Unidades

Hojas de Estilo en Cascada

Unidad Absoluta

Una unidad cuyo valor no depende de otra cosa, por lo que su tamaño será constante sin importar el contexto. En general, la unidad px es la única de este tipo que se usa en la web.

Unidad Relativa

Una unidad cuyo valor depende de otra cosa. Estas son las unidades relativas más usadas:

em: Relativa al tamaño de la fuente. Por ejemplo, si el tamaño de la fuente es 14px, entonces 1.5em será 21px. Si la unidad em se usa para definir el tamaño de fuente, será relativa al tamaño de la fuente del elemento padre.

rem: Relativa al tamaño de la fuente del elemento raíz. Por defecto, suele ser 16px, pero puede ser modificada por el usuario en la hoja de estilos. Además, el autor de la hoja de estilos puede cambiar esto definiendo un tamaño de fuente en el selector html o en la pseudo-clase :root. Por ejemplo, por defecto 1.5rem será 24px.

Unidad Relativa

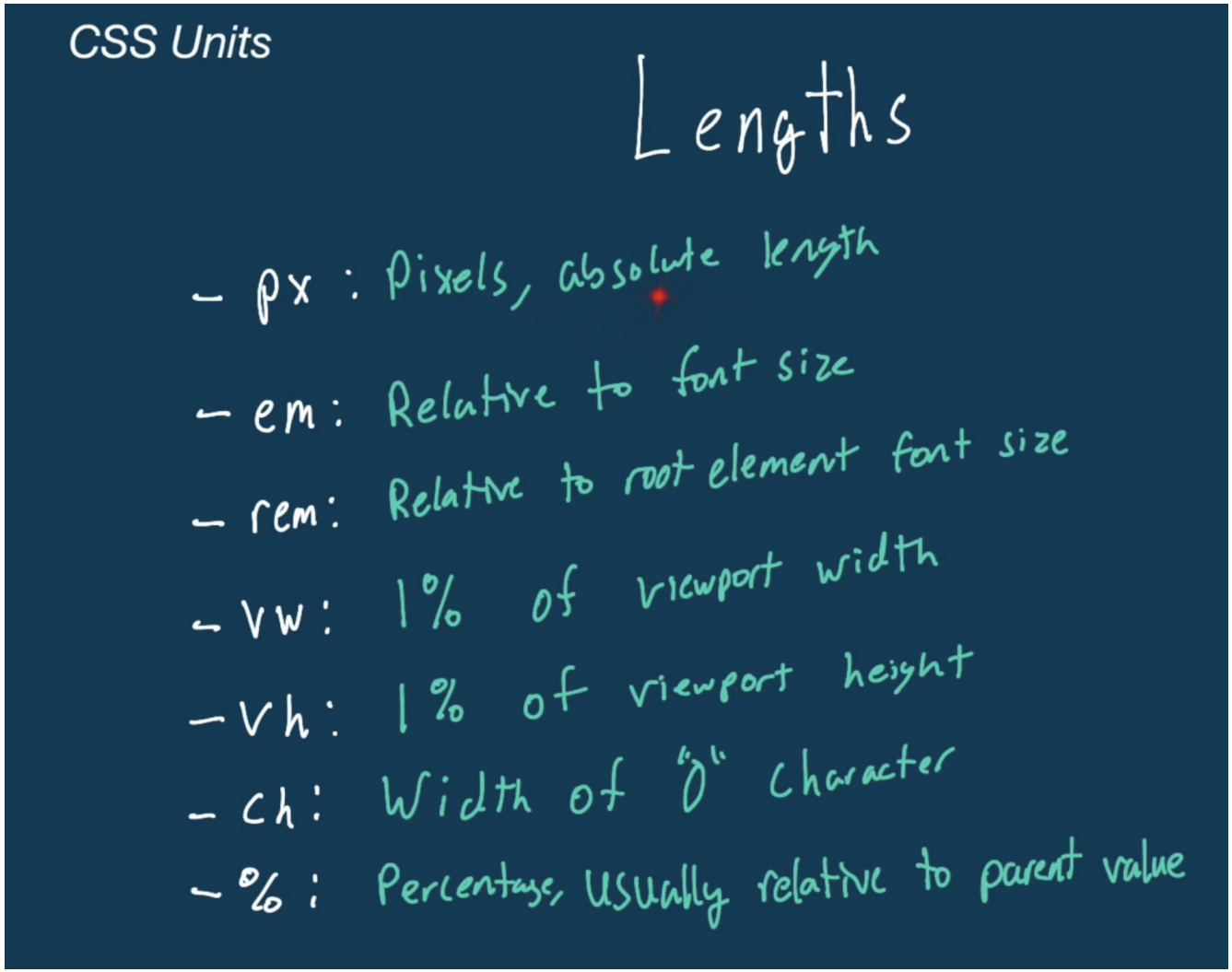
%: Un porcentaje, normalmente relativo al valor del elemento padre para la misma propiedad. Por ejemplo, un ancho de 50% será la mitad del ancho del elemento padre.

vw: Un porcentaje del ancho de la ventana gráfica (viewport). Por ejemplo, 50vw será la mitad del ancho del viewport.

vh: Un porcentaje de la altura de la ventana gráfica (viewport). Por ejemplo, 50vh será la mitad de la altura del viewport.

ch: El número de caracteres en una línea, basado en el tamaño del carácter "0" en la fuente del elemento. Esto puede ser útil para evitar que los párrafos tengan más de ~70 caracteres de ancho, lo que puede dificultar la lectura.

Unidad Relativa



unidades CSS para ancho y alto:

% → relativo al elemento padre

vw / vh → relativo al viewport (ventana del navegador)

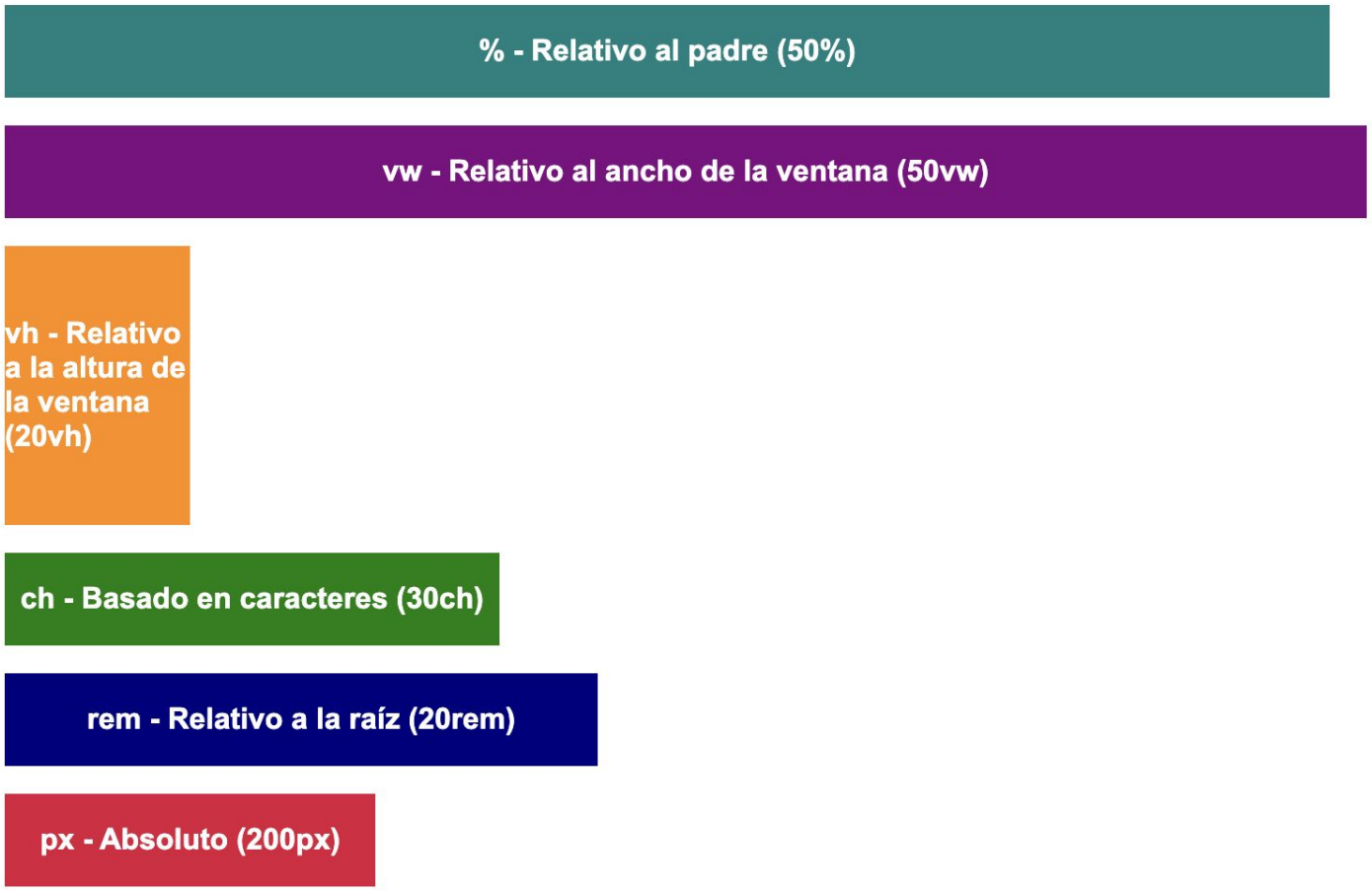
ch → útil para definir anchos de párrafos (basado en el ancho del carácter “0”)

rem → más cercano a valores absolutos (relativo al tamaño de fuente raíz)

px → como último recurso para valores verdaderamente absolutos

unidades CSS para ancho y alto:

Ejemplos de Unidades CSS



Margin / Padding:

rem → más cercano a valores absolutos (relativo al tamaño de fuente raíz).

em → escala según el tamaño de fuente del elemento.

px → útil para valores pequeños, como último recurso.

Margin / Padding:

Ejemplos de Margin y Padding con distintas unidades

Caja con margin y padding en REM

Caja con margin y padding en EM

Caja con margin y padding en PX

Margin / Padding:

La caja con rem siempre depende del tamaño de fuente raíz (html), aunque cambie el tamaño de fuente interno.

La caja con em depende del tamaño de fuente del propio elemento; si cambian font-size, todo el margen y padding cambia proporcionalmente.

La caja con px siempre es fija, sin importar el contexto.

Borders, Shadows:

px → para valores pequeños.

rem / em → también funcionan aquí, aunque al escalar no siempre se ven bien.

Borders, Shadows:

Ejemplos de Bordes y Sombras con distintas unidades

Borders, Shadows:

El borde y sombra en px siempre es fijo y no cambia aunque modifiques el font-size.

El borde y sombra en rem dependen del tamaño de fuente raíz (html).

El borde y sombra en em cambian si ajustas el font-size del propio elemento.

Font Size:

rem → generalmente es la mejor opción (depende del tamaño de fuente raíz).

em → se escala en relación al tamaño de fuente del elemento padre.

px → como último recurso, valor absoluto y fijo.

Font Size:

Ejemplos de Font Size con distintas unidades

Texto con font-size en REM (2rem = 32px)

Texto con font-size en EM (2em, relativo al padre = 40px)

Texto con font-size en PX (24px, valor fijo)

Font Size:

El texto con rem siempre es consistente, ya que depende del tamaño raíz (html).

El texto con em cambia dependiendo del tamaño de fuente del padre.

El texto con px siempre será fijo sin importar el contexto.

Units – Colores:

Palabras clave (keywords): red, blue, green, etc.

Hex RGB: #4B7DAF

Cada par representa rojo, verde y azul.

RGB: rgb(75, 125, 175)

RGBA (con transparencia): rgba(75, 125, 175, 0.5)

HSL (matiz, saturación, luminosidad): hsl(210, 40%, 49%)

HSLA (con transparencia): hsla(210, 40%, 49%, 0.5)

Units – Colores:

Keywords (Palabras clave):

- Usan nombres predefinidos de colores: red, blue, green, black, white, etc.

- Son fáciles de recordar, pero limitados en opciones.

Units – Colores:

Hexadecimal (Hex RGB):

- Formato: #RRGGBB → cada par representa rojo, verde y azul en valores de 00 a FF (0 a 255 en decimal).

- Ejemplo: #4B7DAF → Rojo=4B, Verde=7D, Azul=AF.

- Es el formato más usado en diseño web.

Units – Colores:

RGB y RGBA:

- RGB: define el color con valores entre 0 y 255 para Rojo, Verde y Azul.

  Ejemplo: rgb(75,125,175)

- RGBA: añade un canal Alpha para controlar la transparencia (0=invisible, 1=opaco).

  Ejemplo: rgba(75,125,175,0.5)

Units – Colores:

HSL y HSLA:

- HSL: define el color con tres valores:

- Hue (matiz): ángulo en la rueda de color (0–360°).

- Saturation (saturación): intensidad del color (0% gris → 100% color puro).

- Lightness (luminosidad): claridad del color (0% negro → 100% blanco).

  Ejemplo: hsl(210, 40%, 49%)

Units – Colores:

HSLA: igual que HSL pero con canal Alpha (transparencia).

Ejemplo: hsla(210, 40%, 49%, 0.5)

Units – Colores:

Ejemplos de Colores en CSS

Keyword: red

Hex: #4B7DAF

RGB: rgb(75,125,175)

RGBA: rgba(75,125,175,0.5)

HSL: hsl(210,40%,49%)

HSLA: hsla(210,40%,49%,0.5)

Units – Colores:

- Con keywords se usan nombres de colores ya definidos.
- El hexadecimal es la forma más común y precisa.
- RGB controla rojo, verde y azul con valores de 0 a 255.
- RGBA añade un cuarto valor de transparencia (0 = transparente, 1 = opaco).
- HSL es más intuitivo:
- hue (matiz, en grados 0–360),
- saturation (0% gris → 100% color puro),
- lightness (0% negro → 100% blanco).
- HSLA también incluye transparencia.

Units – Colores:

- Usa keywords para pruebas rápidas.
- Usa Hex para colores comunes en diseño.
- Usa RGB/RGBA cuando necesites control exacto.
- Usa HSL/HSLA para ajustar colores de manera más intuitiva.

Units – Colores:

Tener cuidado con cómo se usa y combina para evitar resultados no deseables.

I am a very long paragraph. I am a very long paragraph. I am a very long paragraph. I am a very long paragraph. I am a very long paragraph. I am a very long paragraph. I am a very long paragraph. I am a very long paragraph.

Actividad:

Crear una página web sencilla con una tarjeta que muestre tu nombre, una breve descripción y un botón o enlace.

Archivos requeridos:

index.html

styles.css

Crea el archivo index.html con la siguiente estructura básica:

Actividad:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Mi Tarjeta CSS</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <section class="card">
    <img src="foto.jpg" alt="Foto de perfil">
    <h1>Tu nombre</h1>
    <p>Una breve descripción sobre ti o una frase favorita.</p>
    <button>Contáctame</button>
  </section>
</body>
</html>
```

Actividad:

Crea el archivo styles.css y aplica los estilos necesarios para dar formato a tu tarjeta.

Deberás usar diferentes unidades CSS y formatos de color.

Asegúrate de incluir comentarios en tu código explicando cada sección.

Actividad:

Tu tarjeta debe incluir:

- Selectores básicos: de tipo (h1, p), de clase (.card), y al menos una pseudo-clase o pseudo-elemento (:hover, ::before, ::after).
- Unidades CSS: usa al menos cuatro diferentes (%, px, rem, em, ch, vw, vh).
- Colores: usa al menos tres formatos distintos (keyword, hex, rgb/rgba, hsl/hsla).
- Propiedades: aplica márgenes, paddings, bordes, sombras y tamaños de fuente.
- Comentarios en el CSS explicando qué hace cada bloque de código.

Actividad:

Tu tarjeta debe incluir:

- Experimenta con diferentes colores usando rgba() o hsl().
- Usa :hover para cambiar el color del botón al pasar el cursor.
- Aplica box-shadow o border-radius para mejorar la presentación.
- Cambia valores de em y rem para entender su diferencia.