

Estilos

Tailwind CSS v4 y daisyUI 5

Computación en Internet 3 • Semana 13 • Universidad Icesi



Tailwind CSS v4



daisyUI 5



Ejemplo: login

Qué vamos a ver



Tailwind CSS v4

De las utilidades básicas a la personalización con @theme



daisyUI 5

Componentes semánticos, temas y colores sobre Tailwind



Ejemplo integrado

Una pantalla de login construida con ambos

Proyecto base: rama [week-12/next-redux](#) con Next.js 16, Tailwind v4 y daisyUI 5 ya configurados.

Tailwind CSS v4

De cero a una interfaz completa

Del CSS clásico a utility-first

CSS clásico

Guardar

styles.css

```
.boton-guardar {  
  padding: 1rem;  
  border-radius: 0.75rem;  
  background: #2563eb;  
  color: white;  
}
```

JSX / HTML

```
<button class="boton-guardar">  
  Guardar  
</button>
```

Con CSS clásico inventas un nombre y lo defines en otro archivo.

Del CSS clásico a utility-first

CSS clásico

Guardar

styles.css

```
.boton-guardar {  
  padding: 1rem;  
  border-radius: 0.75rem;  
  background: #2563eb;  
  color: white;  
}
```

JSX / HTML

```
<button class="boton-guardar">  
  Guardar  
</button>
```

Utility-first

Guardar

JSX / HTML

```
<button class="p-4 rounded-xl bg-blue-600  
  text-white font-semibold">  
  Guardar  
</button>
```

Sin archivo CSS: el estilo vive en las clases.

Con utility-first las clases describen el estilo; no hay archivo que mantener.

Anatomía de una clase

p-4



```
padding: 1rem; /* 16 px */
```

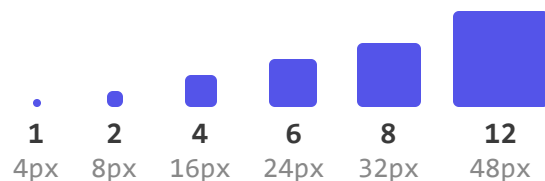
p propiedad

p = padding, m = margin, w = width, bg = background...

4 valor de la escala

$4 \times 0.25 \text{ rem} = 1 \text{ rem} = 16 \text{ px}$

LA ESCALA (cada paso = 4 px)



mt-2 margin-top: 0.5rem

rounded-lg border-radius:
0.5rem

text-sm font-size: 0.875rem

bg-blue-600 azul 600

Cada clase aplica una declaración CSS con un valor de la escala del tema.

Instalar Tailwind v4 en Next.js

Terminal

```
npm install tailwindcss @tailwindcss/postcss postcss
```

postcss.config.mjs

```
const config = {  
  plugins: {  
    "@tailwindcss/postcss": {},  
  },  
};  
  
export default config;
```

app/globals.css

```
@import "tailwindcss";  
  
@plugin "daisyui" {  
  themes: light --default;  
}
```

Dos archivos y listo: sin tailwind.config.js y sin lista content, Tailwind detecta tus archivos.

Qué cambió de v3 a v4

EN V3	EN V4
<code>@tailwind base; @tailwind utilities;</code>	<code>@import "tailwindcss";</code>
<code>tailwind.config.js</code> con <code>content</code>	<code>@theme</code> en CSS y detección automática
<code>bg-gradient-to-r</code>	<code>bg-linear-to-r</code>
<code>shadow-sm</code> · <code>shadow</code>	<code>shadow-xs</code> · <code>shadow-sm</code>
<code>rounded-sm</code> · <code>rounded</code>	<code>rounded-xs</code> · <code>rounded-sm</code>
<code>ring</code> (3 px)	<code>ring-3</code> (<code>ring</code> ahora es 1 px)
<code>outline-none</code>	<code>outline-hidden</code>
<code>!flex</code> (importante al inicio)	<code>flex!</code> (al final)

Para migrar un proyecto: `npx @tailwindcss/upgrade` (Node 20 o superior).

Espaciado y dimensiones

m-6



p-6 relleno interno: 1.5 rem

px-4 py-2 eje X y eje Y por separado

m-6 espacio exterior: 1.5 rem

mx-auto centra un bloque en horizontal

Escala: 1 =4 px · 2 =8 px · 4 =16 px · 6 =24 px · 8 =32 px

p es el relleno interno; m el espacio exterior. Ambos usan la misma escala.

Espaciado y dimensiones

gap-2



gap-4



gap-8



flex gap-4 separa a los hijos de un flex

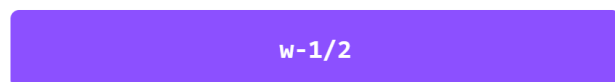
grid gap-6 igual en un grid

space-y-2 separación vertical entre hijos

Prefiere **gap** en el contenedor: así no hay que poner margen en cada hijo.

gap lo decide el contenedor: separa filas y columnas sin tocar a los hijos.

Espaciado y dimensiones



w-24 ancho fijo de la escala (6 rem)

w-1/2 la mitad del contenedor

w-full 100 % del contenedor

size-12 ancho y alto a la vez

max-w-md ancho máximo (28 rem)

min-h-screen alto mínimo: toda la pantalla

El tamaño sale de la escala, de fracciones o de porcentajes; **size-*** fija ancho y alto.

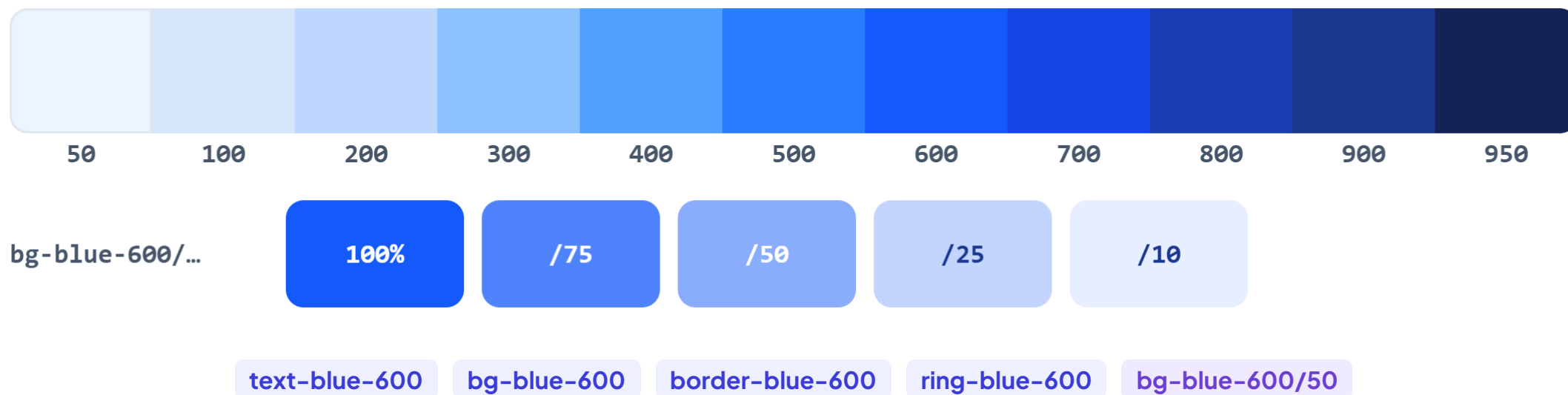
Tipografía

<code>text-sm</code>	Texto pequeño
<code>text-xl</code>	Texto grande
<code>text-3xl font-bold</code>	Título
<code>font-light</code>	Peso ligero
<code>tracking-widest</code>	ESPACIADO
<code>truncate w-40</code>	Un texto demasia...

<code>text-xs ... 9xl</code>	tamaño de fuente
<code>font-light · font-bold</code>	peso
<code>leading-tight</code>	interlineado
<code>tracking-wide</code>	espacio entre letras
<code>text-center</code>	alineación
<code>uppercase</code>	transformación
<code>truncate</code>	corta con puntos suspensivos

Tamaño, peso y espaciado de texto son clases independientes que se combinan.

Color



Un color es familia + intensidad (50 a 950). La barra `/50` agrega opacidad.

Color: gradientes



`bg-linear-to-r`
`from-blue-500 to-purple-500`



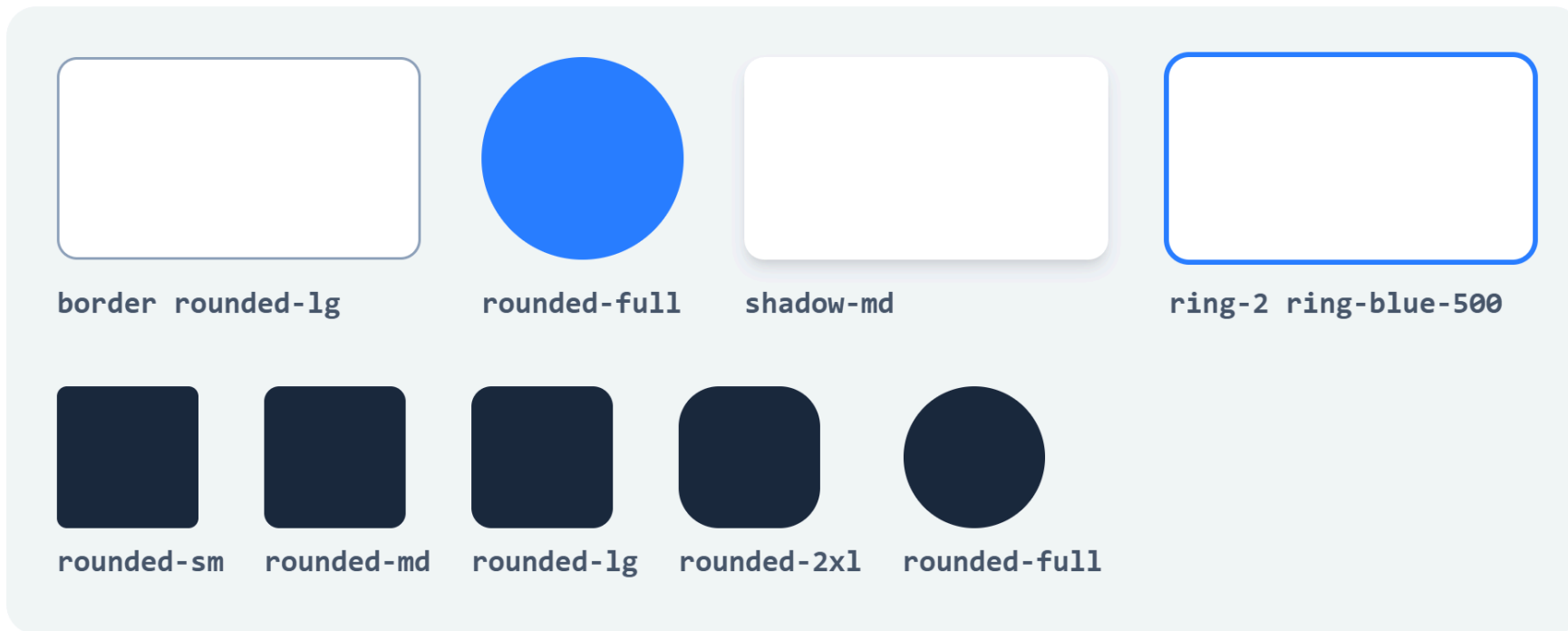
`bg-linear-to-b`
`from-red-500 to-amber-400`



`bg-linear-to-br from-emerald-400`
`via-sky-500 to-indigo-600`

En v4 el prefijo es `bg-linear-to-*` (antes `bg-gradient-to-*`). Usa `from-`, `via-` y `to-`.

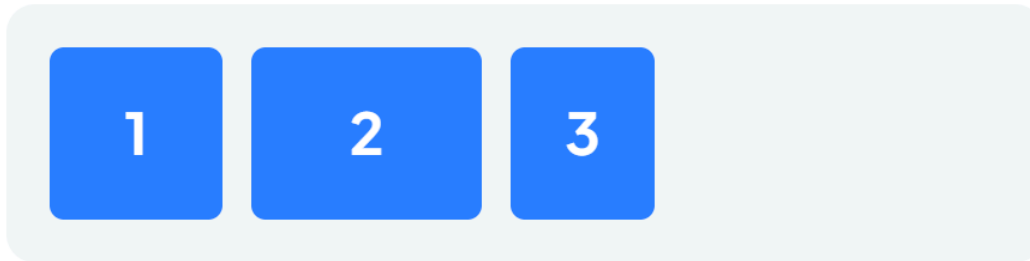
Bordes, radios y sombras



Radios y sombras tienen escala: `rounded-sm` a `rounded-full` ; `shadow-xs` a `shadow-2xl` .

Flexbox

`flex`



`flex flex-col`



`flex` pone a los hijos en fila; `flex-col` los apila.

Flexbox

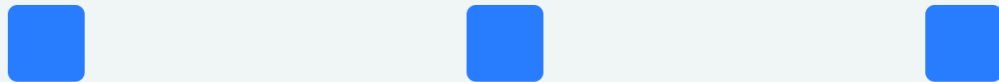
justify-start



justify-center



justify-between



justify-end



items-start



items-center



items-end



justify-* reparte en el eje principal; **items-*** alinea en el eje cruzado.

Flexbox

GymTrack

Rutinas

Historial

Perfil

Entrar

flex

items-center

justify-between

px-5 py-3

gap-5

Un navbar típico es `flex items-center justify-between`.

Grid

`grid grid-cols-3 gap-4`



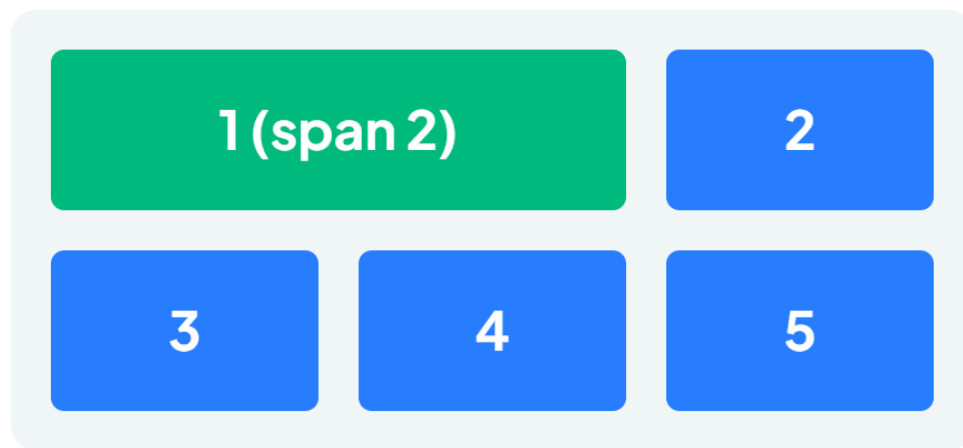
`grid grid-cols-2 gap-2`



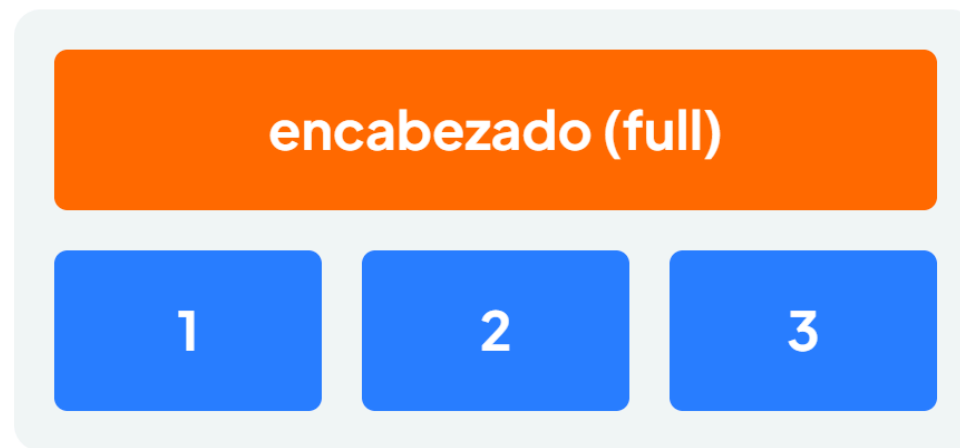
`grid-cols-N` crea N columnas iguales; `gap-*` separa filas y columnas.

Grid

col-span-2



col-span-full



col-span-2

col-span-full

row-span-2

grid-cols-1 md:grid-cols-3

col-span-* hace que un hijo ocupe varias columnas.

Posicionamiento



relative el padre es la referencia

absolute sale del flujo normal

top-3 right-3 distancia a los bordes

inset-x-0 izquierda y derecha en 0

-top-3 valor negativo: hacia afuera

z-10 orden de apilado

sticky top-0 se pega al hacer scroll

absolute se ubica respecto al ancestro **relative** más cercano.

Transiciones y animación



transition anima los cambios de estilo

duration-200 duración: 200 ms

ease-in-out curva de aceleración

hover:scale-105 crece un 5 % al pasar el mouse

animate-spin gira sin parar (también pulse, bounce)

motion-reduce:animate-none respeta la preferencia del usuario

Una transición necesita dos cosas: **transition** y el estado que cambia (**hover:**).

Estados: hover, focus, active, disabled



base



hover:bg-blue-700



focus:ring-2



active:scale-95



disabled:opacity-50

JSX / HTML

```
<button class="bg-blue-600 hover:bg-blue-700  
  focus:ring-2 active:scale-95 disabled:opacity-50">  
  Guardar  
</button>
```

Un prefijo y dos puntos aplican la clase solo en ese estado.

Estados de otros elementos

group: reacciona al padre

```
<a class="group flex gap-2">  
  Ver rutina  
  <span class="group-hover:translate-x-1">  
    →  
  </span>  
</a>
```

has-*: reacciona a un hijo

```
<label class="has-checked:bg-blue-50">  
  <input type="checkbox" /> Fuerza  
</label>
```

peer: reacciona al hermano anterior

```
<input type="checkbox" class="peer" />  
<p class="hidden peer-checked:block">  
  Aceptaste los términos  
</p>
```

Otros útiles: first: last: odd: even: placeholder: focus-visible:

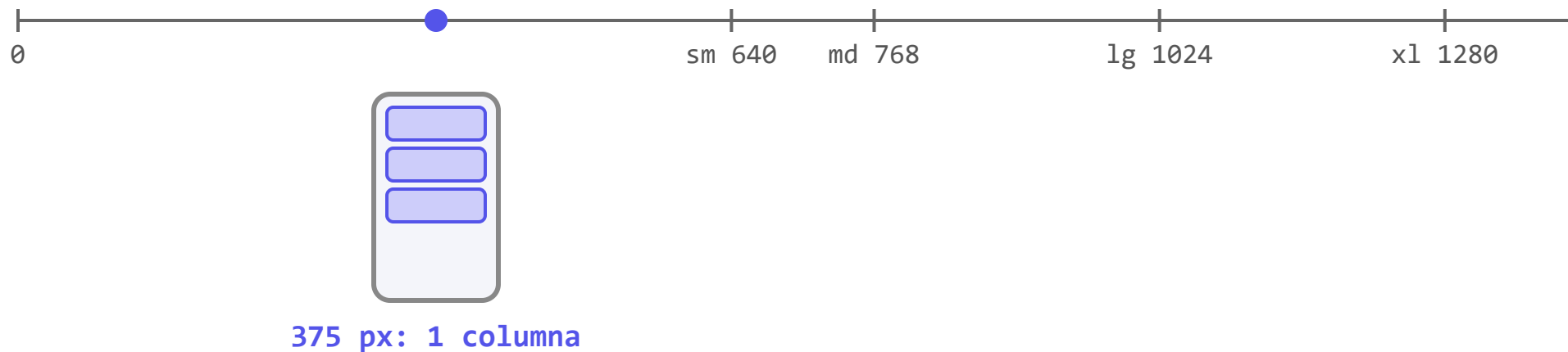
group, **peer** y **has-***: un elemento reacciona al estado de otro.

Mobile-first con breakpoints

grid-cols-1

gana

ancho de pantalla



Sin prefijo aplica desde 0 px: una columna.

Mobile-first con breakpoints

`grid-cols-1`

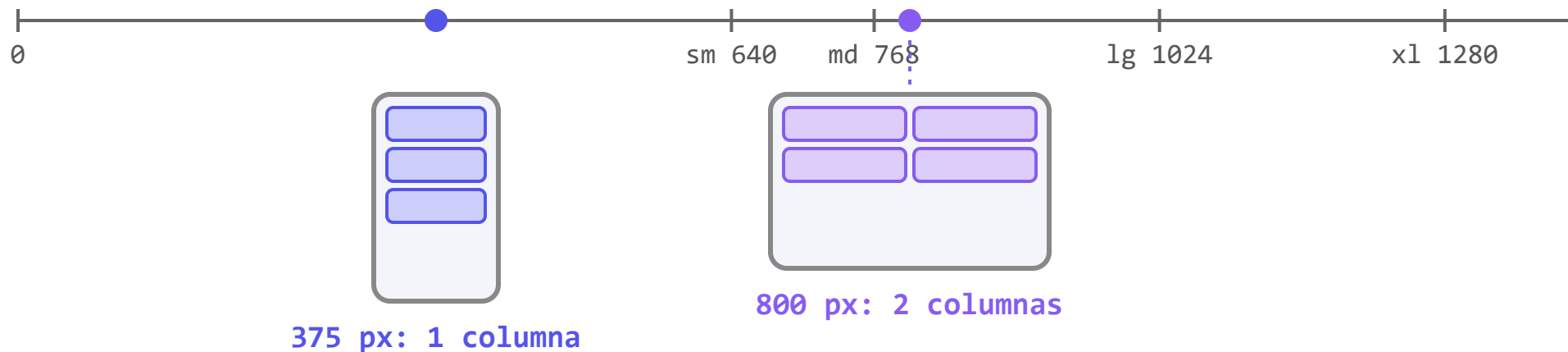
gana

sobrescrita

`md:grid-cols-2`

gana

ancho de pantalla



`md:` desde 768 px sobrescribe: dos columnas.

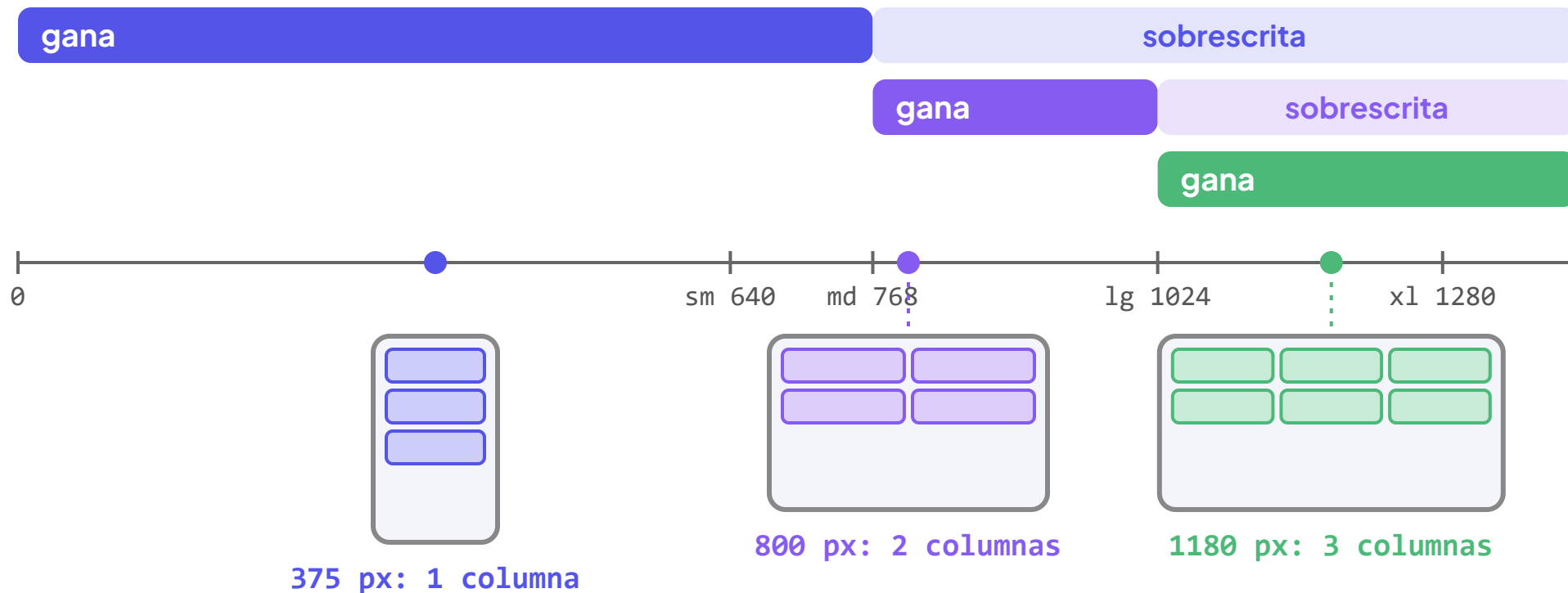
Mobile-first con breakpoints

`grid-cols-1`

`md:grid-cols-2`

`lg:grid-cols-3`

ancho de pantalla



lg: desde 1024 px sobrescribe de nuevo: tres columnas.

Breakpoints y rangos

PREFIJO	DESDE	MEDIA QUERY
sm	640 px (40 rem)	<code>width >= 40rem</code>
md	768 px (48 rem)	<code>width >= 48rem</code>
lg	1024 px (64 rem)	<code>width >= 64rem</code>
x1	1280 px (80 rem)	<code>width >= 80rem</code>
2x1	1536 px (96 rem)	<code>width >= 96rem</code>

`max-md:hidden` solo por debajo de 768 px

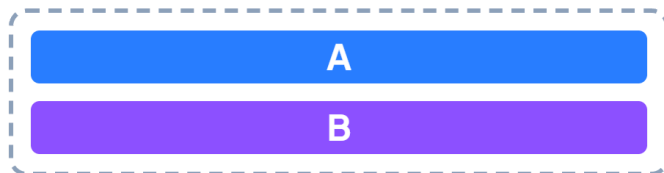
`md:max-lg:flex` solo entre 768 y 1023 px

`min-[600px]:p-8` breakpoint a medida

Sin prefijo es la base móvil; `md:` aplica hacia arriba y `max-md:` hacia abajo.

Container queries

contenedor de 300 px



contenedor de 500 px



El contenedor decide

```
<div class="@container">  
  <div class="flex flex-col  
    @md:flex-row">  
    <div>A</div>  
    <div>B</div>  
  </div>  
</div>
```

@md: mira el ancho del contenedor (28 rem), no el de la pantalla.

Dark mode

Rutina de torso

45 min · Fuerza

modo claro

Rutina de torso

45 min · Fuerza

dark:bg-slate-900 dark:text-white

Clases

```
<div class="bg-white text-slate-900  
  dark:bg-slate-900 dark:text-white">
```

globals.css

```
@import "tailwindcss";  
@custom-variant dark  
  (&:where([data-theme=dark], [data-theme=dark] *));
```

Por defecto `dark:` sigue al sistema; con `@custom-variant` lo controlas tú.

Valores arbitrarios

CLASE	CSS GENERADO	CUÁNDO
<code>w-[320px]</code>	<code>width: 320px</code>	medida única
<code>bg-[#bada55]</code>	<code>background-color: #bada55</code>	color puntual
<code>grid-cols-[1fr_2fr]</code>	<code>grid-template-columns: 1fr 2fr</code>	grid a medida (el guion bajo es un espacio)
<code>top-[117px]</code>	<code>top: 117px</code>	posición exacta
<code>bg(--brand)</code>	<code>background: var(--brand)</code>	variable CSS
<code>[mask-type:luminance]</code>	<code>mask-type: luminance</code>	cualquier propiedad CSS

Los corchetes dan valores a medida; si se repiten, llévalos a `@theme`.

Personalizar con @theme

app/globals.css

```
@theme {  
  --color-brand: #2563eb;  
  --font-display: "Satoshi", sans-serif;  
  --breakpoint-3xl: 120rem;  
}
```

bg-brand

font-display

3xl:grid-cols-6

--color-brand crea bg-brand, text-brand, border-brand

--font-display crea font-display

--breakpoint-3xl crea la variante 3xl:

Cada token es también una variable CSS: `var(--color-brand)` .

En v4 la configuración vive en CSS: cada token de `@theme` crea sus utilidades.

Utilidades y componentes propios

Utilidad nueva (admite hover:, md:...)

```
@utility content-auto {  
  content-visibility: auto;  
}
```

Clase de componente

```
@layer components {  
  .btn-brand {  
    @apply px-4 py-2 rounded-lg  
      bg-brand text-white;  
  }  
}
```



@utility

Crea una utilidad que funciona con todas las variantes.



@layer components

Clases de componente que las utilidades aún pueden sobrescribir.

Primero componentes de React; CSS propio solo cuando haga falta.

daisyUI 5

Componentes semánticos sobre Tailwind

Un nombre en vez de muchas clases

Solo Tailwind

```
inline-flex items-center justify-center  
px-4 py-2 text-sm font-medium rounded-lg  
text-white bg-blue-600 hover:bg-blue-700  
focus:ring-2
```

Guardar

Un botón con buen aspecto y estados necesita muchas utilidades.

Un nombre en vez de muchas clases

Solo Tailwind

```
inline-flex items-center justify-center  
px-4 py-2 text-sm font-medium rounded-lg  
text-white bg-blue-600 hover:bg-blue-700  
focus:ring-2
```

Guardar

Con daisyUI

```
btn btn-primary
```

Guardar

daisyUI pone el nombre; Tailwind ajusta el layout: `btn btn-primary w-full md:w-auto`.

Instalar daisyUI 5

Terminal

```
npm install -D daisyui@latest
```

app/globals.css

```
@import "tailwindcss";

@plugin "daisyui" {
  themes: light --default, dark --prefersdark;
}
```

themes qué temas incluir (all, false)

include / exclude qué componentes cargar

prefix ej. d- → d-btn

root selector de las variables

logs mensajes en consola

daisyUI es CSS puro: tus elementos siguen siendo `<button>`, `<input>` y `<dialog>`.

Temas con data-theme

light

primary

accent

Acción

dark

primary

accent

Acción

cupcake

primary

accent

Acción

synthwave

primary

accent

Acción

Toda la aplicación

```
<html data-theme="dark">
```

Solo esta sección

```
<div data-theme="cupcake">...</div>
```

`data-theme` cambia toda la paleta; también funciona en un solo contenedor.

Tema propio

```
app/globals.css
@plugin "daisyui/theme" {
  name: "icesi";
  default: true;
  color-scheme: light;
  --color-primary: #5454e9;
  --color-secondary: #865cf0;
  --color-accent: #4cb979;
  --radius-box: 1rem;
}
```

Tema "icesi"

Colores y radios propios, mismos componentes.

primary

secondary

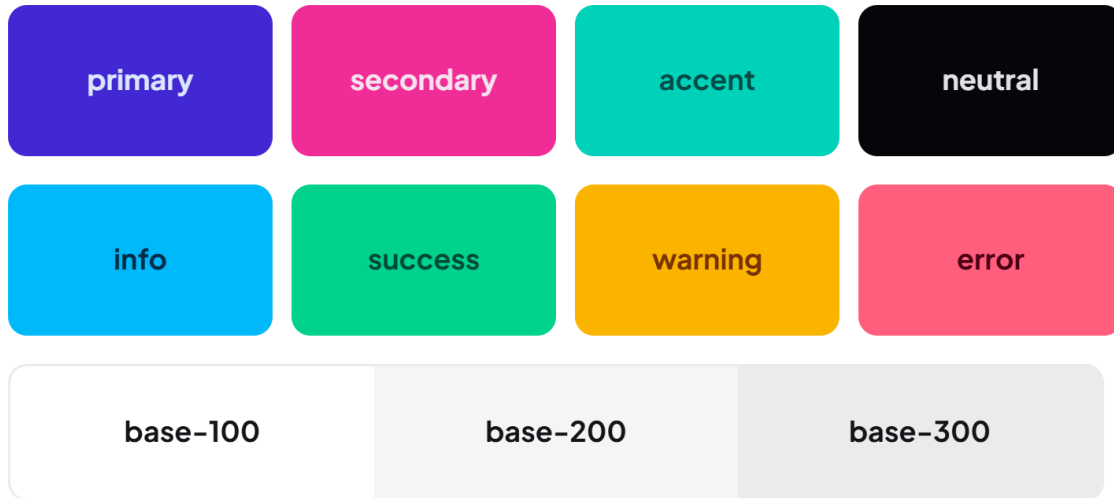
accent

Guardar

Cancelar

Un tema es un bloque de variables CSS: `--color-*`, `--radius-*`, `--size-*`, `--border`.

Colores semánticos



`bg-primary` color principal

`text-primary-content` texto sobre primary

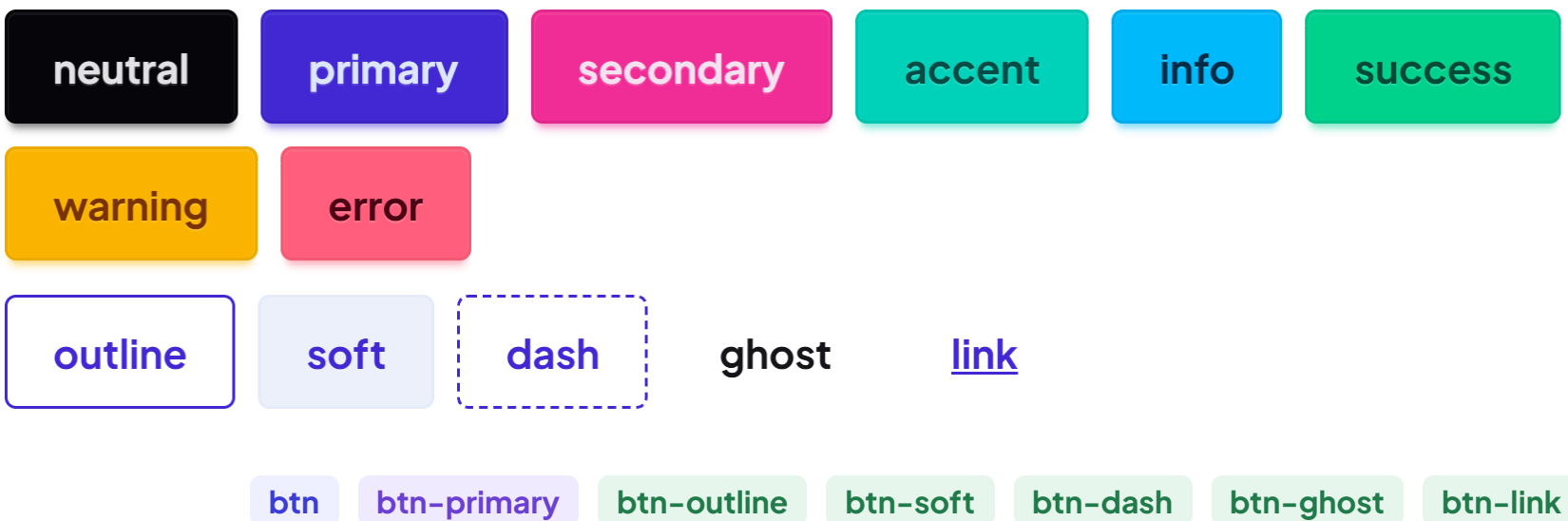
`bg-base-100` superficie base

`text-base-content` texto sobre la base

`bg-primary/50` con opacidad

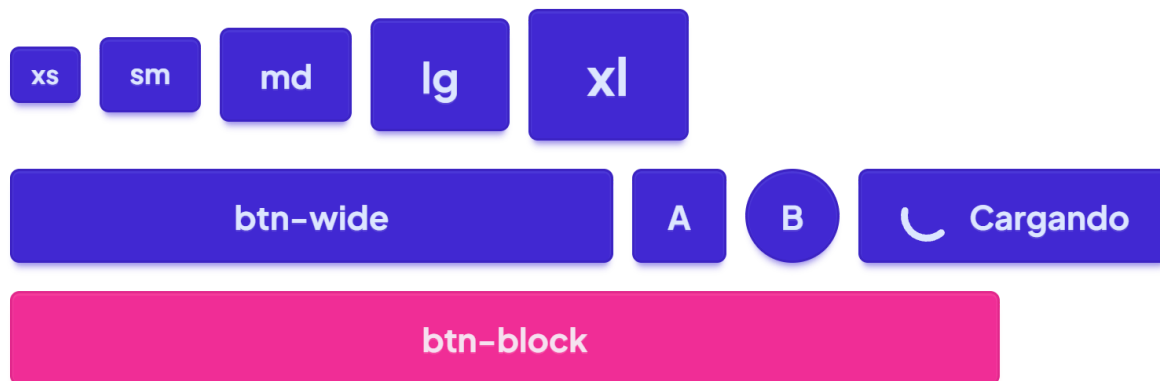
Usa nombres con significado (`primary` , `error`): así respetan el tema activo.

Botón



Se combina la base **btn** con un color y un estilo.

Botón: tamaños y forma



btn-xs ... btn-xl cinco tamaños

btn-wide ancho extra

btn-block ocupa todo el ancho

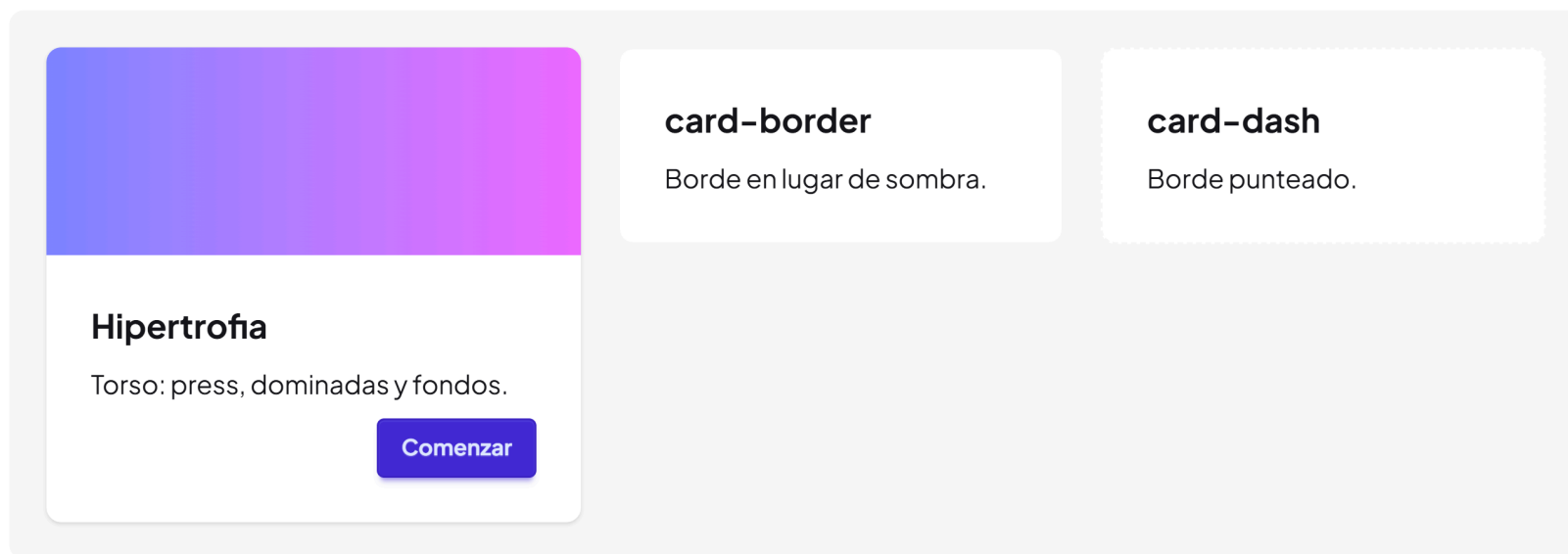
btn-square cuadrado

btn-circle circular

loading loading-spinner indicador de carga

Tamaño y forma son modificadores independientes del color.

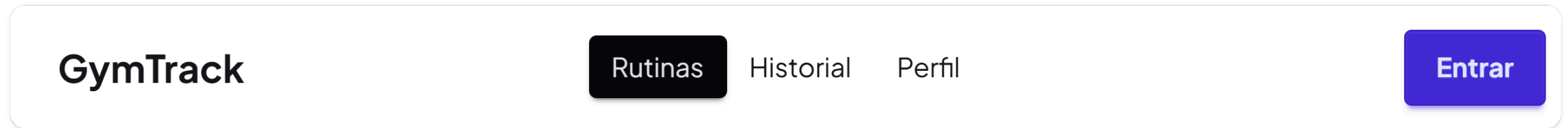
Tarjeta



```
<div class="card bg-base-100  
  shadow-sm">  
  <div class="card-body">  
    <h2 class="card-title">...</h2>  
    <p>...</p>  
    <div class="card-actions  
      justify-end">  
  
      ...  
    </div>  
  </div>  
</div>
```

`card-body`, `card-title` y `card-actions` ordenan el contenido; `card-border` y `card-side` cambian el estilo.

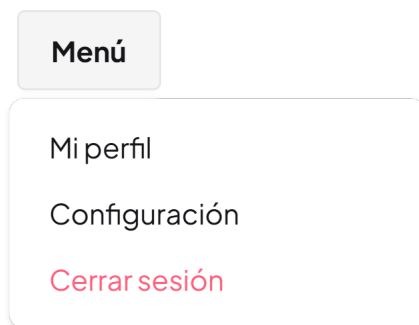
Navbar y menu



```
<div class="navbar bg-base-100">
  <div class="navbar-start">...logo...</div>
  <div class="navbar-center">
    <ul class="menu menu-horizontal">...</ul>
  </div>
  <div class="navbar-end">...acción...</div>
</div>
```

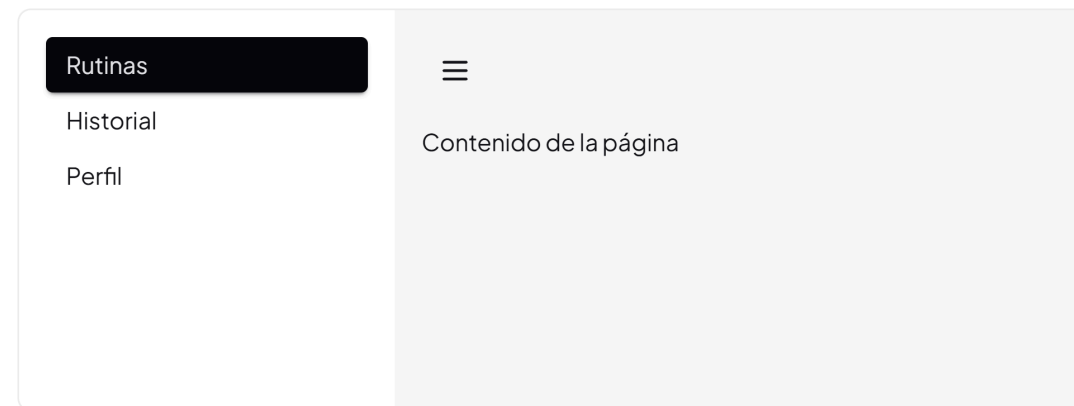
`navbar-start/center/end` reparten la barra; `menu-horizontal` y `menu-active` dibujan los enlaces.

Dropdown y drawer



dropdown

```
<div class="dropdown dropdown-end">
  <div tabindex="0" role="button"
    class="btn">Menú</div>
  <ul class="dropdown-content menu">...</ul>
</div>
```

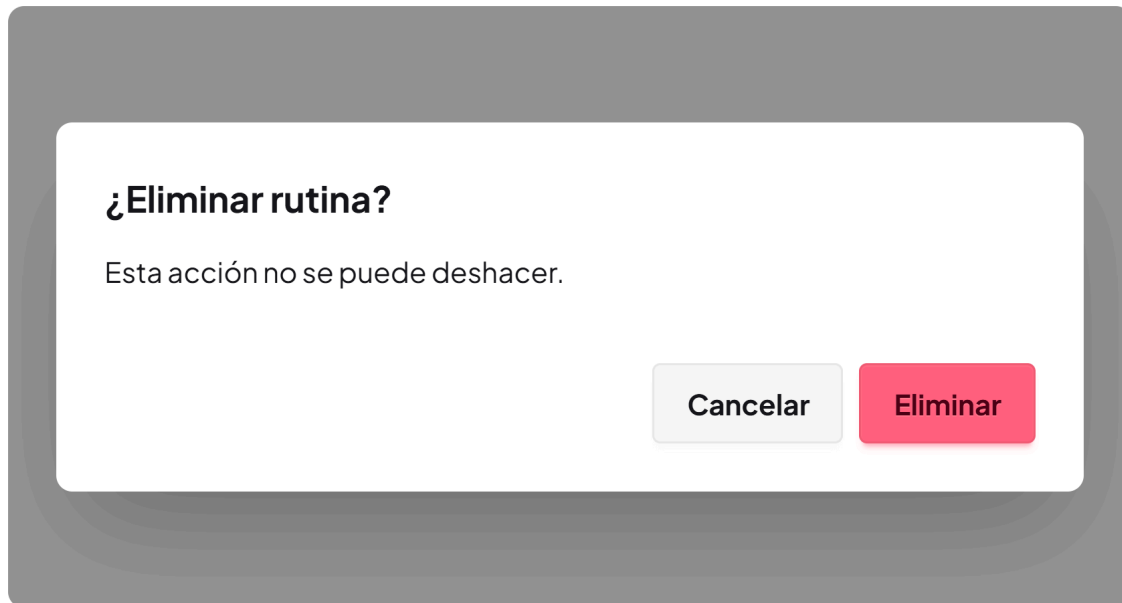


drawer

```
<div class="drawer lg:drawer-open">
  <input id="d" type="checkbox"
    class="drawer-toggle" />
  <div class="drawer-content">...</div>
  <div class="drawer-side">...</div>
</div>
```

El drawer es el panel lateral: en móvil se oculta y con `lg:drawer-open` queda abierto.

Modal



```
<button class="btn"
  onclick="m.showModal()">Abrir</button>

<dialog id="m" class="modal">
  <div class="modal-box">
    <h3 class="text-lg font-bold">...</h3>
    <div class="modal-action">
      <form method="dialog">
        <button class="btn">Cerrar</button>
      </form>
    </div>
  </div>
</dialog>
```

Usa el `<dialog>` nativo: `showModal()` abre y un formulario `method="dialog"` cierra.

Formularios: fieldset e input

Nombre de la rutina

Mínimo 3 caracteres

Correo

Escribe un correo válido

```
<fieldset class="fieldset">
  <legend class="fieldset-legend">Correo</legend>
  <input type="email" class="input w-full" />
  <p class="label">Mínimo 3 caracteres</p>
</fieldset>
```

input-error borde de error

input-primary borde de color

input-sm ... input-xl tamaños

En daisyUI 5 los campos ya traen borde; **fieldset** y **label** reemplazan a **form-control**.

Formularios: más controles

Fuerza ▼

☒ Recordarme

☐ Notificaciones

☐ Opción

select

checkbox

toggle

radio

checkbox-primary

toggle-primary

radio-secondary

Cada control tiene su clase y acepta los mismos colores y tamaños.

Badge y alert

primary secondary success warning error outline soft

Completaste 4 rutinas esta semana.

Rutina guardada correctamente.

No se pudo guardar. Revisa los campos.

```
<span class="badge badge-primary">
  Fuerza
</span>

<div role="alert" class="alert alert-info">
  <span>Completaste 4 rutinas.</span>
</div>
```

badge-outline

badge-soft

alert-soft

alert-outline

badge etiqueta; **alert** comunica un estado con **info**, **success**, **warning** o **error**.

Tailwind y daisyUI, cada uno en lo suyo

NECESIDAD	HERRAMIENTA	EJEMPLO
Componentes estándar	daisyUI	<code>btn card modal alert</code>
Colores que cambian con el tema	daisyUI	<code>bg-base-100 btn-primary</code>
Distribución de elementos	Tailwind	<code>flex grid gap-6 col-span-2</code>
Tamaño y espaciado	Tailwind	<code>p-4 w-full max-w-4xl</code>
Responsive	Tailwind	<code>md:grid-cols-3 hidden lg:block</code>
Ajustes puntuales	Tailwind	<code>rounded-full shadow-xl</code>

Se complementan: `btn btn-primary w-full md:w-auto` mezcla las dos.

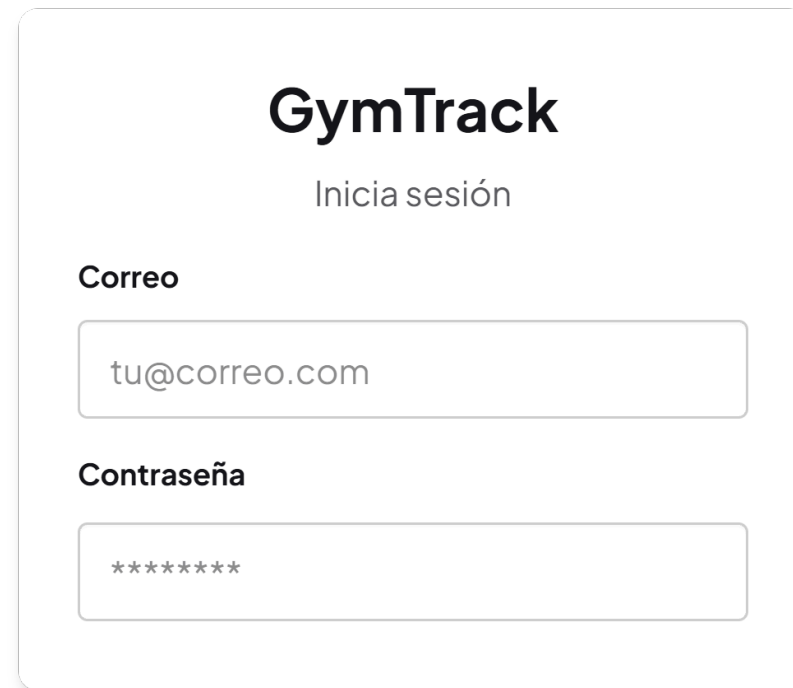
De daisyUI 4 a daisyUI 5

ANTES (V4)	AHORA (V5)
<code>form-control</code> + <code>label-text</code>	<code>fieldset</code> + <code>fieldset-legend</code> + <code>label</code>
<code>card-bordered</code> · <code>card-compact</code>	<code>card-border</code> · <code>card-sm</code>
<code>input-bordered</code> · <code>select-bordered</code>	ya no hacen falta: traen borde por defecto
<code>btn-group</code> / <code>input-group</code>	<code>join</code> + <code>join-item</code>
<code>btm-nav</code>	<code>dock</code>
<code>tabs-boxed</code> · <code>tabs-lifted</code>	<code>tabs-box</code> · <code>tabs-lift</code>
<code>footer</code> (horizontal)	<code>footer</code> <code>footer-horizontal</code>

Si un tutorial usa las clases de la izquierda, es de la versión 4.

Ejemplo integrado: pantalla de login

```
<div class="card bg-base-100 w-80 shadow-md">
  <div class="card-body">
    <h2 class="card-title justify-center">
      GymTrack
    </h2>
    <fieldset class="fieldset">
      <legend class="fieldset-legend">Correo</legend>
      <input type="email" class="input w-full" />
      <legend class="fieldset-legend">Clave</legend>
      <input type="password" class="input w-full" />
    </fieldset>
  </div>
</div>
```



GymTrack

Inicia sesión

Correo

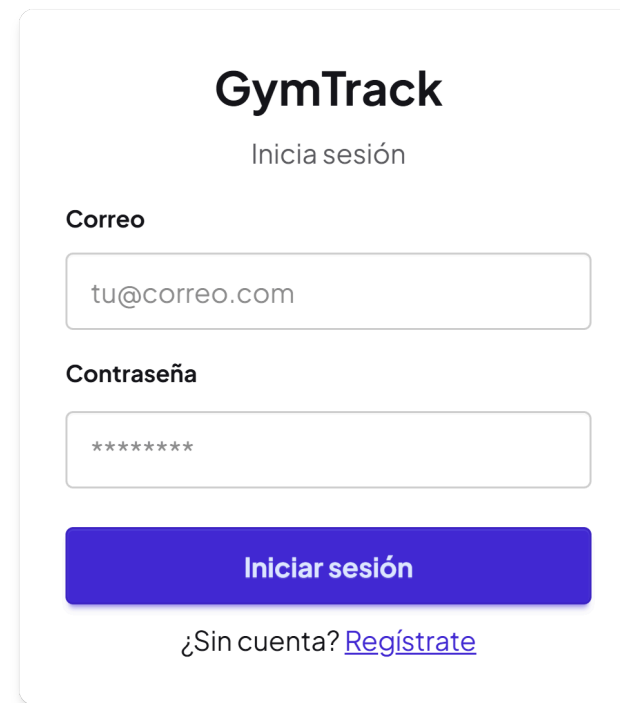
Contraseña

Primero la estructura: `card`, `card-body`, `fieldset` e `input`.

Ejemplo integrado: pantalla de login

```
<div class="card bg-base-100 w-80 shadow-md">
  <div class="card-body">
    <h2 class="card-title justify-center">...</h2>
    <fieldset class="fieldset">...</fieldset>

    <button class="btn btn-primary w-full mt-2">
      Iniciar sesión
    </button>
    <p class="text-center text-sm">
      ¿Sin cuenta?
      <a class="link link-primary">Regístrate</a>
    </p>
  </div>
</div>
```



GymTrack

Inicia sesión

Correo

Contraseña

Iniciar sesión

¿Sin cuenta? [Regístrate](#)

daisyUI aporta **card**, **input** y **btn**; Tailwind, el ancho, el espacio y el responsive.