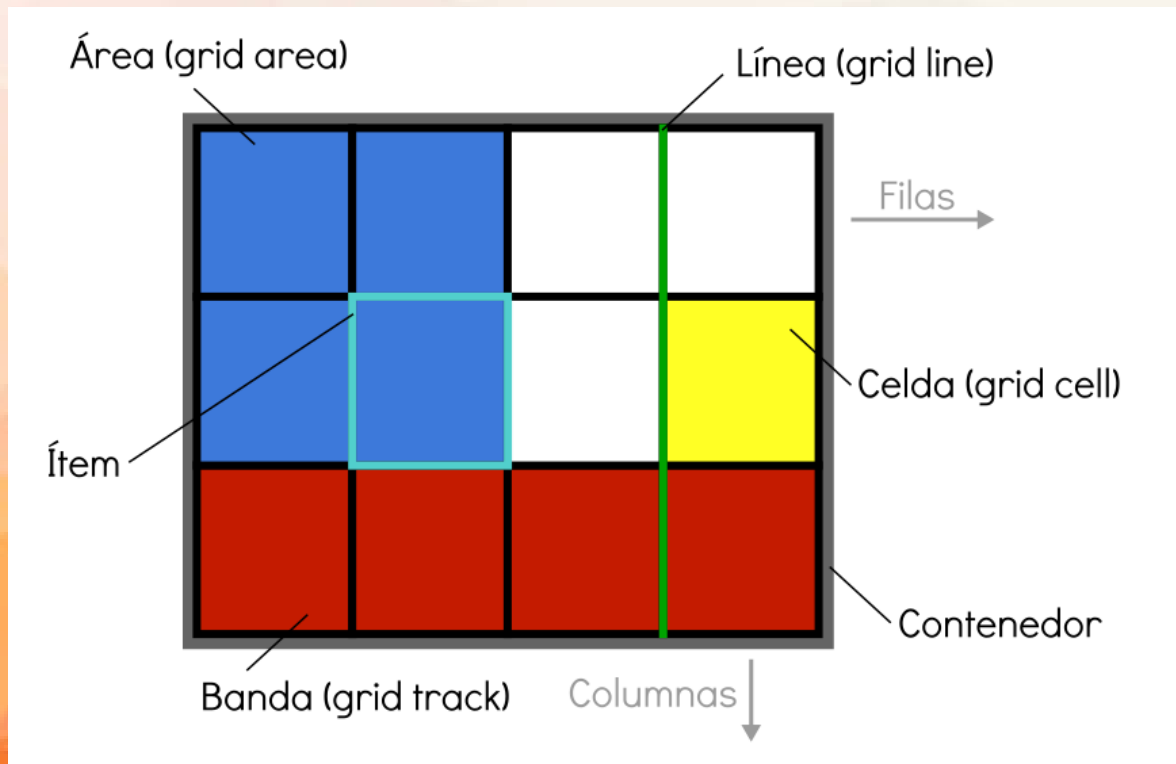




# GRID CSS

- Aunque flexbox es muy versátil, es insuficiente para definir de forma sencilla algunos layouts más avanzados.
- A partir del diseño grid, aparece grid CSS con el objetivo de simplificar las cosas.

# GRID CSS



# CSS GRID

- `Display: inline-grid | grid` Define la visualización en formato grid
- `Grid-template-columns | grid-template-rows` Especifica las columnas/filas. Se pueden definir en px o en fracciones (fr)

<https://codepen.io/cmilianb/pen/LYYNjoG>

# CSS GRID

- es posible indicar el nombre y posición concreta de cada área de una cuadrícula. Para ello utilizaremos la propiedad `grid-template-areas`, donde debemos especificar el orden de las áreas en la cuadrícula
- Posteriormente, en cada ítem hijo, utilizamos la propiedad `grid-area` para indicar el nombre del área del que se trata
- Ponemos `none` si no queremos poner nada en esa celda.
- Con `grid-gap` definimos la separación entre filas y columnas.

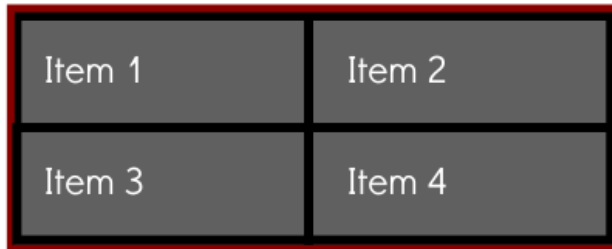
# CSS GRID

→ La distribución de los elementos en el grid se hace de forma similar al flexbox:

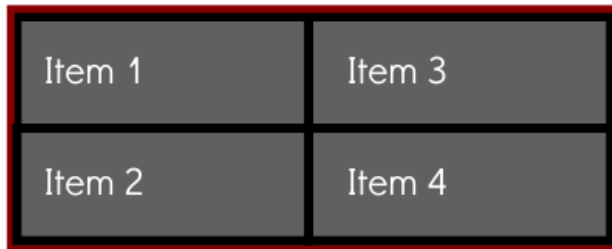
- `Justify-content: start | end | center | stretch | space-around | space-between | space-evenly`
- `Align-content: start | end | center | stretch | space-around | space-between | space-evenly`

# CSS GRID

→ Ajuste de celdas:



grid-auto-flow: row



grid-auto-flow: column



# CSS GRID

## → Propiedades para los elementos hijos:

- `grid-column-start` Indica en que columna empezará el ítem de la cuadrícula.
- `grid-column-end` Indica en que columna terminará el ítem de la cuadrícula.
- `grid-row-start` Indica en que fila empezará el ítem de la cuadrícula.
- `grid-row-end` Indica en que fila terminará el ítem de la cuadrícula.

The background is an abstract geometric pattern composed of numerous triangles of varying sizes. The colors transition from a warm orange on the left to a cool blue on the right, with a pale yellow/white area at the top. The text is centered in the middle of the image.

# **PRÁCTICA 2.3**