

**React**

Frontend

# Props

- "Props" é uma abreviação para propriedades.
- Representa a forma como os dados são passados de um componente pai para um componente filho no React.
- Componentes pais podem enviar dados para componentes filhos através de props.
- Props são imutáveis. Não podem ser modificadas pelo componente filho. Props são imutáveis. Não podem ser modificadas pelo componente filho.

# Props

- No componente filho, as props são acessadas como argumentos da função de componente.
- Props podem ser utilizadas para realizar renderização condicional com base nos dados recebidos.

# Estilização condicional

- É comum a utilização de variáveis para renderizar estilos para diferentes condições
- Costuma-se utilizar um ternário para condicionar a estilização

```
export function Modal({showModal}){
  return(
    <div className={`${showModal ? "flex" : "hidden"} h-screen w-screen bg-black opacity-75`}>
      <div className="bg-white w-full h-full max-h-[260px] max-w-[300px]">
        Contéudo do Modal
      </div>
    </div>
  )
}

export default function Home(){
  return(
    <Modal showModal={true}/>
  )
}
```

# Renderização Condicional de Componente

Da mesma forma que são utilizadas condições para renderizar estilo no componente, o React permite a renderização do componente segundo condições.

Para isso, utiliza-se o ternário:

**isAuth? cond-True : cond-False**

```
export function Home({isAuth}) {
  return (
    isAuth ?
      <h1> Olá Fulano você está logado</h1>
      :
      <div>
        <h1>Deseja fazer o Login? clique no botão</h1>
        <button>Fazer Login</button>
      </div>
    )
}

export default function MeuSite(){
  return(
    <Home isAuth={true}>/>
  )
}
```

# Renderização de Componente a partir de Lista

Para a renderização de listas é utilizado o método map()

```
export function Card(props) {
  return (
    <section>
      <header>
        <img src={props.post.avatar} alt="" />
        <h1>{props.post.usuario}</h1>
      </header>
      <img src={props.post.foto} alt="" />
      <p>{props.post.texto}</p>
    </section>
  )
}

export default function Home(){
  const posts = [
    {id:1, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:2, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:3, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
  ]

  return(
    <div>
      {posts.map((post)=>(
        <Card key={post.id} post={post}/>
      ))}
    </div>
  )
}
```

# Renderização de Componente a partir de Lista

Exemplos de renderização de listas

Mais pedidos [Ver mais](#)

Carrefour Hiper - Fátima  
7 • Mercado • 1.1 km  
14 min • R\$ 9,49  
  
Carrefour

Carrefour Hiper - Bezerra de Menezes  
3 • Mercado • 2.8 km  
10 min • R\$ 9,99  
  
Carrefour

Gbarbosa - Fortaleza  
5 • Mercado • 1.7 km  
18 min • R\$ 10,49  
  
gbarbosa cencosud

BIG by Carrefour - Fátima  
5 • Mercado • 1.1 km  
4 min • R\$ 9,49  
  
BIG by Carrefour

Atacadão Fortaleza Br116  
7 • Mercado • 4.0 km  
50 min • R\$ 13,99  
  
ATACADÃO

Pão de Açúcar - Fátima  
5 • Mercado • 0.4 km  
5 min • R\$ 8,99  
  
Pão de Açúcar

[Home](#) [Search](#) [Cart](#) [Profile](#)

Instagram Entrar Abrir aplicativo



Socorrista do Samu viraliza ao cantar para idosa em atendimento na Bahia

1.718 curtidas

opovoonline • O condutor socorrista Joseval de Jesus viralizou nas redes sociais após sua gravação cantando uma música do bloco Ilê...  
mais

Ver todos os 39 comentários HÁ 46 MINUTOS

 opovoonline • Seguir [...](#)

DELTA-V  
PROGRAMA DE ACELERAÇÃO DE STARTUPS

CHEGOU  
A HORA DE

[Home](#) [Search](#) [Cart](#) [Profile](#)

you excited? [bit.ly/1eRsMHI](#)

Wes Miller retweeted Brandon Paddock @BrandonLive 1h Hey #WinApps fans. Who would come to a tweet-up if we did it next Monday evening? #WinDev cc @Clarkezone @HumanCompiler @codefoster

PostSecret @postsecret 1h Billy Nye Debates Creationist Ken Ham Happening Now (livestream) [on.digg.com/1boREpe](#)

Taylor Buley @taylorbuley 1h Wife: "what cha doing?" Response: "Just looking at some cable porn." [media.dma.mil/2013/Jan/17/20...](#)

newballpark retweeted Mike Rosenberg @RosenbergMerc 1h MLB "remains committed to working toward a solution" on A's to San Jose,

[Twitter](#) [Bell icon](#) [Email icon](#) [User icon](#) [Search icon](#) [More icon](#)

# Renderização de Componente a partir de Lista

Parte por parte

```
export default function Home(){
  const posts = [
    {id:1, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:2, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:3, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
  ]
```

```
return(
  <div>
    {posts.map((post)=>(
      <Card key={post.id} post={post}/>
    ))}
  </div>
)
```

O react requisita um identificador único para cada componente renderizado no método map()

# Renderização de Componente a partir de Lista

Parte por parte

```
export default function Home(){
  const posts = [
    {id:1, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:2, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
    {id:3, avatar:"www.w3school.com/avatar", usuario:"Fulano", foto:"https://picsum.photos/", texto:"A vida é bela!"},
  ]
}
```

```
export function Card(props) {
  return (
    <section>
      <header>
        <img src={props.post.avatar} alt="" />
        <h1>{props.post.usuario}</h1>
      </header>
      <img src={props.post.foto} alt="" />
      <p>{props.post.texto}</p>
    </section>
  )
}
```

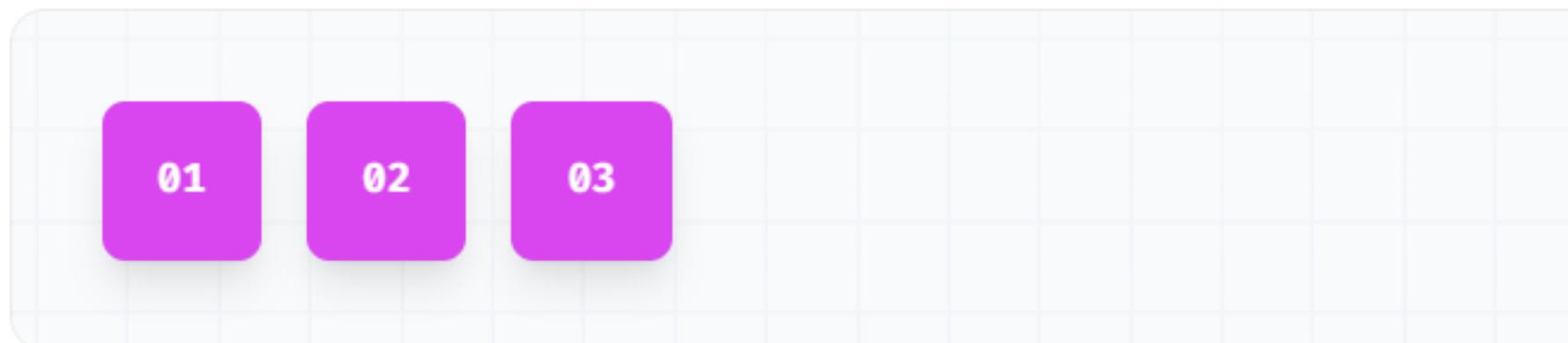
Vamos à prática

# Flexbox

## Flex directions

### Row

Use `flex-row` to position flex items horizontally in the same direction as



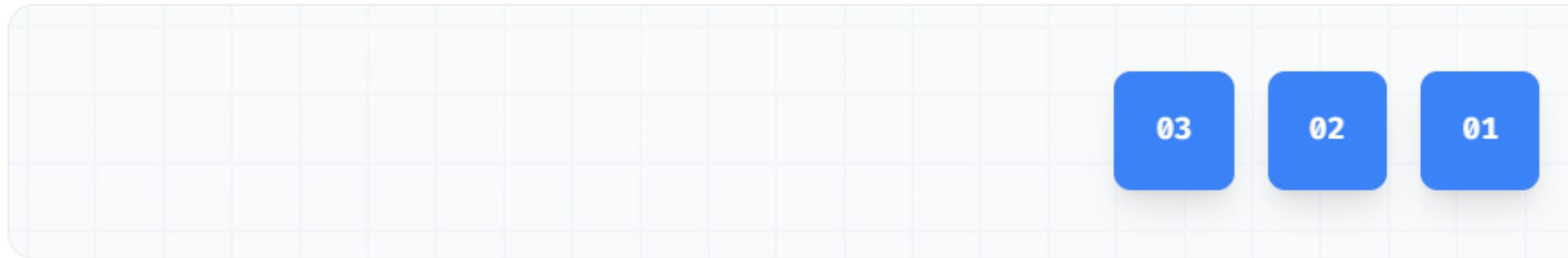
```
<div class="flex flex-row ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```

# Flexbox

## Flex directions

### Row reversed

Use `flex-row-reverse` to position flex items horizontally in the opposite direction:



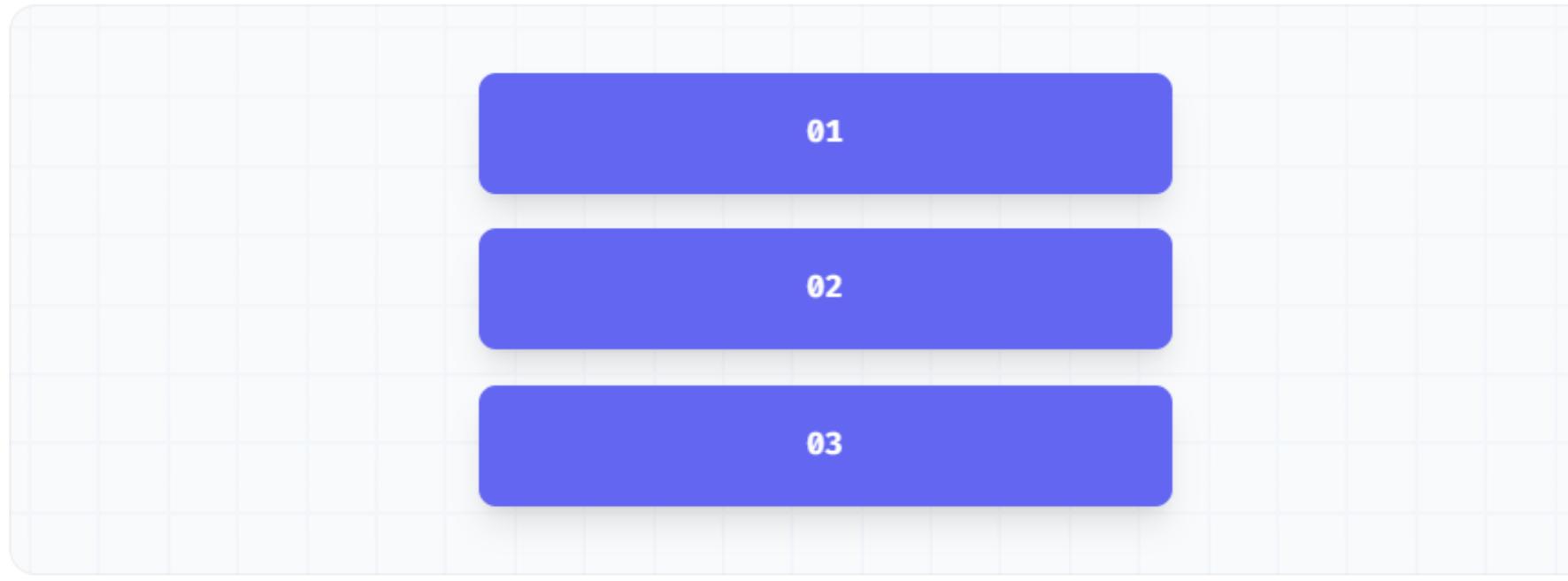
```
<div class="flex flex-row-reverse ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```

# Flexbox

## Flex directions

### Column

Use `flex-col` to position flex items vertically:



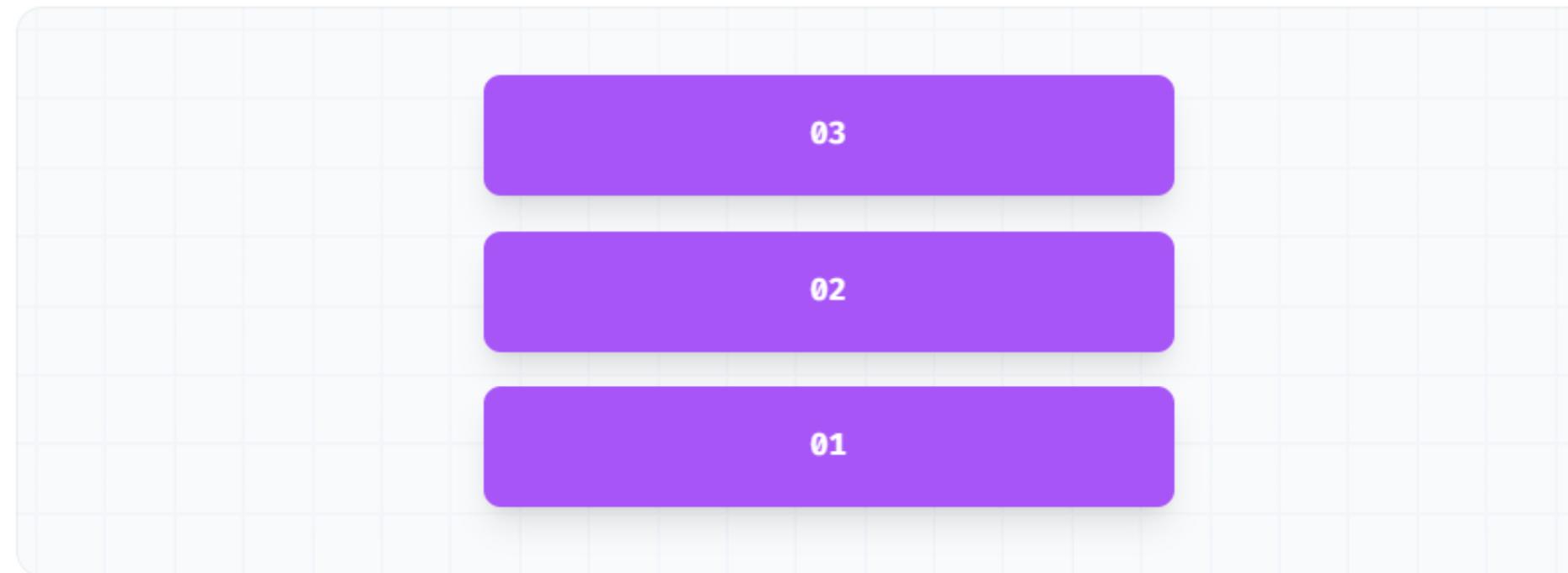
```
<div class="flex flex-col ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# Flexbox

## Flex directions

### Column reversed

Use `flex-col-reverse` to position flex items vertically in the opposite direction:



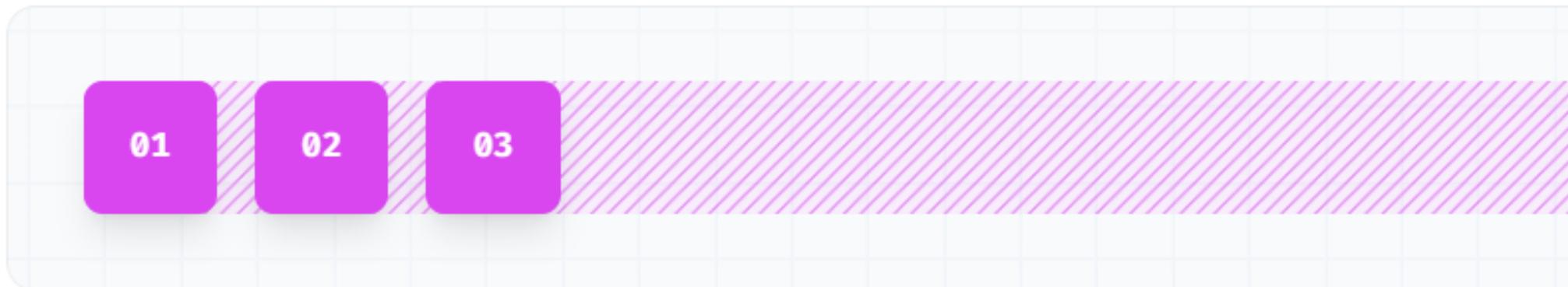
```
<div class="flex flex-col-reverse ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# Flexbox

## Justify content

### Start

Use `justify-start` to justify items against the start of the container's main axis:



```
<div class="flex justify-start ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# Flexbox

## Justify content

### Center

Use `justify-center` to justify items along the center of the container's main axis:



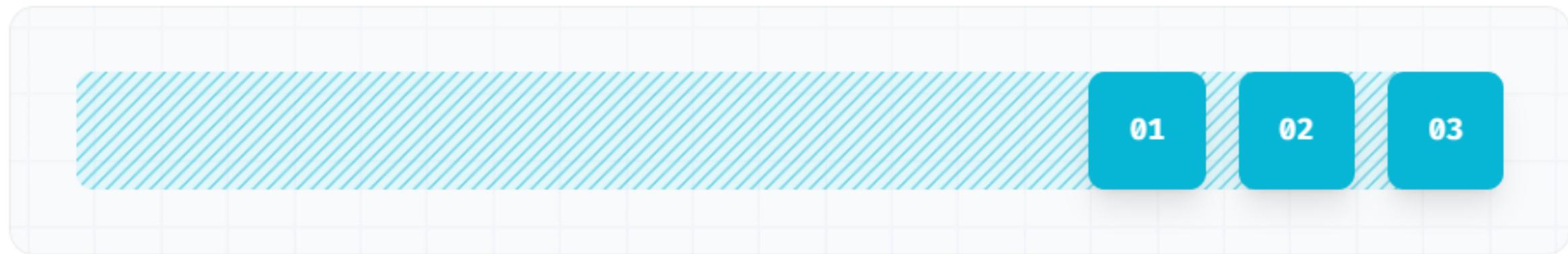
```
<div class="flex justify-center ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# Flexbox

## Justify content

### End

Use `justify-end` to justify items against the end of the container's main axis:



```
<div class="flex justify-end ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# Flexbox

## Justify content

### Space between

Use `justify-between` to justify items along the container's main axis such that there is an equal amount of space between each item:



```
<div class="flex justify-between ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```

# Flexbox

## Justify content

### Space around

Use `justify-around` to justify items along the container's main axis such that there is an equal amount of space on each side of each item:



```
<div class="flex justify-around ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```

# Flexbox

## Justify content

### Space evenly

Use `justify-evenly` to justify items along the container's main axis such that there is an equal amount of space around each item, but also accounting for the doubling of space you would normally see between each item when using `justify-around`:



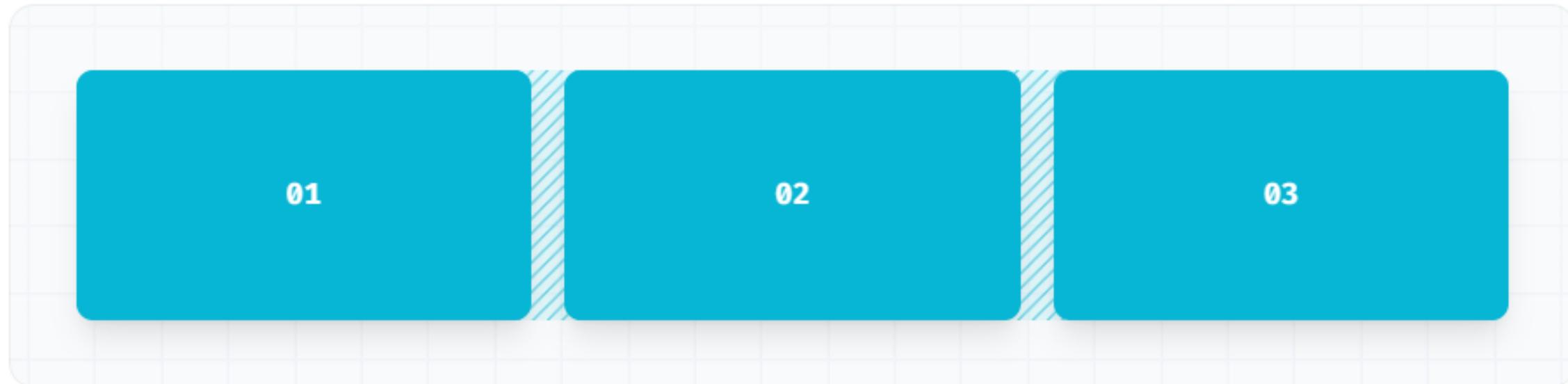
```
<div class="flex justify-evenly ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```

# Flexbox

## Align-items

### Stretch

Use `items-stretch` to stretch items to fill the container's cross axis:



```
<div class="flex items-stretch ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

# Flexbox

## Align-items

### Start

Use `items-start` to align items to the start of the container's cross axis:



```
<div class="flex items-start ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

# Flexbox

## Align-items

### Center

Use `items-center` to align items along the center of the container's cross axis:



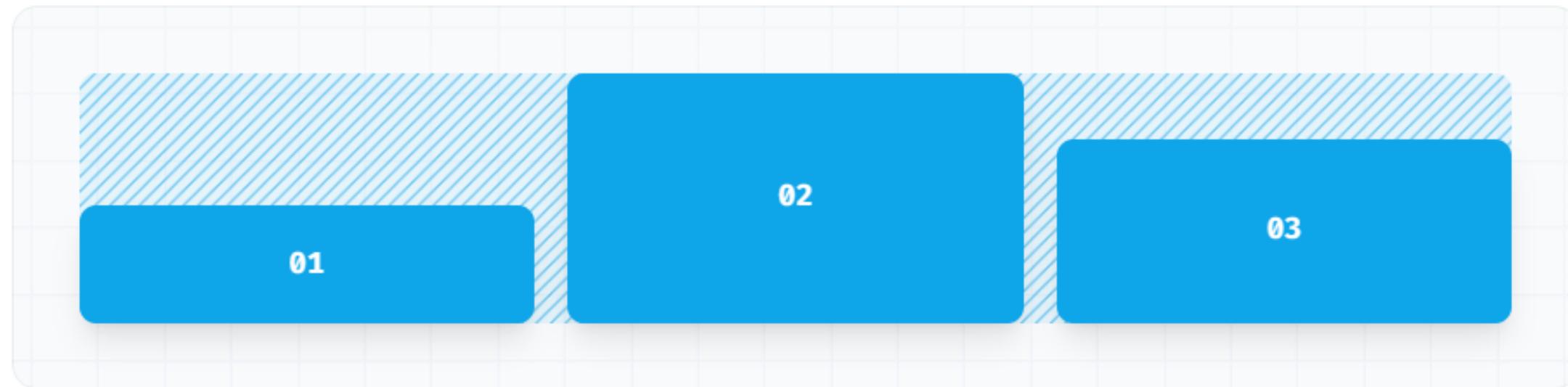
```
<div class="flex items-center ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

# Flexbox

## Align-items

### End

Use `items-end` to align items to the end of the container's cross axis:



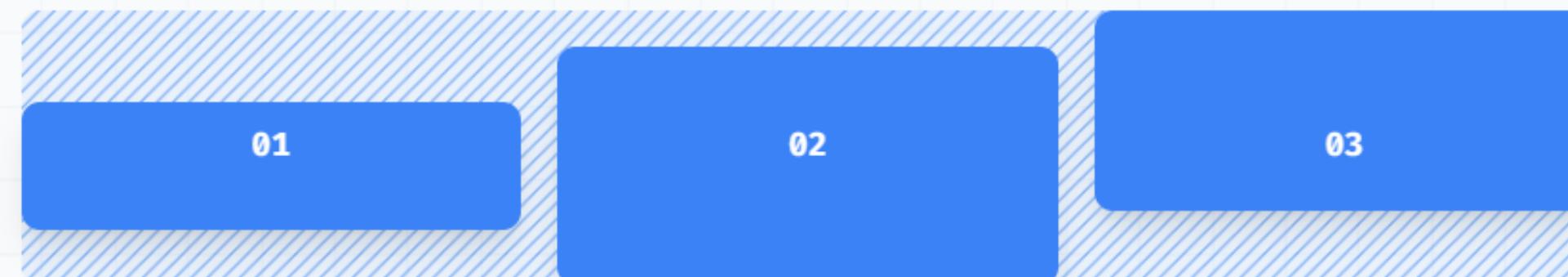
```
<div class="flex items-end ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

# Flexbox

## Align-items

### Baseline

Use `items-baseline` to align items along the container's cross axis such that all of their baselines align:



```
<div class="flex items-baseline ...">
  <div class="pt-2 pb-6">01</div>
  <div class="pt-8 pb-12">02</div>
  <div class="pt-12 pb-4">03</div>
</div>
```

# Mobile first

- O Tailwind oferece classes responsivas para facilitar a adaptação de estilos a diferentes tamanhos de tela.
- Exemplo: sm, md, lg, xl para definir estilos específicos para small, medium, large e extra-large. A aplicação de prefixos responsivos como md:, lg:, etc., permite personalizar estilos para diferentes breakpoints.
- Classes como hidden, block, inline, etc., podem ser combinadas com classes responsivas para controlar a visibilidade de elementos em diferentes dispositivos.

# Vamos à prática