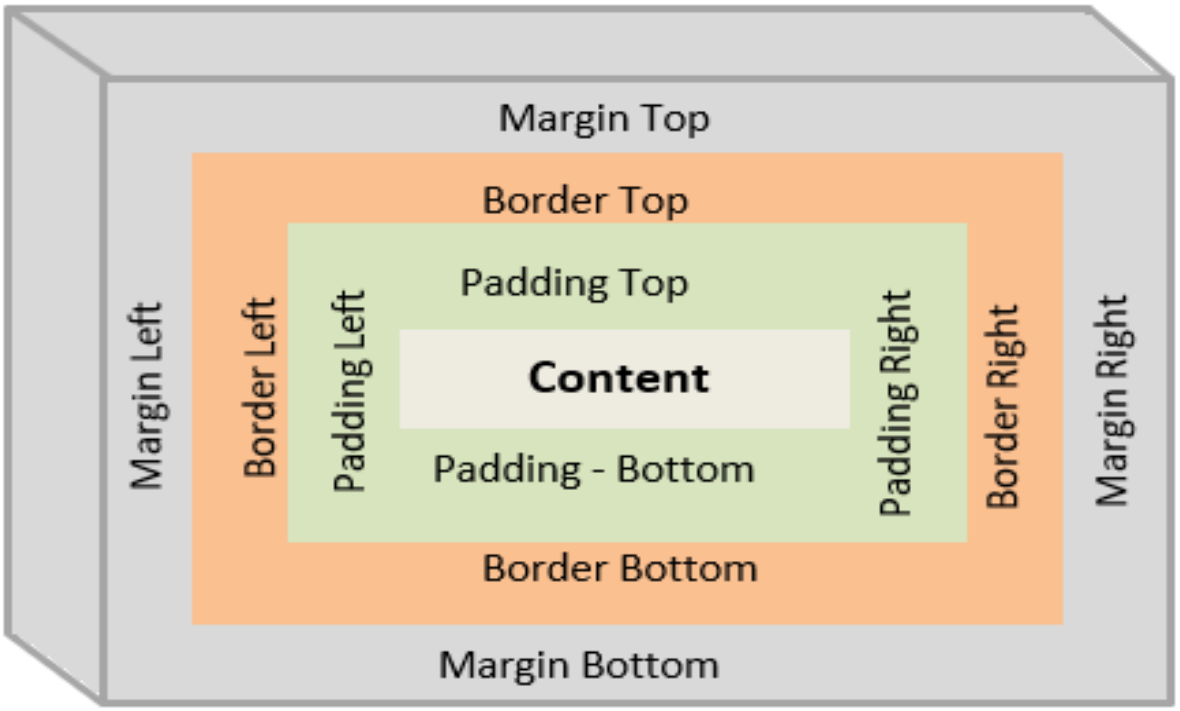


Box Model

Os navegadores carregam elementos HTML com valores de posição padrão. Isso geralmente leva a uma experiência de usuário inesperada e indesejada, limitando as visualizações que podemos criar.

Aprenderemos agora sobre o *modelo de caixa* (Box model), um conceito importante para entender como os elementos são posicionados e exibidos em um site.

Se já usou HTML e CSS, sem saber, viu aspectos do modelo de caixa. Por exemplo, se definimos a cor de fundo de um elemento, podemos notar que a cor foi aplicada não apenas na área diretamente atrás do elemento, mas também na área à direita do elemento.

Além disso, se tiver texto alinhado, saberá que ele está alinhado em relação a algo.

O que é isso?

Todos os elementos de uma página web são interpretados pelo navegador como "vivendo" dentro de uma caixa. Isso é o que se entende por modelo de caixa.

Por exemplo, ao alterar a cor de fundo de um elemento, alteramos a cor de fundo de toda a caixa.

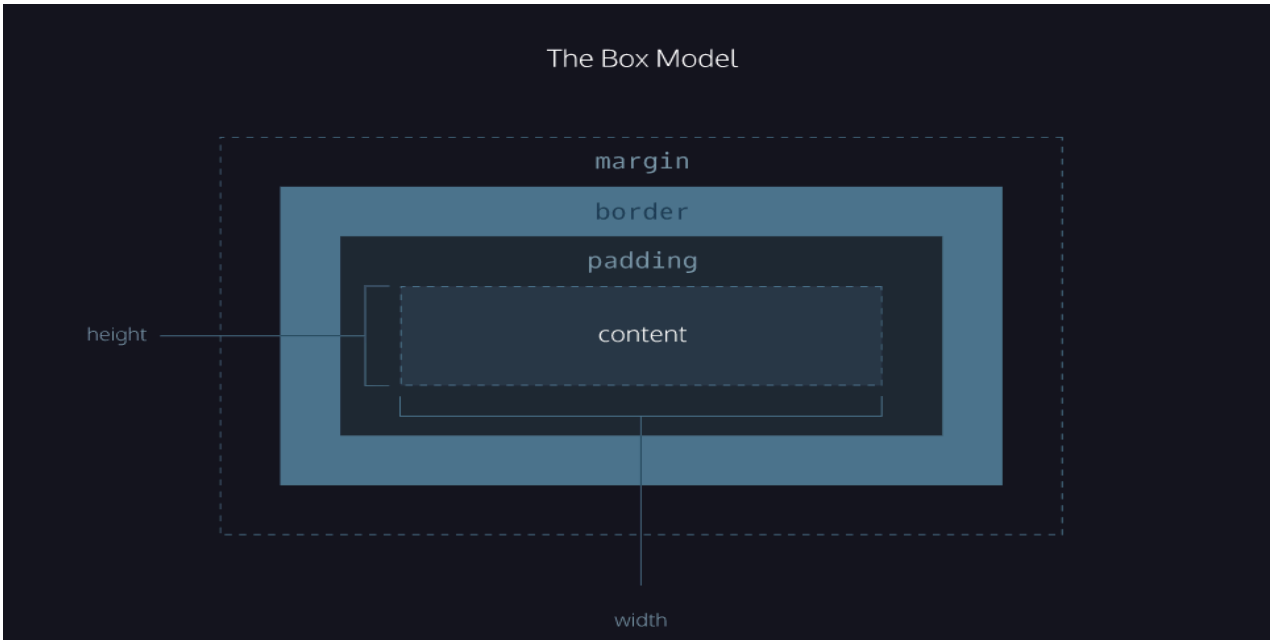
O Box Model

Os navegadores carregam elementos HTML com valores de posição padrão. Isso geralmente leva a uma experiência de usuário inesperada e indesejada, limitando as visualizações que podemos criar.

Aprenderemos sobre a largura(width) e a altura(height) , um conceito importante para entender como os elementos são posicionados e exibidos em um site. Temos também o preenchimento , a borda e a margem do elemento .

As propriedades incluem:

- width e height: A largura e a altura da área de conteúdo.
- padding: A quantidade de espaço entre a área de conteúdo e a borda.
- border: a espessura e o estilo da borda ao redor da área de conteúdo e do preenchimento.
- margin: A quantidade de espaço entre a borda e a borda externa do elemento.



Height e Width

O conteúdo de um elemento tem duas dimensões: uma altura e uma largura. Por padrão, as dimensões de uma caixa HTML são definidas para conter o conteúdo bruto da caixa.

As propriedades height e width podem ser usados para modificar essas dimensões padrão.

```
p {
  height: 80px;
  width: 240px;
}
```

Neste exemplo, os elementos height e width dos parágrafos são definidos como 80 pixels e 240 pixels, respectivamente — o px no código acima significa pixels .

Os pixels permitem definir o tamanho exato da caixa de um elemento (largura e altura). Quando a largura e a altura de um elemento são definidas em pixels, ele terá o mesmo tamanho em todos os dispositivos — um elemento que preenche uma tela de laptop transbordará em uma tela de celular.

Border

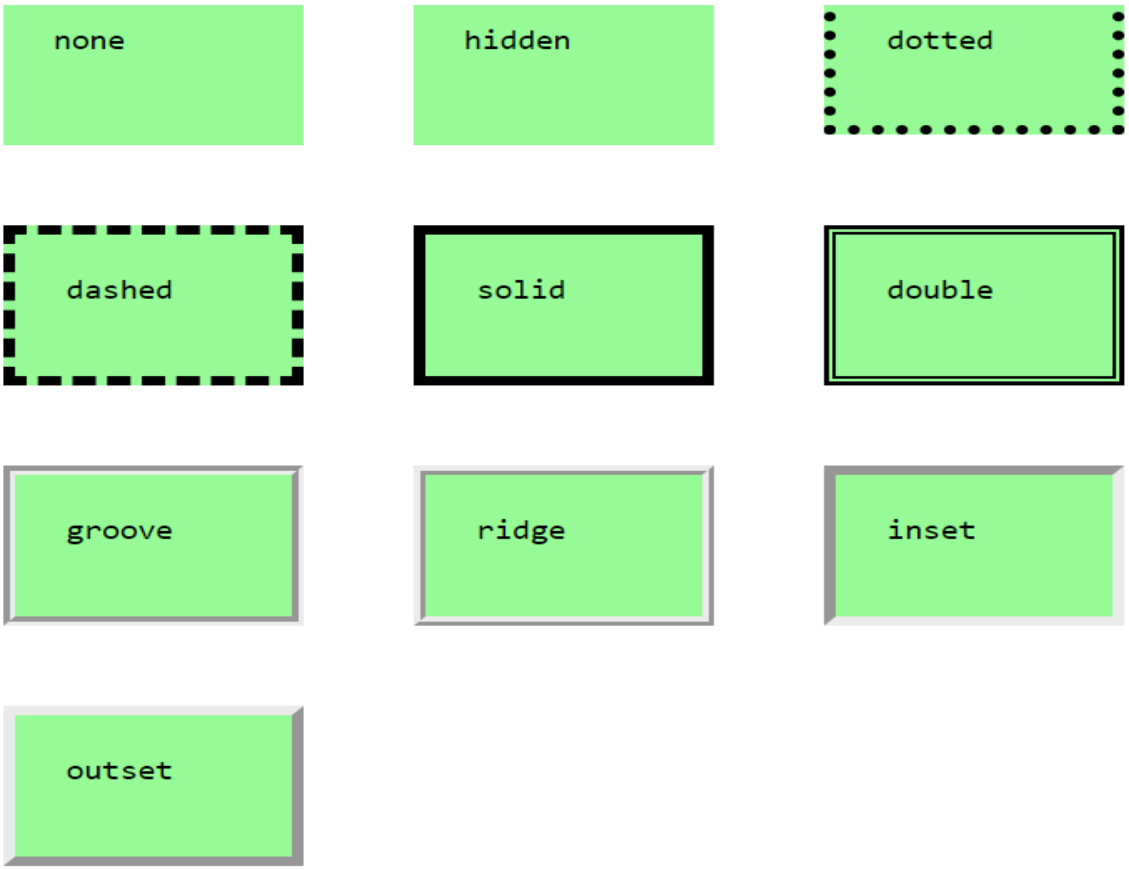
Uma borda é uma linha que envolve um elemento, como uma moldura em torno de uma pintura.

As bordas podem ser definidas as seguintes propriedades: width, style e color onde:

- Width — A espessura da borda. A espessura de uma borda pode ser definida em pixels ou com uma das seguintes palavras-chave: thin, medium, ou thick.
- Style — O desenho da borda. Os navegadores da Web podem renderizar qualquer um dos 10 estilos diferentes .

Alguns desses estilos incluem:

- none: Como a palavra- chave hidden, não exibe bordas. A menos que a background-image seja definido, o valor calculado do
- hidden: Como a palavra-chave none, não exibe bordas. A menos que a background-image seja definido.
- dotted: Exibe uma série de pontos arredondados. O espaçamento dos pontos não é definido pela especificação e é específico da implementação.
- dashed: Exibe uma série de traços curtos de extremidade quadrada ou segmentos de linha. O tamanho e comprimento exatos dos segmentos não são definidos pela especificação e são específicos da implementação.
- solid: Exibe uma única linha reta e sólida.
- double: Exibe duas linhas retas que somam o tamanho do pixel definido por border-width.
- groove: Exibe uma borda com aparência esculpida. É o contrário de ridge.
- ridge: Exibe uma borda com aparência de extrusão. É o contrário de groove.
- inset: Exibe uma borda que faz o elemento parecer incorporado. É o contrário de outset
- outset: Exibe uma borda que faz com que o elemento apareça em relevo. É o contrário de inset.



- Color — A cor da borda. Os navegadores da Web podem renderizar cores usando alguns formatos diferentes., incluindo 140 palavras-chave de cores incorporadas .

Exemplo:

```
p {
  border: 3px solid coral;
}
```

No exemplo acima, a borda tem uma largura de 3 pixels, um estilo de solid e uma cor de coral. Todas as três propriedades são definidas em uma linha de código.

A borda padrão é medium none color, onde color é a cor atual do elemento.

Se width, style, ou color não estiverem configurados no arquivo CSS, o navegador da web designará o valor padrão para essa propriedade.

```
p.content-header {
  height: 80px;
  width: 240px;
  border: solid coral;
}
```

Neste exemplo, o estilo da borda é definido como solid e a cor é definida como coral. A largura não está definida, portanto, o padrão é medium.

Border Radius

Desde que revelamos as bordas das caixas, notamos que as bordas destacam a verdadeira forma da caixa de um elemento: quadrado.

Graças ao CSS, uma borda não precisa ser quadrada.

Podemos modificar os cantos da caixa de borda de um elemento com a propriedade border-radius.

```
div.container {
  border: 3px solid blue;
  border-radius: 5px;
}
```

O código no exemplo acima irá definir todos os quatro cantos da borda para um raio de 5 pixels (ou seja, a mesma curvatura que um círculo com um raio de 5 pixels teria).

Você pode criar uma borda que seja um círculo perfeito criando primeiro um elemento com a mesma largura e altura e, em seguida, definindo o raio igual à metade da largura da caixa, que é 50%.

```
div.container {
    height: 60px;
    width: 60px;
    border: 3px solid blue;
    border-radius: 50%;
}
```

O código no exemplo acima cria um círculo perfeito.

Padding

O espaço entre o conteúdo de uma caixa e as bordas de uma caixa é conhecido como preenchimento . O preenchimento é como o espaço entre uma imagem e o quadro que a cerca. Em CSS, você pode modificar este espaço com a propriedade padding.

```
p.content-header {
  border: 3px solid coral;
  padding: 10px;
}
```

O código neste exemplo coloca 10 pixels de espaço entre o conteúdo do parágrafo (o texto) e as bordas, nos quatro lados.

A propriedade padding é frequentemente usada para expandir a cor do plano de fundo e fazer com que o conteúdo pareça menos apertado.

Se quiser ser mais específico sobre a quantidade de preenchimento em cada lado do conteúdo de uma caixa, você pode usar as seguintes propriedades:

- padding-top
- padding-right
- padding-bottom
- padding-left

Cada propriedade afeta o preenchimento em apenas um lado do conteúdo da caixa, oferecendo mais flexibilidade na personalização.

```
p.content-header {
  border: 3px solid fuchsia;
  padding-bottom: 10px;
}
```

No exemplo acima, apenas a parte inferior do conteúdo do parágrafo terá padding de 10 pixels.

Padding Shorthand

Outra implementação da propriedade padding permite especificar exatamente quanto preenchimento deve haver em cada lado do conteúdo em uma única declaração. Uma declaração que usa várias propriedades como valores é conhecida como propriedade abreviada.

A abreviação de preenchimento permite especificar todas as propriedades padding como valores em uma única linha:

- padding-top
- padding-right
- padding-bottom
- padding-left

Podemos especificar essas propriedades de algumas maneiras diferentes:

4 Valores

```
p.content-header {
  padding: 6px 11px 4px 9px;
}
```

No exemplo acima, os quatro valores 6px 11px 4px 9px correspondem à quantidade de preenchimento de cada lado, em rotação no sentido horário. Em ordem, ele especifica o valor de preenchimento superior (6px), o valor de preenchimento direito (11px), o valor de preenchimento inferior (4px) e o valor de preenchimento esquerdo (9px) do conteúdo.

3 Valores

```
p.content-header {
  padding: 5px 10px  20px;
}
```

Se os lados esquerdo e direito do conteúdo puderem ser iguais, a propriedade abreviada de preenchimento permite que 3 valores sejam especificados. O primeiro valor define o valor de preenchimento superior (5px), o segundo valor define os valores de preenchimento esquerdo e direito (10px) e o terceiro valor define o valor de preenchimento inferior (20px).

2 Valores

```
p.content-header {
  padding: 5px 10px;
}
```

E, finalmente, se os lados superior e inferior puderem ser iguais e os lados esquerdo e direito puderem ser iguais, você poderá especificar 2 valores. O primeiro valor define os valores padding-top e padding-bottom (5px), e o segundo valor define os valores padding-left e padding-right (10px).

Margin

O quarto e último componente do modelo de caixa é a margem . Margem refere-se ao espaço diretamente fora da caixa. A propriedade margin é usada para especificar o tamanho desse espaço.

```
p {
  border: 1px solid aquamarine;
  margin: 20px;
}
```

O código no exemplo acima colocará 20 pixels de espaço do lado de fora da caixa do parágrafo em todos os quatro lados. Isso significa que outros elementos HTML na página não podem chegar a 20 pixels da borda do parágrafo.

Se quisermos ser ainda mais específicos sobre a quantidade de margem em cada lado de uma caixa, podemos usar as seguintes propriedades:

- margin-top
- margin-right
- margin-bottom
- margin-left

Cada propriedade afeta a margem em apenas um lado da caixa, proporcionando mais flexibilidade na personalização.

```
p {
  border: 3px solid DarkSlateGrey;
  margin-right: 15px;
}
```

No exemplo acima, apenas o lado direito da caixa do parágrafo terá uma margem de 15 pixels. É comum ver valores de margem usados para um lado específico de um elemento.

Margin Shorthand

E se não quisermos margens iguais nos quatro lados da caixa e não tiver tempo para separar as propriedades de cada lado?

A margem também pode ser escrita como uma propriedade abreviada. A sintaxe abreviada para margens é a mesma do preenchimento.

Semelhante à abreviação de preenchimento, a abreviação de margem permite especificar todas as propriedades margin como valores em uma única linha:

- margin-top
- margin-right
- margin-bottom
- margin-left

Você pode especificar essas propriedades de algumas maneiras diferentes:

4 Valores

```
p {
  margin: 6px 10px 5px 12px;
}
```

No exemplo acima, os quatro valores 6px 10px 5px 12px correspondem à espessura da margem de cada lado, no sentido horário.

Em ordem, ele especifica o valor da margem superior (6px), o valor da margem direita (10px), o valor da margem inferior (5px) e o valor da margem esquerda (12px) do conteúdo.

3 Valores

```
p {
  margin: 5px 12px 4px;
}
```

Se os lados esquerdo e direito do conteúdo puderem ser iguais, a propriedade abreviada de margem permite que 3 valores sejam especificados.

O primeiro valor define o valor da margem superior (5px), o segundo valor define os valores da margem esquerda e da margem direita (12px) e o terceiro valor define o valor da margem inferior (4px).

2 Valores

```
p {
  margin: 20px 10px;
}
```

E, finalmente, se os lados superior e inferior puderem ser iguais e os lados esquerdo e direito puderem ser iguais, você poderá especificar 2 valores.

O primeiro valor define os valores margin-top e margin-bottom (20px), e o segundo valor define os valores margin-left e margin-right (10px).

Auto

A propriedade margin também permite que você centralize o conteúdo.

No entanto, você deve seguir alguns requisitos de sintaxe.

Exemplo:

```
div.headline {
  width: 400px;
  margin: 0 auto;
}
```

No exemplo acima, margin: 0 auto; centralizará as divs em seus elementos contidos. O 0 define as margens superior e inferior para 0 pixels.

O valor auto instrui o navegador a ajustar as margens esquerda e direita até que o elemento seja centralizado no elemento que o contém.

Para centralizar um elemento, uma largura deve ser definida para esse elemento. Caso contrário, a largura do div será definida automaticamente para a largura total do elemento que o contém, como o <body>, por exemplo.

Não é possível centralizar um elemento que ocupe toda a largura da página, pois a largura da página pode mudar devido à exibição e/ou tamanho da janela do navegador.

No exemplo acima, a largura do div é definida como 400 pixels, que é menor que a largura da maioria das telas. Isso fará com que o div seja centralizado em um elemento que tenha mais de 400 pixels de largura.

Margin Collapse

Como vimos, padding é espaço adicionado dentro da borda de um elemento, enquanto margin é espaço adicionado fora da borda de um elemento.

Uma diferença adicional é que as margens superior e inferior, também chamadas de margens verticais, são recolhidas, enquanto o preenchimento superior e inferior não.

As margens horizontais (esquerda e direita), como preenchimento, são sempre exibidas e somadas. Por exemplo, se dois divs com ids #div-one e #div-two, estiverem próximos um do outro, eles estarão tão distantes quanto a soma de suas margens adjacentes.

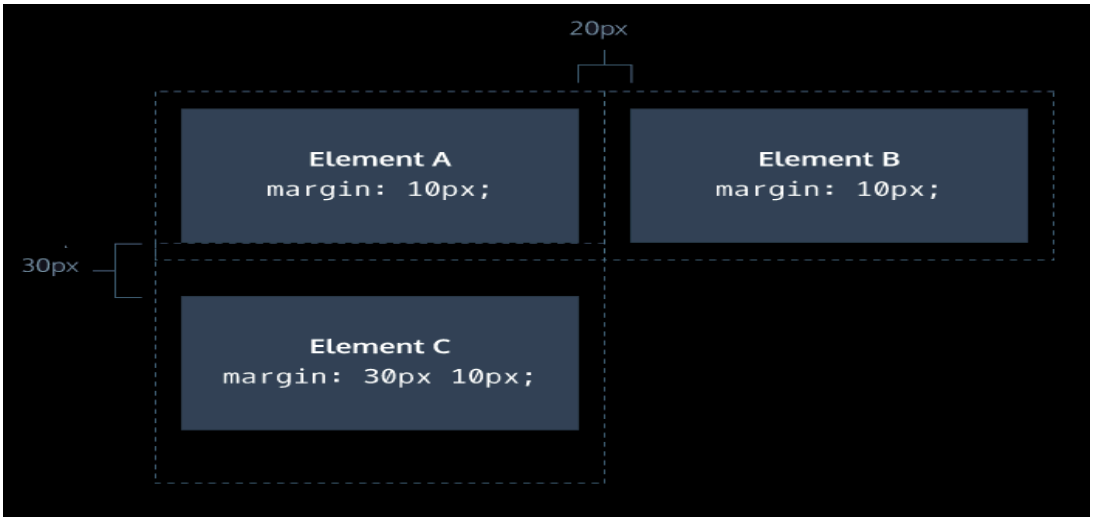
```
#img-one {
  margin-right: 20px;
}

#img-two {
  margin-left: 20px;
}
```

Neste exemplo, o espaço entre as bordas #img-one e #img-two é de 40 pixels. A margem direita de #img-one(20px) e a margem esquerda de #img-two(20px) somam para fazer uma margem total de 40 pixels.

Ao contrário das margens horizontais, as margens verticais não adicionam.

Em vez disso, a maior das duas margens verticais define a distância entre os elementos adjacentes.

```
#img-one {
  margin-bottom: 30px;
}
 #img-two {
  margin-top: 20px;
}
```



Neste exemplo, a margem vertical entre os elementos #img-one e #img-two é de 30 pixels. Embora a soma das margens seja de 50 pixels, a margem é recolhida, de modo que o espaçamento depende apenas da #img-one margem inferior.

Pode ser útil pensar no colapso das margens verticais como uma pessoa baixa tentando empurrar uma pessoa mais alta. A pessoa alta tem braços mais longos e pode empurrar facilmente a pessoa baixa, enquanto a pessoa com braços curtos não consegue alcançar a pessoa com braços longos.

Minimum and Maximum Height and Width

Como uma página da Web pode ser visualizada por meio de exibições de diferentes tamanhos de tela, o conteúdo da página da Web pode sofrer com essas alterações de tamanho. Para evitar esse problema, o CSS oferece duas propriedades que podem limitar o tamanho da caixa de um elemento: estreita ou larga:

- min-width — esta propriedade garante uma largura mínima da caixa de um elemento.
- max-width — esta propriedade garante a largura máxima da caixa de um elemento.

```
p {
  min-width: 300px;
  max-width: 600px;
}
```

No exemplo acima, a largura de todos os parágrafos não diminuirá abaixo de 300 pixels, nem a largura excederá 600 pixels.

O conteúdo, como o texto, pode se tornar difícil de ler quando uma janela do navegador é reduzida ou expandida. Essas duas propriedades garantem que o conteúdo seja legível, limitando as larguras mínima e máxima de um elemento.

Você também pode limitar a altura mínima e máxima de um elemento:

- min-height— esta propriedade garante uma altura mínima para a caixa de um elemento.
- max-height— esta propriedade garante uma altura máxima da caixa de um elemento.

```
p {
  min-height: 150px;
  max-height: 300px;
}
```

No exemplo acima, a altura de todos os parágrafos não diminuirá abaixo de 150 pixels e a altura não excederá 300 pixels.

Overflow

Todos os componentes do modelo de caixa compreendem o tamanho de um elemento.

Por exemplo, uma imagem com as seguintes dimensões tem 364 pixels de largura e 244 pixels de altura.

- ✔ 300 pixels de largura
- ✔ 200 pixels de altura
- ✔ 10 pixels de preenchimento à esquerda e à direita
- ✔ 10 pixels de preenchimento na parte superior e inferior
- ✔ 2 pixels de borda à esquerda e à direita
- ✔ 2 pixels de borda na parte superior e inferior
- ✔ Margem de 20 pixels à esquerda e à direita
- ✔ Margem de 10 pixels na parte superior e inferior

As dimensões totais (364 px por 244 px) são calculadas somando todas as dimensões verticais e todas as dimensões horizontais.

Às vezes, esses componentes resultam em um elemento maior que a área de contenção do pai.

Como podemos garantir que podemos visualizar todo um elemento que é maior que a área de contenção de seu pai?

A propriedade overflow controla o que acontece com o conteúdo que transborda ou transborda fora de sua caixa. Os valores mais usados são:

- *hidden* — quando definido com esse valor, qualquer conteúdo que estoure ficará oculto da visualização.
- *scroll*— quando definido para este valor, uma barra de rolagem será adicionada à caixa do elemento para que o restante do conteúdo possa ser visualizado rolando.
- *visible*— quando definido para esse valor, o conteúdo de estouro será exibido fora do elemento que o contém. Observe que este é o valor padrão.

```
p {
  overflow: scroll;
}
```

No exemplo acima, se algum conteúdo do parágrafo transbordar (talvez um usuário redimensione a janela do navegador), uma barra de rolagem aparecerá para que os usuários possam visualizar o restante do conteúdo.

A propriedade overflow é definida em um elemento pai para instruir um navegador da Web sobre como renderizar elementos filho.

Por exemplo, se a propriedade de estouro de um div estiver definida como scroll, todos os filhos desse div exibirão conteúdo transbordante com uma barra de rolagem.

Redefinindo padrões

Todos os principais navegadores da Web têm uma folha de estilo padrão que usam na ausência de uma folha de estilo externa. Essas folhas de estilo padrão são conhecidas como folhas de estilo do agente do usuário. Nesse caso, o termo agente do usuário é um termo técnico para o navegador.

As folhas de estilo do agente do usuário geralmente têm regras CSS padrão que definem valores padrão para preenchimento e margem. Isso afeta como o navegador exibe os elementos HTML, o que pode dificultar o design ou o estilo de uma página da Web por um desenvolvedor.

Muitos desenvolvedores optam por redefinir esses valores padrão para que possam realmente trabalhar com um quadro limpo.

```
* {
  margin: 0;
  padding: 0;
}
```

O código no exemplo acima redefine os valores padrão de margem e preenchimento de todos os elementos HTML. Geralmente é a primeira regra CSS em uma folha de estilo externa.

Observe que ambas as propriedades estão definidas como 0. Quando essas propriedades são definidas como 0, elas não exigem uma unidade de medida.

Visibility

Os elementos podem ser ocultados da visualização com a propriedade visibility.

A propriedade visibility pode ser definida para um dos seguintes valores:

- hidden— oculta um elemento.
- visible— exibe um elemento.
- collapse— recolhe um elemento.

Exemplo:

```
<ul>
  <li>Explore</li>
  <li>Connect</li>
  <li class="future">Donate</li>
</ul>

<style>

.future {
  visibility: hidden;
}

</style>
```

No exemplo acima, o item da lista com uma classe de future será ocultado da visualização no navegador.

No entanto, lembre-se de que os usuários ainda podem visualizar o conteúdo do item da lista (por exemplo, Donate) visualizando o código-fonte em seu navegador. Além disso, a página da Web apenas ocultará o conteúdo do elemento. Ele ainda deixará um espaço vazio onde o elemento deve ser exibido.

Saiba mais:

https://www.w3schools.com/css/css_boxmodel.asp

https://www.w3schools.com/css/css_icons.asp

https://developer.mozilla.org/en-US/docs/Web/CSS/border-style#values

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model

https://www.w3schools.com/css/css_border.asp

https://www.w3schools.com/css/css_border_color.asp

https://www.youtube.com/watch?v=3ZFYXkzXhqE

https://www.youtube.com/watch?v=rXF1okX0v9E

https://youtu.be/n2sp9tgEdag