CSS3: Estilizando Páginas Web com Estilo - Turma 2025B

3.1.1 Propriedade display

A propriedade **display** é uma das propriedades mais fundamentais para controlar o layout de elementos na web. Ela determina o tipo de caixa de renderização usada para um elemento, influenciando diretamente como esse elemento é tratado em termos de fluxo de documento, layout e comportamento de visualização. Esta propriedade é crucial para definir tanto os tipos de exibição internos quanto externos de um elemento. O tipo de exibição externo de um elemento determina como ele se comporta dentro do fluxo de layout geral da página, ou seja, como ele interage com outros elementos em termos de sequência e posicionamento. Por outro lado, o tipo de exibição interno afeta a organização dos elementos filhos dentro do próprio elemento.

Vejamos a seguir neste documento mais informações sobre estes tipos comuns de valores para a propriedade display.

Palavra-chave block

A palavra-chave "**block**" especifica um tipo de exibição externa do elemento, que é essencialmente como ele se comporta dentro do fluxo de layout geral da página. Quando um elemento é definido como **block**, ele começa em uma nova linha e se estende na largura máxima disponível até as bordas do contêiner pai, a menos que a largura seja explicitamente definida. Isso faz com que o elemento ocupe uma linha inteira por si só no fluxo do documento.

Considere o seguinte exemplo, conforme quadro abaixo:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Posicionamento de Layout</title>
    <style>
        body {
            background-color: #E7E9EB;
        }
        #myDIV {
            height: 150px;
            background-color: #FFFFFF;
        }
        #myDIV p {
            background-color: blue;
            padding: 10px;
            margin: 20px;
            display: block;       /* definição do tipo de posicionamento */
        }
        #myDIV span {
            background-color: red;
            padding: 10px;
            color: white;
        }
    </style>
</head>
<body>
    <h1>Exemplo de uso da propriedade "display"</h1>
    <div id="myDIV">
        <p>
            <span>Texto 1</span>
            <span>Texto 2</span>
            <span>Texto 3</span>
        </p>
        <p>
            <span>Texto 1</span>
            <span>Texto 2</span>
            <span>Texto 3</span>
        </p>
    </div>
</body>
</html>
```

Agora, veja o resultado no navegador, conforme figura abaixo:

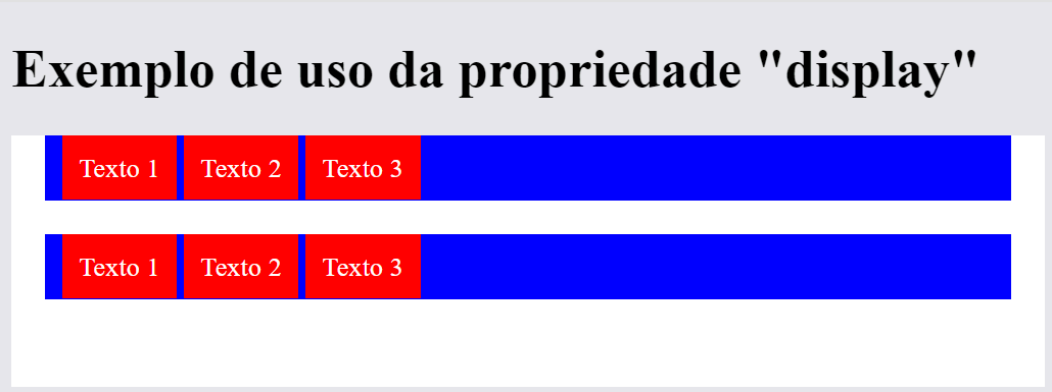

Figura 1 - Exemplo de uso da palavra-chave "block"

Descrição da imagem: No navegador, você observa o título seguido por uma divisão branca contendo dois parágrafos azuis. Cada parágrafo contém três spans vermelhos com texto branco, organizados em blocos dentro dos parágrafos.

Observe que o elemento **<p>** definido como **block** começa em uma nova linha e se estende na largura máxima disponível até as bordas do contêiner pai (** **).

Palavra-chave inline

A palavra-chave "**inline**" também especifica um tipo de exibição externa do elemento, que é essencialmente como ele se comporta dentro do fluxo de layout geral da página. Quando um elemento é definido como **inline**, ele não começa em uma nova linha e não aceita a definição de largura (**width**) e altura (**height**). Em vez disso, ele se comporta como parte de uma sequência de texto ou outros elementos inline, ocupando apenas o espaço necessário para o seu conteúdo.

Para mostrar o uso de **display: inline;**, vamos considerar o exemplo anterior, alterando apenas a seguinte linha do código CSS que trata sobre a definição do tipo de posicionamento do elemento **<p>**, conforme quadro abaixo:

```
display: inline;   /* definição do tipo de posicionamento */
```

Agora, veja o resultado no navegador, conforme figura abaixo:

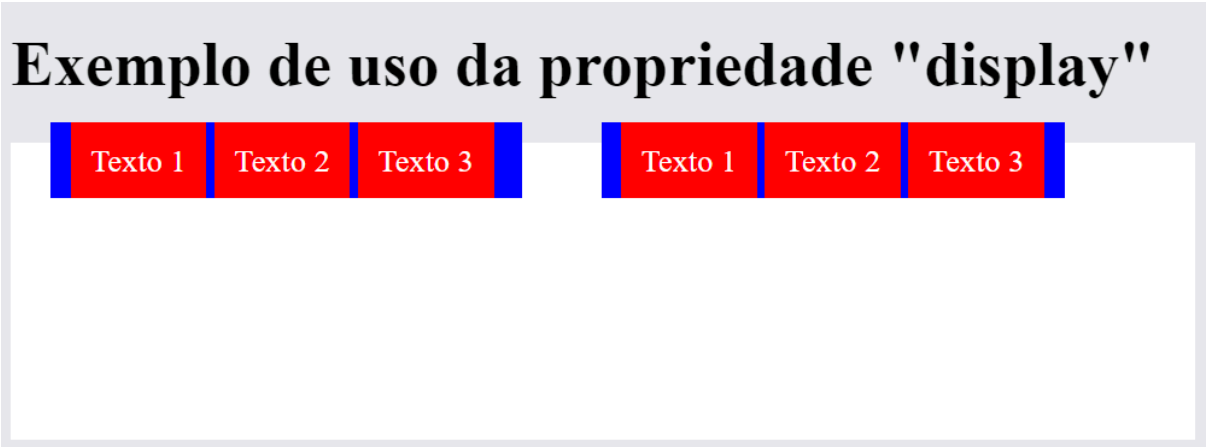

Figura 2 - Exemplo de uso da palavra-chave "inline"

Descrição da imagem: No navegador, você observa o título seguido por uma divisão branca contendo dois parágrafos azuis. Cada parágrafo contém três spans vermelhos com texto branco. Os parágrafos e os spans estarão dispostos em linha, ou seja, os spans aparecem lado a lado dentro dos parágrafos e os parágrafos aparecem na mesma linha se houver espaço suficiente.

Observe que o elemento **<p>** definido como **inline** começa em uma nova linha e se estende na largura máxima disponível até as bordas do contêiner pai (** **).

Palavra-chave inline-block

A propriedade **display: inline-block;** combina características dos tipos de exibição **inline** e **block**. Ela é usada para fazer elementos se comportarem como **inline**, alinhando-se na mesma linha que elementos adjacentes e conteúdo de texto, mas mantendo as características de dimensionamento de **block**, como a capacidade de definir largura (**width**) e altura (**height**).

Para mostrar o uso de **display: inline-block;**, vamos considerar o exemplo anterior, alterando a linha do código CSS que trata sobre a largura e definição do tipo de posicionamento do elemento **<p>**, conforme quadro abaixo:

```
width: 300px;
display: inline-block;   /* definição do tipo de posicionamento */
```

Agora, veja o resultado no navegador, conforme figura abaixo:

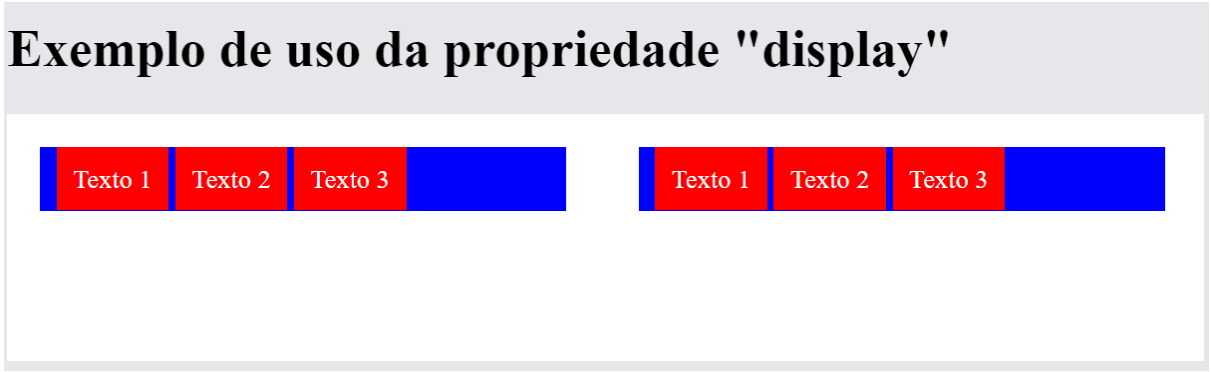

Figura 3 - Exemplo de uso da palavra-chave "inline-block"

Descrição da imagem: No navegador, você observa o título seguido por uma divisão branca contendo dois parágrafos azuis, cada um com largura de 300 pixels. Dentro de cada parágrafo, os spans vermelhos com texto branco são exibidos em linha. Os parágrafos são exibidos um ao lado do outro se houver espaço suficiente, ou empilhados verticalmente se não houver.

Palavra-chave none

A propriedade **display: none;** é usada para remover completamente um elemento do fluxo do documento. Quando aplicada, a propriedade faz com que o elemento e todos os seus filhos não sejam renderizados, como se não existissem na página.

Para mostrar o uso de **display: none;**, vamos considerar o exemplo anterior, alterando apenas a seguinte linha do código CSS que trata sobre a definição do tipo de posicionamento do elemento **<p>**, conforme quadro abaixo:

```
display: none;   /* definição do tipo de posicionamento */
```

Veja o resultado no navegador, conforme figura abaixo:

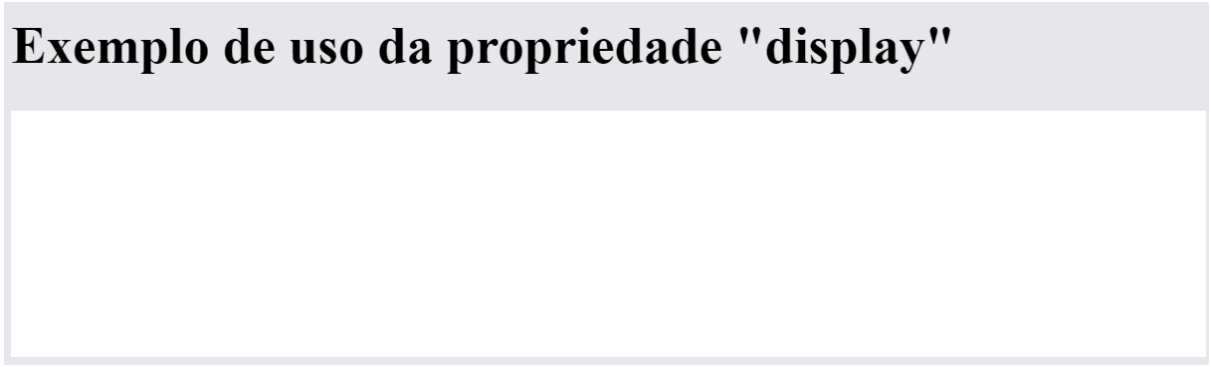

Figura 4 - Exemplo de uso da propriedade display="none"

Descrição da imagem: No navegador, você observa o título seguido por uma divisão branca vazia. Os parágrafos e spans não são exibidos porque a propriedade display: none oculta os parágrafos completamente.

OBS: Isso difere de outras técnicas de ocultação, como **visibility: hidden;**, que oculta o elemento mas ainda reserva espaço para ele no layout.

Palavra-chave flex

A propriedade **display: flex;** faz parte do Flexbox Layout (Flexible Box), um modelo de layout unidimensional projetado para distribuir espaço e alinhar itens de forma eficiente em contêineres, mesmo quando o tamanho dos itens é desconhecido ou dinâmico. Definir um elemento com **display: flex;** torna-o um contêiner flexível, fazendo com que seu comportamento de layout siga as regras do Flexbox. Os filhos diretos deste contêiner se tornam itens flexíveis.

Ao utilizar o Flexbox, é crucial entender que todas as operações estão baseadas em dois eixos: o **eixo principal** e o **eixo transversal**. O eixo principal é estabelecido pela propriedade **flex-direction**, enquanto o eixo transversal é perpendicular a ele. Compreender esses eixos é essencial, pois eles são os pilares centrais do Flexbox, influenciando diretamente o alinhamento e a distribuição dos elementos dentro do contêiner.

A propriedade **flex-direction** pode tem quatro valores possíveis:

- **row:** Este é o valor padrão. Os itens são dispostos na mesma direção do texto, da esquerda para a direita em línguas escritas da esquerda para a direita (como o português), e da direita para a esquerda em línguas de direção oposta. Os itens são alinhados ao longo do eixo horizontal.
- **row-reverse:** Semelhante ao **row**, mas inverte a direção dos itens. Ou seja, se o row alinha os itens da esquerda para a direita, row-reverse os alinhará da direita para a esquerda, independentemente da direção do texto da língua.
- **column:** Os itens são dispostos verticalmente, de cima para baixo. Isso é útil para layouts empilhados onde os itens precisam ser organizados um sobre o outro ao longo do eixo vertical.
- **column-reverse:** Como o **column**, mas inverte a ordem dos itens, alinhando-os de baixo para cima. Esta configuração pode ser útil para layouts especiais onde a ordem visual invertida é necessária.

O eixo transversal é sempre perpendicular ao eixo principal. Portanto, quando a propriedade **flex-direction** é configurada para alinhar os itens ao longo de linhas, seja como **row** ou **row-reverse**, o eixo transversal se alinha automaticamente na direção vertical, similarmente ao que acontece nas configurações **column** ou **column-reverse**.

Para esclarecer estes conceitos, considere o seguinte exemplo, conforme quadro abaixo:

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Posicionamento de Layout</title>
    <style>
        .flex-container {
            display: flex;
            margin-bottom: 20px;  /* Espaço entre cada contêiner */
        }
        .row {
            flex-direction: row;
        }
        .row-reverse {
            flex-direction: row-reverse;
        }
        .column {
            flex-direction: column;
        }
        .column-reverse {
            flex-direction: column-reverse;
        }
        .flex-item {
            width: 50px;
            height: 50px;
            background-color: lightblue;
            border: 1px solid darkblue;
            display: flex;
            align-items: center;
            justify-content: center;
            margin: 2px;
        }
    </style>
</head>
<body>
    <div class="flex-container row">
        <div class="flex-item">1</div>
        <div class="flex-item">2</div>
        <div class="flex-item">3</div>
    </div>
    <div class="flex-container row-reverse">
```

```
        <div class="flex-item">4</div>
        <div class="flex-item">5</div>
        <div class="flex-item">6</div>
    </div>
    <div class="flex-container column">
        <div class="flex-item">7</div>
        <div class="flex-item">8</div>
        <div class="flex-item">9</div>
    </div>
    <div class="flex-container column-reverse">
        <div class="flex-item">10</div>
        <div class="flex-item">11</div>
        <div class="flex-item">12</div>
    </div>
</body>
</html>
```

Veja o resultado no navegador, conforme figura abaixo:

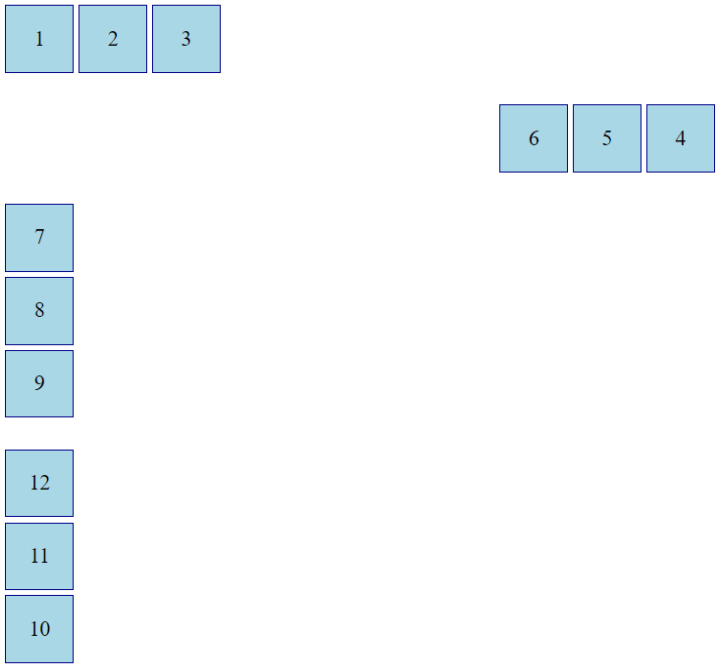

Figura 5 - Exemplo de uso da propriedade display=flex;

Descrição da imagem: No corpo do documento, há quatro contêineres flexíveis (flex-container), cada um com três itens (flex-item) numerados de 1 a 12. Cada item flexível (flex-item) é um quadrado de 50x50 pixels com fundo azul claro e borda azul escura, e os itens são centralizados tanto vertical quanto horizontalmente dentro de cada contêiner. Os itens 1, 2 e 3 são dispostos em uma linha, da esquerda para a direita. Os itens 4, 5 e 6 são dispostos em uma linha, mas da direita para a esquerda. Os itens 7, 8 e 9 são dispostos em uma coluna, de cima para baixo. Os itens 10, 11 e 12 são dispostos em uma coluna, mas de baixo para cima.

Palavra-chave grid

A propriedade **display: grid;** é uma parte fundamental do Grid Layout, que é um sistema poderoso e versátil para criar layouts de página baseados em grades bidimensionais. Quando você define **display: grid;** em um elemento, esse elemento se torna um grid container e seus filhos diretos se tornam grid items. O contêiner **grid** define o espaço dentro do qual você pode criar um layout baseado em linhas e colunas.

Você pode definir linhas (**grid-template-rows**) e colunas (**grid-template-columns**) com tamanhos fixos, flexíveis ou uma combinação de ambos. Os tamanhos podem ser especificados em unidades fixas (como pixels), em percentuais, ou usando a unidade fracionária (fr), que distribui o espaço disponível proporcionalmente.

O Grid Layout permite que você crie áreas no layout ao nomear seções do **grid**. Isso é feito usando **grid-template-areas**, o que facilita a referência a essas áreas quando você posiciona seus itens de grid.

Para esclarecer estes conceitos veja no quadro abaixo um exemplo:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Posicionamento de Layout</title>
    <style>
        .grid-container {
            display: grid;
            grid-template-columns: 1fr 3fr;
            /* Duas colunas, uma menor para nav e aside, e uma maior para o conteúdo principal */
            grid-template-rows: auto 1fr auto;
            /* Três linhas: cabeçalho, conteúdo principal, rodapé */
            grid-template-areas:
                "header header"
                "nav main"
                "aside main"
                "footer footer";
            gap: 10px;
            height: 100vh;
        }
        header {
            grid-area: header;
            background: lightblue;
        }
        nav {
            grid-area: nav;
            background: lightgreen;
        }
        main {
            grid-area: main;
            background: lightcoral;
        }
        aside {
            grid-area: aside;
            background: lightyellow;
        }
        footer {
            grid-area: footer;
            background: lightgrey;
        }
    </style>
</head>
```

```
<body>
    <div class="grid-container">
        <header>Header</header>
        <nav>Navigation</nav>
        <main>Main Content</main>
        <aside>Aside</aside>
        <footer>Footer</footer>
    </div>
</body>
</html>
```

Veja o resultado no navegador, conforme figura abaixo:

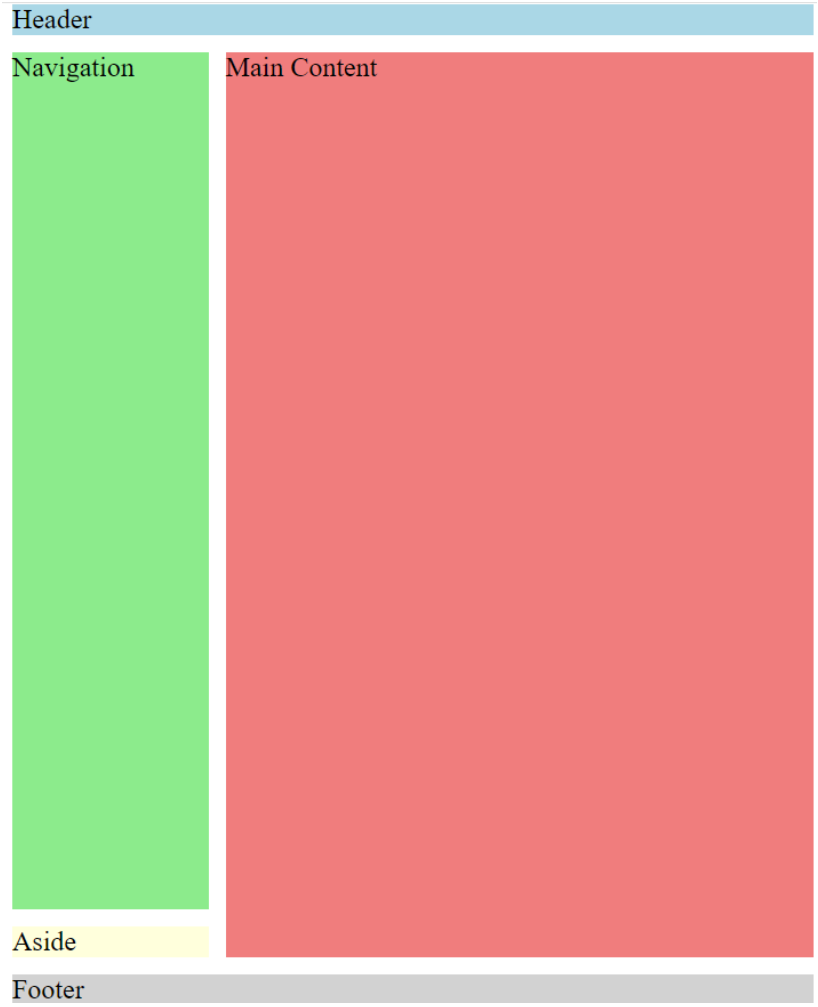

Figura 6 - Exemplo de uso Exemplo de uso da propriedade display: grid

Descrição da imagem: A estrutura da página é dividida em várias áreas, cada uma estilizada com uma cor de fundo diferente. O cabeçalho (header) ocupa toda a largura na parte superior, com um fundo azul claro. A barra de navegação (nav) é colocada na primeira coluna da segunda linha, com um fundo verde claro. O conteúdo principal (main) ocupa a segunda coluna da segunda e terceira linha, com um fundo coral claro. A barra lateral (aside) é colocada na primeira coluna da terceira linha, com um fundo amarelo claro. O rodapé (footer) ocupa toda a largura na parte inferior, com um fundo cinza claro.

◄ 3.1 Posicionamento

Seguir para...

3.1.2 Propriedade position ►