CSS3: Estilizando Páginas Web com Estilo - Turma 2025B

3.1.2 Propriedade position

A propriedade **position** define a posição de um elemento em um documento (**static**, **relative**, **absolute**, **fixed** ou **sticky**). Esta propriedade funciona em conjunto com as subpropriedades **left**, **right**, **top**, **bottom** e **z-index** para determinar a posição final de um elemento em uma página, a saber:

- **left:** Define a distância entre o lado esquerdo do elemento e o lado esquerdo de seu contêiner de posicionamento.
- **right:** Define a distância entre o lado direito do elemento e o lado direito de seu contêiner de posicionamento.
- **top:** Define a distância entre o topo do elemento e o topo de seu contêiner de posicionamento.
- **bottom:** Define a distância entre a parte inferior do elemento e a parte inferior de seu contêiner de posicionamento.
- **z-index:** Controla a pilha de empilhamento dos elementos. Elementos com um valor maior de **z-index** aparecerão na frente de elementos com um valor menor de **z-index**.

Veremos a seguir mais informações sobre os posicionamentos com CSS.

Palavra-chave static

O posicionamento padrão de todo elemento no HTML é estático, ou seja, possui o valor **position: static;**, mesmo que este valor não tenha sido declarado. Todo elemento estático é posicionado alinhado pelo canto superior esquerdo no corpo do documento, não sendo possível alterar sua posição. Este elemento não aceita as propriedades auxiliares **top**, **bottom**, **left**, **right** e **z-index** . Assim, elementos posicionados como **static** são colocados no fluxo normal do documento, seguindo a ordem em que aparecem no código HTML.

Vamos usar um exemplo para mostrar que **position: static;** não afeta a posição de um elemento. No quadro abaixo, colocamos três elementos **div** em um contêiner (elemento pai).

```
<div class="parent-element">
    <div class="sibling-element">Eu sou o outro elemento irmão.</div>
    <div class="main-element">Eu sou o elemento principal.</div>
    <div class="sibling-element">Eu sou o outro elemento irmão.</div>
</div>
```

Em seguida, aplicaremos a propriedade **position: static;** à **div** com a classe **main-element**. Entretanto, é importante notar que ajustes usando as propriedades **left** e **top** não terão efeito neste caso, pois o **position: static;** não suporta tais deslocamentos. Para garantir uma distinção visual clara, também estilizaremos as outras **divs** com cores diferentes, destacando o elemento principal em relação aos demais. Veja o exemplo no quadro abaixo:

```
.main-element {
    position: static;
    left: 10px;  /*Não vai fazer efeito */
    bottom: 10px;
    background-color: yellow;
    padding: 10px;
}
.sibling-element {
    padding: 10px;
    background-color: #f2f2f2;
}
```

Veja o resultado no navegador, conforme figura abaixo:

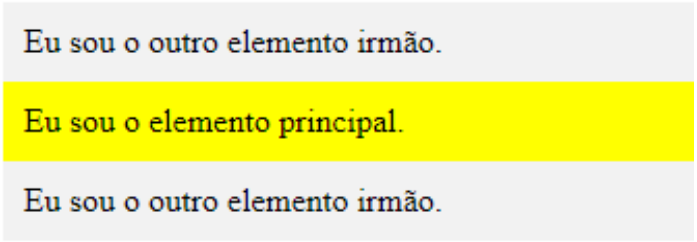

Figura 1 - Exemplo de uso de position: static;

Descrição da imagem: Este código cria um contêiner (parent-element) com três elementos filhos, sendo um deles o elemento principal (main-element) e dois elementos irmãos (sibling-element). O primeiro e o terceiro elementos irmãos têm um fundo cinza claro (#f2f2f2) e um preenchimento de 10 pixels. O elemento principal, que está entre os elementos irmãos, tem um fundo amarelo e um preenchimento de 10 pixels. Embora o CSS inclua propriedades left e bottom para o elemento principal, essas propriedades não têm efeito porque a posição estática (position: static) é o valor padrão e não utiliza essas propriedades para posicionamento. Portanto, os três elementos são exibidos em ordem, um abaixo do outro, dentro do contêiner principal.

Palavra-chave relative

Elementos configurados com **position: relative;** mantêm-se no fluxo normal do documento, assim como aqueles com **position: static**. No entanto, a principal diferença é que, com **position: relative**, as propriedades **left**, **right**, **top**, **bottom**, e **z-index** influenciam efetivamente sua posição. Isso permite que o elemento seja deslocado em relação à sua posição original, conforme definido pelos valores dessas propriedades, sem alterar o layout dos elementos ao redor.

Para ilustrar isso, vamos alterar a propriedade de **position: static;** para **position: relative;** no nosso exemplo existente (aquele com os três elementos div em um contêiner), conforme quadro abaixo:

```
.main-element {
    position: relative;
    left: 10px;
    bottom: 10px;
    background-color: yellow;
    padding: 10px;
}
```

Veja o resultado no navegador, conforme figura abaixo:

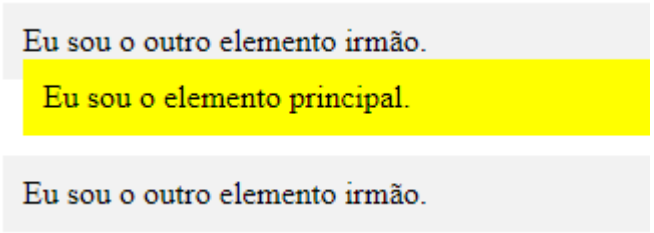

Figura 2 - Exemplo de uso de position: relative;

Descrição da imagem: Este código cria um contêiner (parent-element) com três elementos filhos: dois elementos irmãos (sibling-element) e um elemento principal (main-element). O efeito da propriedade position: relative combinado com left: 10px e bottom: 10px faz com que o elemento principal apareça ligeiramente deslocado para a direita e para cima, mas ainda ocupa o espaço original na página.

Observe que, agora, as propriedades **left** e **bottom** influenciam diretamente a posição do elemento que possui a classe **main-element**. Importante ressaltar que, mesmo com esses deslocamentos, o elemento continua fazendo parte do fluxo normal do documento. O deslocamento é aplicado em relação à sua posição original. Tenha isso em mente à medida que exploramos outros valores de posicionamento.

Palavra-chave absolute

Elementos configurados com **position: absolute;** são posicionados em relação ao seu contêiner de posicionamento (contêiner pai) mais próximo que tenha uma propriedade de posição não estática (**relative**, **absolute**, **fixed** ou **sticky**). Assim, a posição de um elemento posicionado como absoluto é relativa ao primeiro ancestral na árvore de elementos que tem uma posição não estática. Se tal ancestral não existir, o elemento será posicionado em relação ao corpo do documento (**<body>**). Este tipo de posicionamento é usado frequentemente para criar elementos de interface que precisam "flutuar" sobre outros conteúdos, como caixas de diálogo, menus dropdown e tooltips.

Uma vez definido como absoluto, o elemento é removido do fluxo normal do documento, fazendo com que os outros elementos se comportem como se ele não existisse. Isso significa que nenhum espaço é reservado para o elemento no layout da página. As propriedades **left**, **top**, **right** e **bottom** são usadas para determinar a posição exata do elemento em relação a esse contêiner de posicionamento (contêiner pai).

Retornando ao nosso exemplo, vamos modificar a posição do elemento principal para **position: absolute**. Para garantir que este elemento não seja posicionado em relação ao elemento **<html>**, atribuímos ao seu elemento pai o valor **position: relative**. Isso faz com que a posição do elemento principal seja calculada em relação a seu contêiner mais próximo configurado de forma explícita, que é seu pai direto, ao invés de se basear no documento inteiro. Veja como fica o código no quadro abaixo;

```
.main-element {
    position: absolute;
    left: 10px;
    bottom: 10px;
    background-color: yellow;
    padding: 10px;
}
.parent-element {
    position: relative;
    height: 100px;
    padding: 10px;
    background-color: #81adc8;
}
.sibling-element {
    background: #f2f2f2;
    padding: 10px;
    border: 1px solid #81adc8;
}
```

Veja o resultado no navegador, conforme figura abaixo:

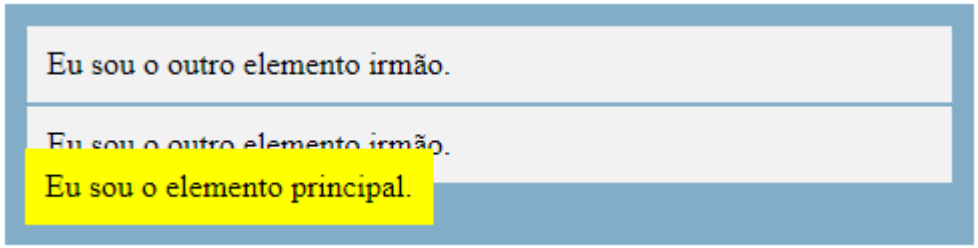

Figura 3 - Exemplo de uso da palavra-chave position: absolute

Descrição da imagem: Este código cria um contêiner (parent-element) com três elementos filhos: dois elementos irmãos (sibling-element) e um elemento principal (main-element). Dentro do contêiner principal, que tem uma altura de 100 pixels, um preenchimento de 10 pixels e um fundo azul claro, há três elementos. Os dois elementos irmãos têm um fundo cinza claro, um preenchimento de 10 pixels e uma borda azul clara. O elemento principal, com um fundo amarelo e um preenchimento de 10 pixels, é posicionado usando position: absolute. Ele aparece 10 pixels à esquerda e 10 pixels acima do fundo do contêiner principal. Devido ao posicionamento absoluto, ele é removido do fluxo normal do documento, não afetando a posição dos elementos irmãos. Como resultado, o elemento principal se

sobrepõe aos outros elementos, ocupando uma posição fixa dentro do contêiner principal, enquanto os elementos irmãos são dispostos normalmente dentro do contêiner.

Note que, com o uso de **position: absolute**, não se reserva um espaço no layout do documento para o elemento. Ele passa a ser posicionado em relação ao seu elemento pai, que é uma característica distintiva do posicionamento absoluto. Esse tipo de posicionamento remove o elemento do fluxo normal do documento, permitindo que ele ocupe uma posição específica sem influenciar a disposição de outros elementos ao redor.

Palavra-chave fixed

Elementos configurados com **position: fixed** compartilham algumas semelhanças com aqueles que têm **position: absolute**, uma vez que eles também são removidos do fluxo normal do documento (o que significa que eles não ocupam espaço no layout e não afetam a posição de outros elementos). Contudo, ao contrário dos elementos com posicionamento absoluto, os fixos somente são posicionados em relação ao elemento **<html>** (e não em relação a um contêiner de posicionamento mais próximo).

Um aspecto crucial a destacar é que os elementos com **position: fixed** não são afetados pela rolagem da página. Eles mantêm sua posição fixa na janela de visualização, permanecendo visíveis no mesmo lugar na tela, independente do deslocamento vertical ou horizontal realizado pelo usuário. Assim, é comum usar **position: fixed** para cabeçalhos de sites ou barras de navegação. Isso garante que os usuários tenham acesso contínuo à navegação, mesmo enquanto rolam para explorar o conteúdo extenso de uma página.

Retornando ao nosso exemplo, vamos modificar a posição do elemento principal para **position: fixed** e definir uma altura da janela da página com 800 pixels (para forçar o uso da barra de rolagem). Veja no quadro abaixo como ficou esta alteração:

```
.main-element {
    position: fixed;
    bottom: 10px;
    left: 10px;
    background-color: yellow;
    padding: 10px;
}
html {
    height: 800px;
}
```

Veja o resultado no navegador, conforme figura abaixo:

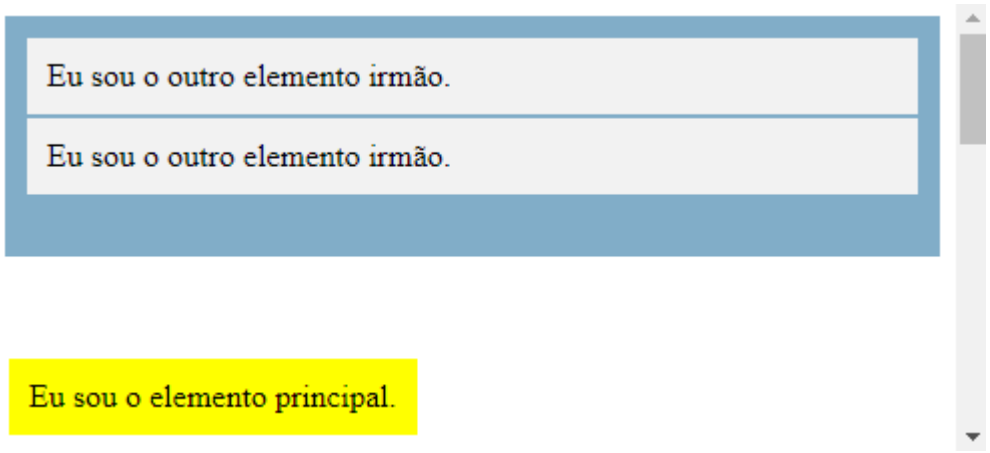

Figura 4 - Exemplo de uso da palavra-chave position: fixed;

Descrição da imagem: Este código cria um contêiner (parent-element) com três elementos filhos: dois elementos irmãos (sibling-element) e um elemento principal (main-element). Dentro do contêiner principal, há três elementos. Os dois elementos irmãos têm um fundo cinza claro, um preenchimento de 10 pixels e uma borda azul clara. O elemento principal tem um fundo amarelo e um preenchimento de 10 pixels. Ele é posicionado fixamente, com 10 pixels de distância da parte inferior e 10 pixels de distância da parte esquerda da janela do navegador. Devido ao position: fixed, este elemento permanece fixo em relação à janela do navegador, mesmo quando a página é rolada.

Neste exemplo, o elemento com **position: fixed** é posicionado em relação ao elemento **<html>**. Ao utilizar a barra de rolagem, observe que o elemento mantém sua posição fixa na tela, independente do movimento de rolagem.

Palavra-chave sticky

A propriedade **position: sticky** é uma funcionalidade híbrida entre o posicionamento **relative** e **fixed**. Quando você configura um elemento com **position: sticky**, ele se comporta como um elemento com **position: relative** (está no fluxo normal do documento) até que seu bloco de contêiner atinja um limite definido na propriedade (como **top**, **bottom**, **left** ou **right**). A partir deste ponto, ele se

"fixa" nesta posição e começa a agir como se tivesse **position: fixed**. Isso permite que o elemento "grude" em uma posição específica dentro da viewport (área visível do conteúdo em um dispositivo de exibição) quando o usuário rola a página.

Retornando ao nosso exemplo, vamos modificar a posição do elemento principal para **position: sticky** e alterar a altura do elemento principal para 800 pixels.

```
.main-element {
    position: sticky;
    top: 10px;
    background-color: yellow;
    padding: 10px;
}
html {
    height: 800px;
}
.parent-element {
    position: relative;
    height: 800px;
    padding: 50px 10px;
    background-color: #81adc8;
}
.sibling-element {
    background: #f2f2f2;
    padding: 10px;
    border: 1px solid #81adc8;
}
```

Veja o resultado no navegador, conforme figura abaixo:

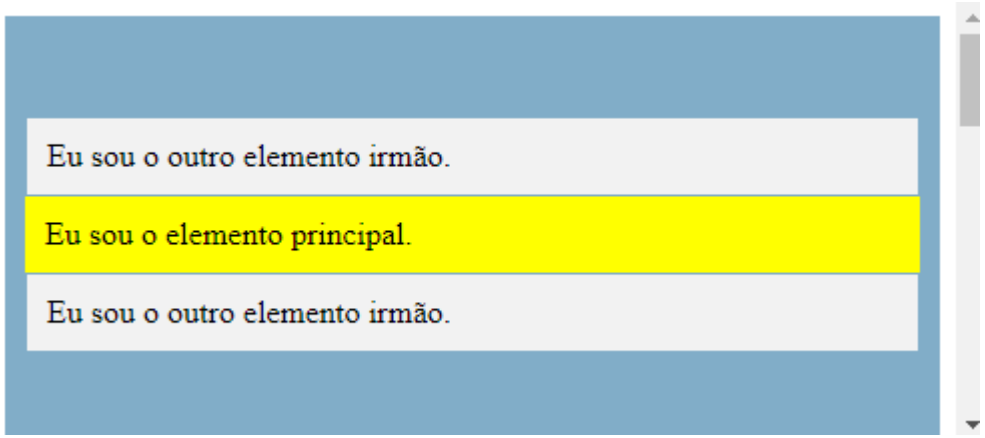

Figura 5 - Exemplo de uso de position: sticky;

Descrição da imagem: Este código cria um contêiner (parent-element) com três elementos filhos: dois elementos irmãos (sibling-element) e um elemento principal (main-element). Dentro do contêiner principal, que tem uma altura de 800 pixels, um preenchimento de 50 pixels no topo e na parte inferior e 10 pixels nas laterais, e um fundo azul claro, há três elementos. Os dois elementos irmãos têm um fundo cinza claro, um preenchimento de 10 pixels e uma borda azul clara. O elemento principal tem um fundo amarelo, um preenchimento de 10 pixels e é posicionado de forma "sticky". Isso significa que ele se comporta como um elemento relativo até que a página seja rolada, e então se torna fixo em 10 pixels do topo da janela do navegador enquanto estiver dentro do contêiner principal.

Role a página para observar o comportamento do elemento em ação. Inicialmente, ele se comporta como um elemento com **position: relative**. No entanto, ao atingir um ponto específico da tela, a 10px do topo, ele começa a agir como um elemento com **position: fixed**. Veja este comportamento na figura abaixo:

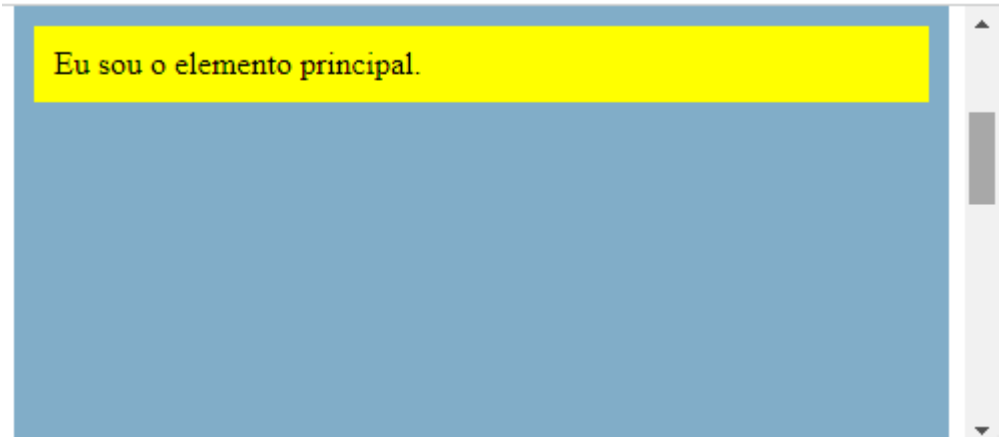

Figura 6 - Exemplo de uso de position: sticky; com rolagem de tela

Descrição da imagem: No navegador, ao rolar a página, o elemento principal ("Eu sou o elemento principal.") permanece visível e fixo a 10 pixels do topo da janela do navegador enquanto estiver dentro do contêiner principal, enquanto os elementos irmãos se deslocam normalmente com a rolagem.

◄ 3.1.1 Propriedade display

Seguir para...

3.2 Margens ►