Box Model

Formatação Visual Básica (CSS)

Caixas Básicas (Basic Boxes)

- Todo elemento gera uma ou mais **caixas retangulares** (element boxes)
- Cada caixa possui:
  - **Área de conteúdo** (content area)
  - Opcionalmente: **padding**, **borda**, **outline** e **margem**

Exemplo

```
<div class="caixa">Exemplo de caixa</div>
```

Exemplo

```
.caixa {
  margin: 10px;
  padding: 20px;
  border: 3px solid blue;
}
```

Eixos básicos

- A direção dos fluxo de ***block*** e do fluxo ***inline*** é condicionada a forma que a escrita do idioma é realizada. Por exemplo, em português a escrita ocorre da esquerda para direita e de cima para baixo quando criamos novas linhas.

- **Block flow direction**: direção em que caixas de bloco são empilhadas (ex.: de cima para baixo em português e inglês)
- **Inline base direction**: direção em que o texto é escrito (ex.: da esquerda para a direita em português e inglês)

Fluxo normal (Normal flow)

- A maioria dos elementos segue o **fluxo normal**
- Só saem desse fluxo se forem:
  - `float`;
  - `position` diferente de `static` (padrão dos elementos);
  - `flex`;
  - `grid`;
  - `table`.

Block box

- Gerado por elementos como `<p>`
- Produzem quebras de linha antes e depois

Exemplo

```
<p>Parágrafo 1</p>
<p>Parágrafo 2</p>
```

- Elementos que geram block box (fluxo em block) ocupam toda a linha na horizontal (no pãdro de escrita em poruguês)

Inline box

- Gerado por elementos como `<span>`, `<strong>`
- **Não** geram quebras de linha

Exemplo

```
<p>Texto <strong>em negrito</strong> dentro de uma linha.</p>
```

- Estes elementos ocupam apenas o espaço para o seu conteúdo. Não geram novas linhas.

Nonreplaced e Replaced Elements

- **Não substituídos (Nonreplaced)**: o navegador renderiza seu próprio conteúdo (ex.: `<p>`, `<div>`)
- **Substituídos (Replaced)**: agem como "caixas de espaço reservado" para outro conteúdo (ex.: `<img>`)

Exemplo

```
<img src="foto.jpg" alt="Exemplo de imagem">
```

- Quando solicitamos uma página simples com apenas uma imagem, na verdade, estamos realizando duas requisições. A requisição para página e outro para a imagem (replaced element).
- **Este elemento é desenhado na página, mas não faz parte dela.**

Elemento raiz (root element)

- Em documentos HTML, o elemento `<html>` é o **root element**
- Ele gera o **bloco de contenção inicial** (initial containing block)
- Esse bloco corresponde à **janela de visualização** (viewport)
- Quando você estudar `position`, notará a importância de entender os blocos de contenção (containing block)

O bloco de contenção (Containing Block)

- É o **contexto de layout** de uma caixa
- Geralmente, é o **conteúdo da caixa ancestral mais próxima**

Exemplo

```
<div class="pai">
  <p class="filho">Exemplo</p>
</div>
```

Exemplo

```
.pai {
  width: 400px;
}
.filho {
  width: 50%; /* relativo ao pai */
}
```

Propriedades de margens, bordas e padding

- Podem ser definidas individualmente:
  - `margin-top`, `border-left`, `padding-block-start`, etc.
- **Background** aplica-se por padrão **até a borda** (inclui padding)
- **Margens são sempre transparentes**
- Padding/borda **não podem ter valores negativos**
- Margens **podem** ser negativas
- Bordas podem usar **cores, estilos ou imagens**

Alterando a Exibição de Elementos

Alterando a Exibição de Elementos

- Controlada pela propriedade `display`
- Permite mudar como o navegador exibe o elemento

Exemplo

```
p {
  display: inline; /* parágrafo exibido inline */
}
```

Mudando papéis (Changing roles)

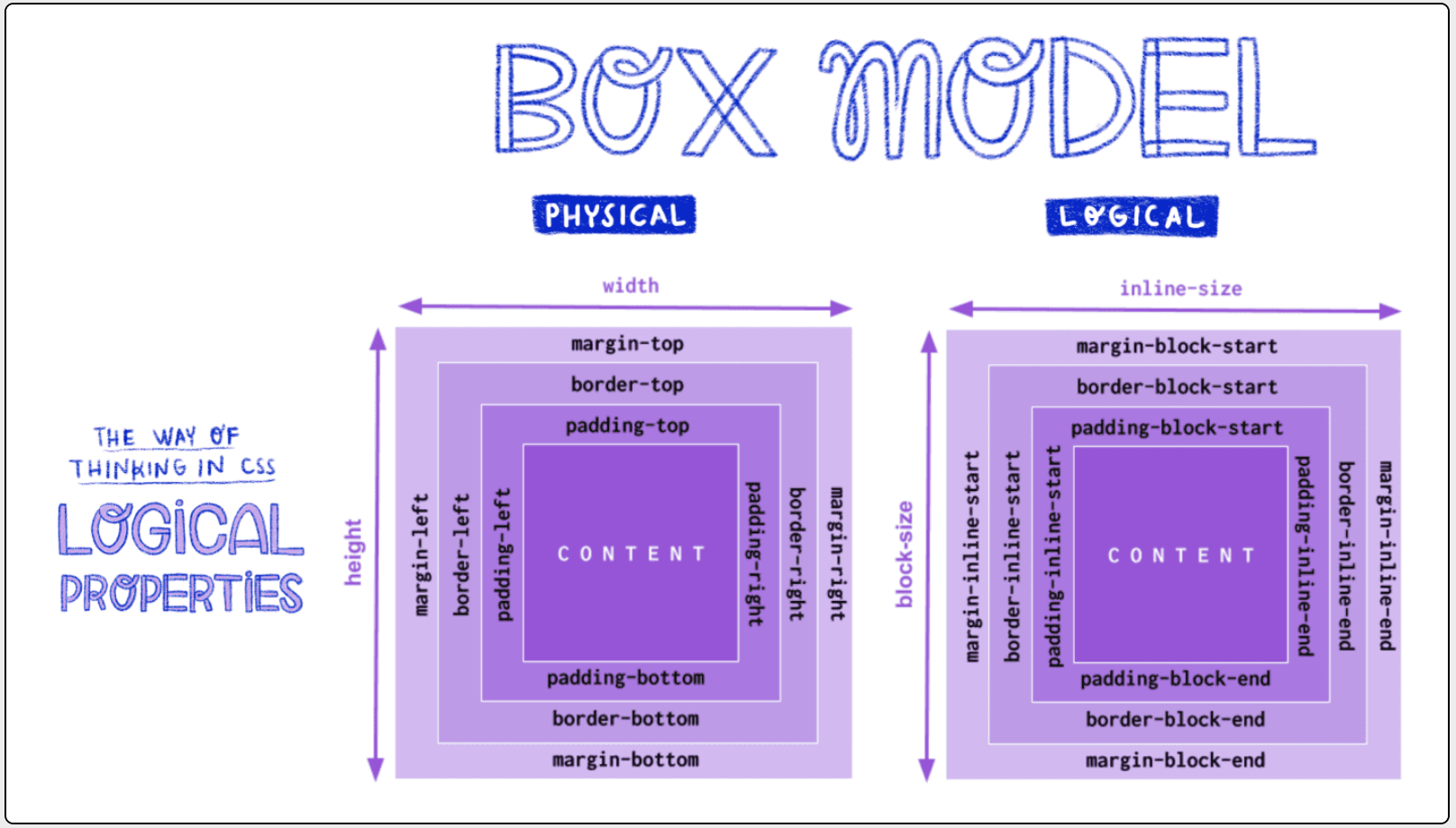
- Alterar `display` muda apenas o **papel de exibição**
- Não muda a **natureza do elemento**
- Um `<p>` exibido como `inline` **ainda é um parágrafo**

CSS e Acessibilidade

- CSS afeta principalmente **apresentação**
- Mas pode afetar **acessibilidade**:
  - Contraste de cor
  - Ordem visual ≠ ordem de leitura
  - Elementos ocultos visíveis para leitores de tela

Manipulando caixas de bloco

Manipulando caixas de bloco



Fonte: https://ultimatecourses.com/blog/css-logical-properties

Manipulando caixas de bloco

- CSS lida com:
  - **Eixo de bloco** (vertical na maioria dos idiomas ocidentais)
  - **Eixo inline** (horizontal em português/inglês)
- **Block size**: altura da área de conteúdo
- **Inline size**: largura da área de conteúdo

Exemplo

```
.caixa {
  block-size: 100px;
  inline-size: 200px;
}
```

Bordas Start e End

- CSS lógico usa **start** e **end** em vez de top/bottom/left/right
- Isso torna o CSS **adaptável a diferentes direções de escrita**

Exemplo

```
.caixa {
  margin-inline-start: 20px;
}
```

Tamanhos lógicos de elementos

Tamanhos lógicos de elementos

- Propriedades permitem definir tamanhos de acordo com:
  - **Eixo de bloco** (altura no fluxo normal)
  - **Eixo inline** (largura no fluxo normal)
- Normalmente, o **tamanho do bloco é automático** (`auto`)
- Quando `auto` , o tamanho é definido pelo **conteúdo**

Content-based sizing

Palavras-chave importantes:

- `min-content` → menor largura possível sem quebrar palavras
- `max-content` → largura necessária para caber todo o conteúdo em **uma linha**
- `fit-content` → tenta ajustar o tamanho ao conteúdo, sem ultrapassar o contêiner

Exemplo

```
.ex1 { inline-size: min-content; }
.ex2 { inline-size: max-content; }
.ex3 { inline-size: fit-content; }
```

Exemplo prático: min-content

Exemplo

```
<div class="ex1">
  CSS é poderoso
</div>
```

Exemplo

```
.ex1 {
  inline-size: min-content;
  border: 1px solid;
}
```

↗ Caixa ficará **tão estreita quanto a menor palavra** (quebra forçada).

Exemplo prático: max-content

Exemplo

```
<div class="ex2">
  CSS é poderoso
</div>
```

Exemplo

```
.ex2 {
  inline-size: max-content;
  border: 1px solid;
}
```

↗ Caixa ficará **larga o suficiente para todo o conteúdo em uma linha**.

Exemplo prático: fit-content

Exemplo

```
<div class="ex3">
  CSS é poderoso
</div>
```

Exemplo

```
.ex3 {
  inline-size: fit-content;
  max-inline-size: 300px;
  border: 1px solid;
}
```

↗ Caixa ajusta ao conteúdo, mas **respeita o limite máximo**.

Tamanhos mínimos e máximos (Min/Max)

Tamanhos mínimos e máximos (Min/Max)

- Podemos definir limites superior e inferior:
  - `min-inline-size` / `max-inline-size`
  - `min-block-size` / `max-block-size`

Exemplo

```
img {
  max-inline-size: 100%;
  block-size: auto;
}
```

👉 Imagem nunca ultrapassa a largura do contêiner.

Altura e Largura (Height e Width)

Altura e Largura (Height e Width)

- São propriedades **físicas**, não lógicas
- Referem-se a **top/right/bottom/left**
- `height` = altura fixa
- `width` = largura fixa ou percentual

Exemplo

```
div {
  width: 50%;   /* metade do bloco pai */
  height: 200px;
}
```

Observações

- `height` e `width` não se aplicam a elementos **inline não substituídos (inline nonreplaced)**. Exemplo: `<a>`.
- Se mudarmos o `display` para `inline-block` ou `block`, passam a funcionar

Exemplo

```
span {
  display: inline-block;
  width: 100px;
  height: 50px;
}
```