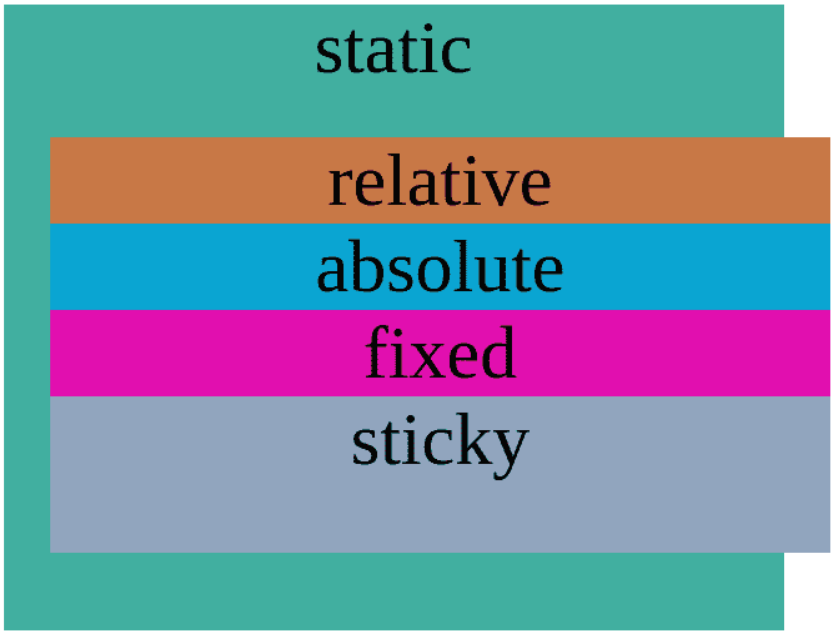


EXIBIÇÃO E POSICIONAMENTO

Fluxo de HTML

Um navegador renderiza os elementos de um documento HTML que não possui CSS da esquerda para a direita, de cima para baixo, na mesma ordem em que existem no documento. Isso é chamado de *fluxo* de elementos em HTML.

Além das propriedades que fornece para estilizar elementos HTML, o CSS inclui propriedades que alteram como um navegador *posiciona* os elementos. Essas propriedades especificam onde um elemento está localizado em uma página, se o elemento pode compartilhar linhas com outros elementos e outros atributos relacionados.

Temos cinco propriedades para ajustar a posição dos elementos HTML no navegador:

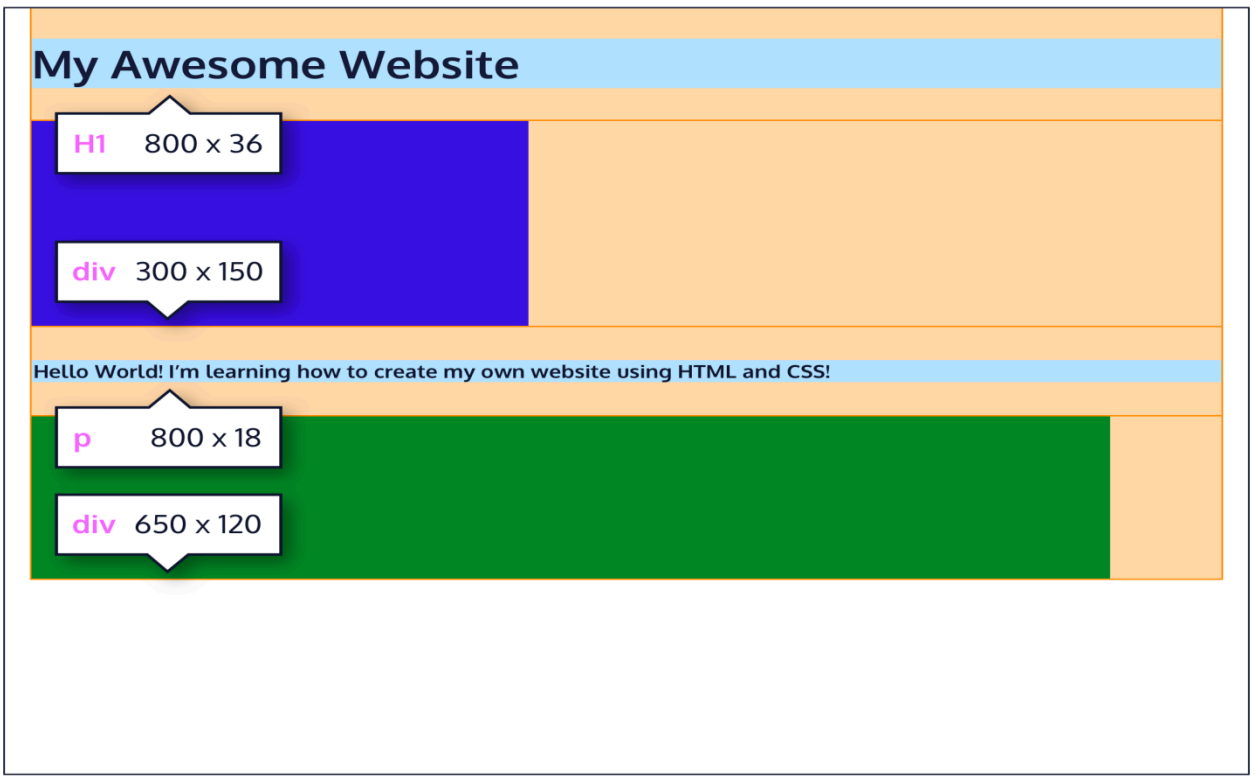
- position
- display
- z-index

- float
- clear

Cada uma dessas propriedades nos permitirá posicionar e visualizar elementos em uma página da web. Eles podem ser usados em conjunto com quaisquer outras propriedades de estilo que você conheça.

Position

Dê uma olhada nos elementos *de nível de bloco* na imagem abaixo:



Elementos de nível de bloco como essas caixas criam um *bloco* com a largura total de seus elementos pai e impedem que outros elementos apareçam no mesmo espaço horizontal.

Observe que os elementos de nível de bloco na imagem acima ocupam sua própria linha de espaço e, portanto, não se sobrepõem. No navegador à direita, você pode ver elementos de nível de bloco também aparecem consistentemente no lado esquerdo do navegador.

Esta é a posição padrão para elementos de nível de bloco.

A posição padrão de um elemento pode ser alterada definindo sua propriedade position. A propriedade position pode ter um dos cinco valores:

- static- o valor padrão (não precisa ser especificado)
- relative
- absolute
- fixed
- sticky

É importante entender que se você usa a posição padrão de um elemento HTML, não precisa definir sua propriedade position.

Position: Relative

Uma maneira de modificar a posição padrão de um elemento é definindo sua propriedade `position` como `relative`.

Esse valor permite posicionar um elemento em *relação* à sua posição estática padrão na página da web.

```
.green-box {
  background-color: green;
  position: relative;
}
```

Embora o código no exemplo acima instrua o navegador a esperar um posicionamento relativo do elemento `.green-box`, ele não especifica onde o elemento `.green-box` deve ser posicionado na página. Isso é feito acompanhando a declaração `position` com uma ou mais das seguintes *propriedades de deslocamento* que afastarão o elemento de sua posição estática padrão:

- `top`- move o elemento para baixo a partir do topo.
- `bottom`- move o elemento de baixo para cima.
- `left`- afasta o elemento do lado esquerdo (para a direita).

- right- afasta o elemento do lado direito (para a esquerda).

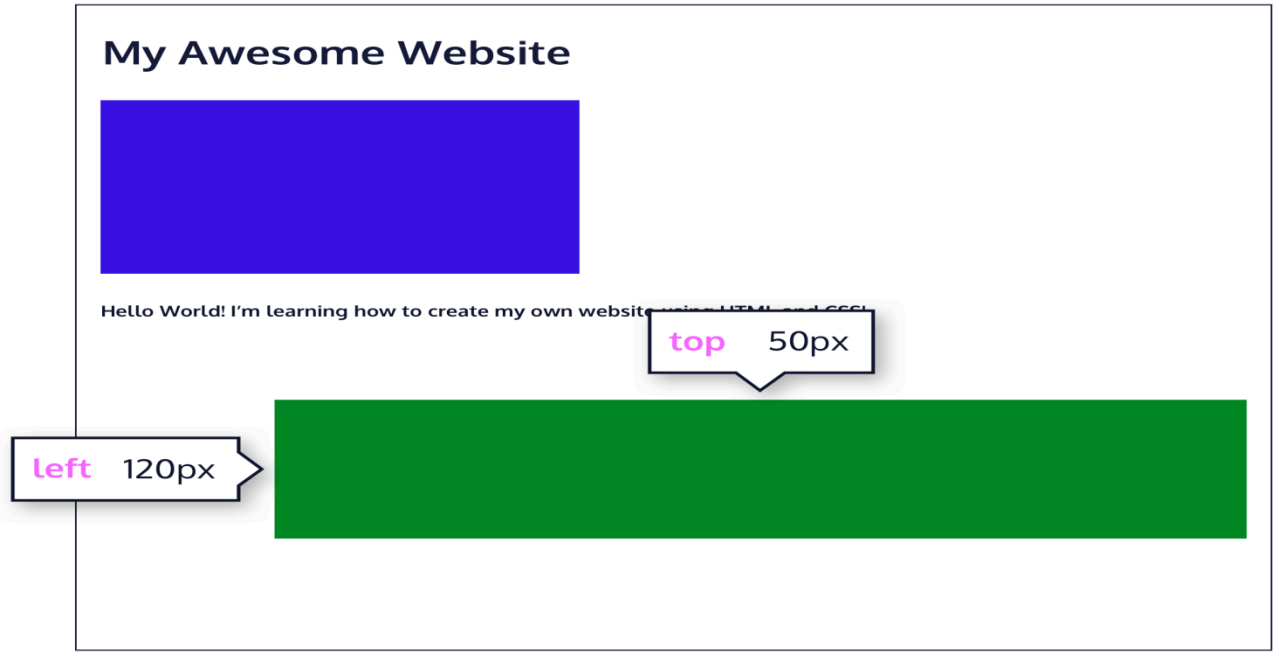
Você pode especificar valores em pixels, ems ou porcentagens, entre outros, para dizer exatamente até onde você precisa que o elemento se mova.

Também é importante observar que as propriedades de deslocamento não funcionarão se a propriedade position do elemento for o padrão static.

```
.green-box {
  background-color: green;
  position: relative;
  top: 50px;
  left: 120px;
}
```

No exemplo acima, o elemento de classe green-box será movido 50 pixels para baixo e 120 pixels para a direita, de sua posição estática padrão.

A imagem abaixo mostra a nova posição da caixa.



O deslocamento do elemento relativo não afetará o posicionamento de outros elementos.

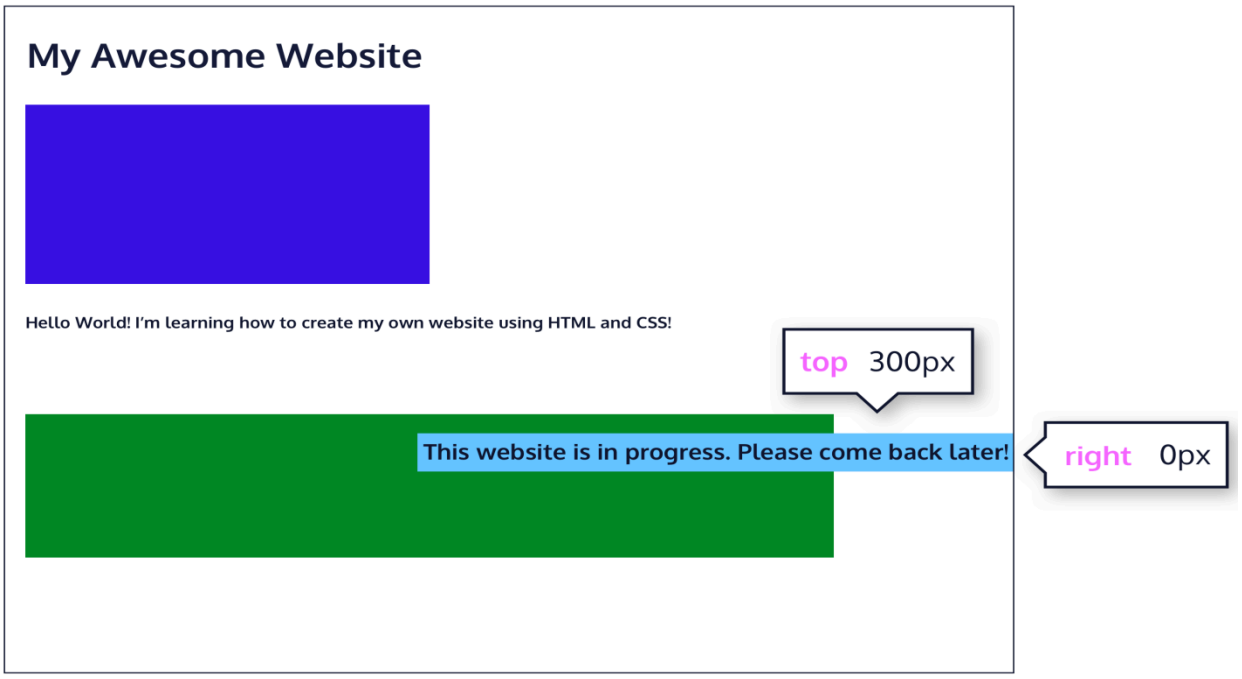
Position: Absolute

Outra maneira de modificar a posição de um elemento é definindo sua posição como `absolute`.

Quando a posição de um elemento é definida como `absolute`, todos os outros elementos na página ignorarão o elemento e agirão como se ele não estivesse presente na página.

O elemento será posicionado em relação ao elemento pai posicionado mais próximo, enquanto as propriedades de deslocamento podem ser usadas para determinar a posição final a partir daí.

Dê uma olhada na imagem abaixo:



O “Este site está em andamento. Por favor, volte mais tarde!" é deslocado de sua posição estática no canto superior esquerdo de seu contêiner pai. Ele possui declarações de propriedade de deslocamento de `top: 300px;` e `right: 0px;`, posicionando-o 300 pixels para baixo e 0 pixels do lado direito da página.

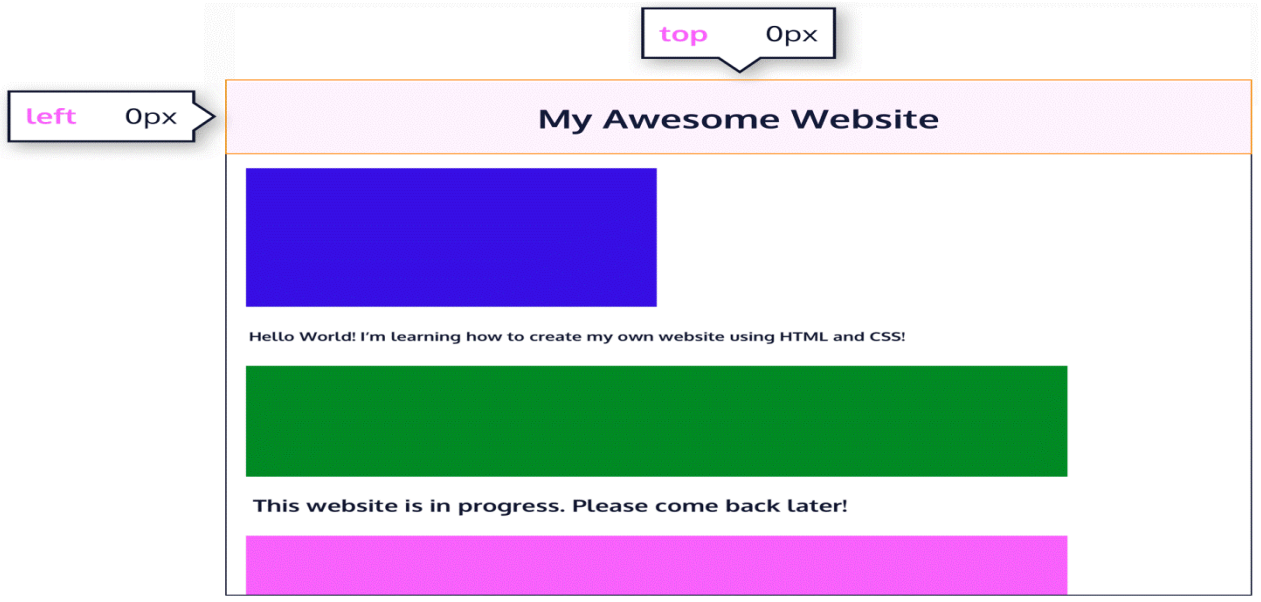
Position: Fixed

Quando a posição de um elemento é definida como absolute, o elemento rolará com o restante do documento quando um usuário rolar.

Podemos fixar um elemento em uma posição específica na página (independentemente da rolagem do usuário) definindo sua posição com o fixed, e acompanhando-o com as propriedades de deslocamento familiares , top, bottom, lefte right.

```
.title {
  position: fixed;
  top: 0px;
  left: 0px;
}
```

No exemplo acima, o elemento .title permanecerá fixo em sua posição independentemente de onde o usuário rolar na página, como na imagem abaixo:



Essa técnica é frequentemente usada para barras de navegação em uma página da web.

Position: Sticky

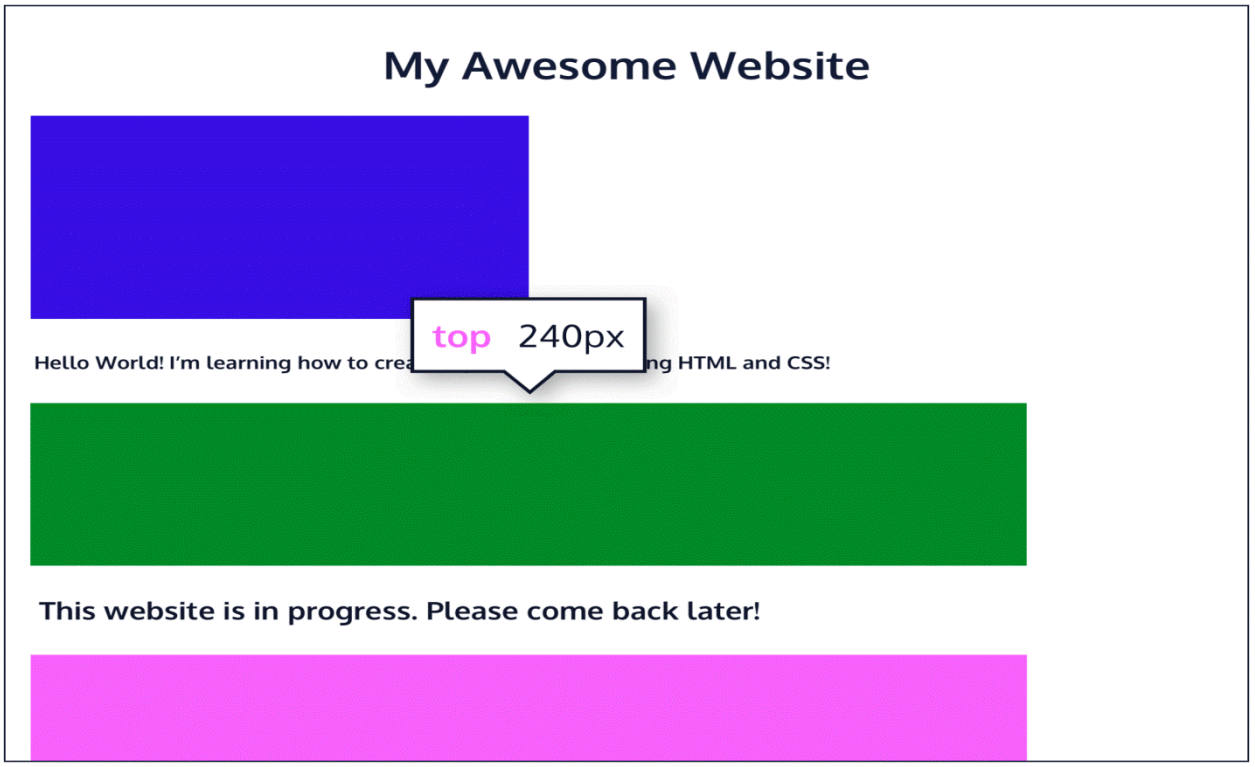
Como os elementos posicionados static e relative permanecem no fluxo normal do documento, quando um usuário rola a página (ou elemento pai), esses elementos também rolam.

E como os elementos posicionados fixed e absolute são removidos do fluxo de documentos, quando um usuário rola, esses elementos permanecerão em sua posição de deslocamento especificada.

O valor sticky é outro valor de posição que mantém um elemento no fluxo do documento à medida que o usuário rola, mas permanece em *uma* posição especificada à medida que a página é rolada ainda mais. Isso é feito usando o valor sticky junto com as propriedades de deslocamento familiares, bem como uma nova.

```
.box-bottom {
  background-color: darkgreen;
  position: sticky;
  top: 240px;
}
```

No exemplo acima, o .box-bottom permanecerá em sua posição relativa e rolará como de costume. Quando atingir 240 pixels a partir do topo, ele permanecerá nessa posição até atingir a parte inferior do contêiner pai, onde será "descolado" e voltará ao fluxo do documento.



Z-Index

Quando as caixas em uma página da Web têm uma combinação de posições diferentes, as caixas (e, portanto, seu conteúdo) podem se sobrepor, dificultando a leitura ou o consumo do conteúdo.

```
.blue-box {
  background-color: blue;
}

.green-box {
  background-color: green;
  position: relative;
  top: -170px;
  left: 170px;
}
```

No exemplo acima, o elemento .green-box se sobrepõe à parte superior do elemento .blue-box.

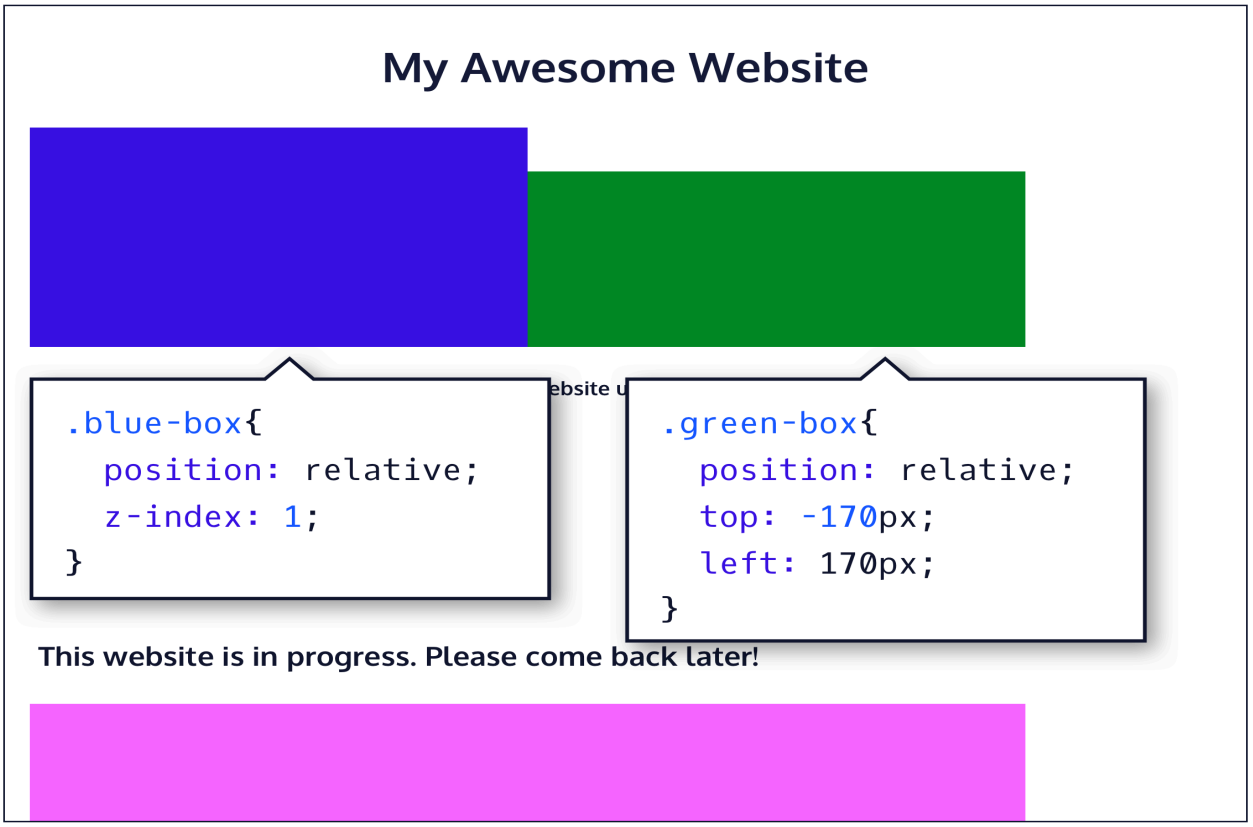
A propriedade z-index controla o quanto um elemento deve aparecer para trás ou para frente na página da Web quando os elementos se sobrepõem. Isso pode ser pensado como a *profundidade* dos elementos, com elementos mais profundos aparecendo atrás de elementos mais rasos.

A propriedade z-index aceita valores inteiros. Dependendo de seus valores, os inteiros instruem o navegador sobre a ordem em que os elementos devem ser colocados em camadas na página da web.

```
.blue-box {
  background-color: blue;
  position: relative;
  z-index: 1;
}

.green-box {
  background-color: green;
  position: relative;
  top: -170px;
  left: 170px;
}
```

No exemplo acima, definimos a posição .blue-box como relative e o z-index como 1. Mudamos a posição para relative, porque a propriedade z-index *não* funciona em elementos estáticos. O z-index de 1 move o elemento .blue-box para frente, porque o z-index valor não foi especificado explicitamente para o elemento .green-box, o que significa que ele tem um valor z-index padrão de 0. Dê uma olhada na imagem de exemplo abaixo:



Float

Até agora, você aprendeu como especificar a posição exata de um elemento usando propriedades de deslocamento. Se você estiver simplesmente interessado em mover um elemento o mais para a esquerda ou para a direita possível no contêiner, você pode usar a propriedade `float`.

A propriedade `float` é comumente usada para envolver o texto em torno de uma imagem. Observe, no entanto, que mover elementos para a esquerda ou para a direita para fins de layout é mais adequado para ferramentas como grade CSS e flexbox, sobre as quais você aprenderá mais tarde.

A propriedade `float` geralmente é definida usando um dos valores abaixo:

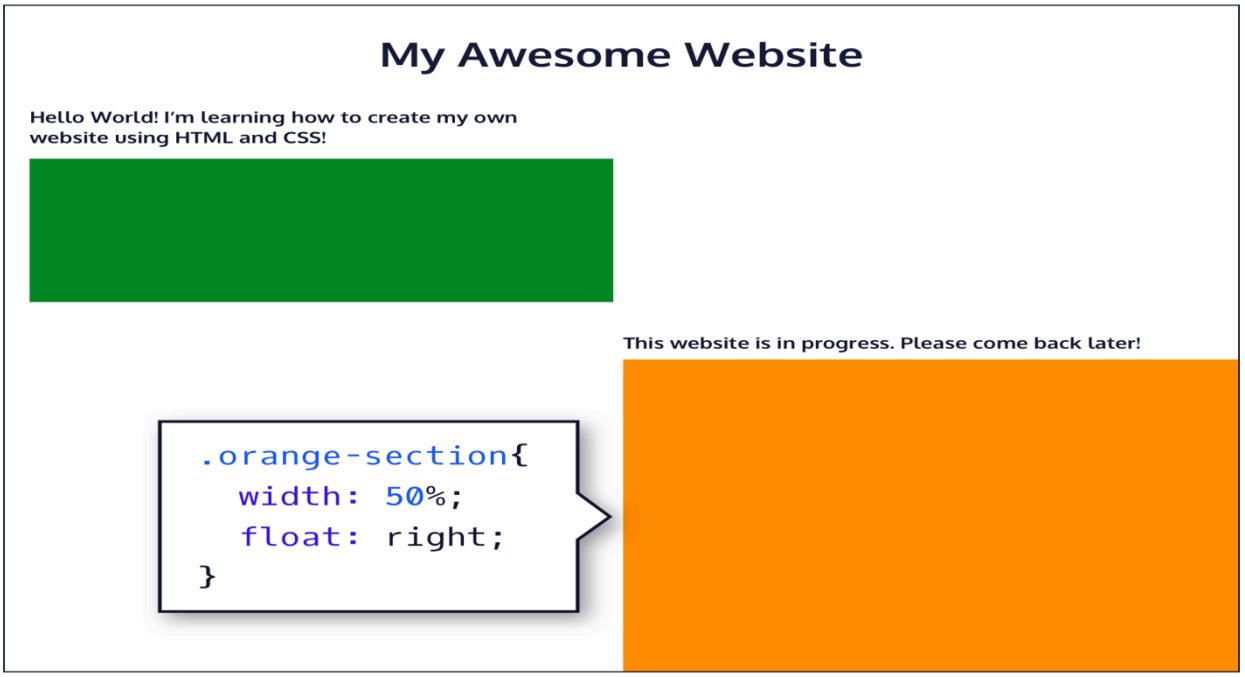
- `left`- move, ou flutua, elementos o mais à esquerda possível.
- `right`- move os elementos o mais para a direita possível.

```
.green-section {
  width: 50%;
  height: 150px;
}

.orange-section {
  background-color: orange;
  width: 50%;
  float: right;
}
```

No exemplo acima, flutuamos o elemento `.orange-section` para o arquivo `right`. Isso funciona para elementos posicionados estáticos e relativos.

Veja o resultado do código abaixo:



Elementos flutuantes devem ter uma largura especificada, como no exemplo acima. Caso contrário, o elemento assumirá a largura total do elemento que o contém e a alteração do valor flutuante não produzirá nenhum resultado visível.

Clear

A propriedade float também pode ser usada para flutuar vários elementos de uma só vez. No entanto, quando vários elementos flutuantes têm alturas diferentes, isso pode afetar seu layout na página. Especificamente, os elementos podem "bater" uns nos outros e não permitir que outros elementos se movam adequadamente para a esquerda ou para a direita.

A propriedade clear especifica como os elementos devem se comportar quando se chocam na página. Pode assumir um dos seguintes valores:

- left— o lado esquerdo do elemento não tocará em nenhum outro elemento dentro do mesmo elemento que o contém.
- right— o lado direito do elemento não tocará em nenhum outro elemento dentro do mesmo elemento que o contém.
- both— nenhum dos lados do elemento tocará em qualquer outro elemento dentro do mesmo elemento que o contém.
- none— o elemento pode tocar qualquer um dos lados.

Exemplo:

```
div {
  width: 200px;
  float: left;
}

div.special {
  clear: left;
}
```

No exemplo acima, todos os s na página flutuam para o lado esquerdo. O elemento com classe special não se moveu totalmente para a esquerda porque um mais alto bloqueou seu posicionamento. Ao definir sua

propriedade clear como left, o special será movido totalmente para o lado esquerdo da página.

Inline Display

Cada elemento HTML tem um display, um valor padrão que determina se ele pode compartilhar espaço horizontal com outros elementos. Alguns elementos preenchem todo o navegador da esquerda para a direita, independentemente do tamanho de seu conteúdo. Outros elementos ocupam apenas o espaço horizontal que seu conteúdo exige e podem estar diretamente ao lado de outros elementos.

Abordaremos três valores para a propriedade display: inline, block e inline-block.

O padrão display para alguns elementos, como , , e <a>, é chamado *inline* . Os elementos inline têm uma caixa que envolve seu conteúdo firmemente, ocupando apenas a quantidade de espaço necessária para exibir seu conteúdo e não exigindo uma nova linha após cada elemento. A altura e a largura desses elementos não podem ser especificadas no documento CSS.

Por exemplo, o texto de uma marca âncora (<a>) será, por padrão, exibido na mesma linha que o texto ao redor e terá a largura necessária para conter seu conteúdo. Os elementos inline não podem ser alterados em tamanho com as propriedades CSS height ou width.

Para saber mais sobre elementos inline leia a documentação MDN.

No exemplo acima, o elemento é inline, pois exibe seu conteúdo na mesma linha do conteúdo ao seu redor, incluindo a marca âncora.

Este exemplo exibirá:

Para saber mais sobre elementos inline , leia a documentação do MDN .

A propriedade CSS display fornece a capacidade de tornar qualquer elemento um elemento inline. Isso inclui elementos que não são embutidos por padrão, como parágrafos, divs e títulos.

```
h1 {
  display: inline;
}
```

O CSS no exemplo acima mudará a exibição de todos os elementos <h1> para inline. O navegador renderizará elementos <h1> na mesma linha que outros elementos inlines imediatamente antes ou depois deles (se houver).

Display: Block

Alguns elementos não são exibidos na mesma linha que o conteúdo ao seu redor. Estes são chamados de elementos de nível de bloco. Esses elementos preenchem toda a largura da página por padrão, mas sua propriedade width também pode ser definida. A menos que especificado de outra forma, eles têm a altura necessária para acomodar seu conteúdo.

Os elementos que são de nível de bloco por padrão incluem todos os níveis de elementos de título (<h1>até <h6>), <p>, e <footer>. Para obter uma lista completa de elementos de nível de bloco, visite a documentação do MDN .

```
strong {
  display: block;
}
```

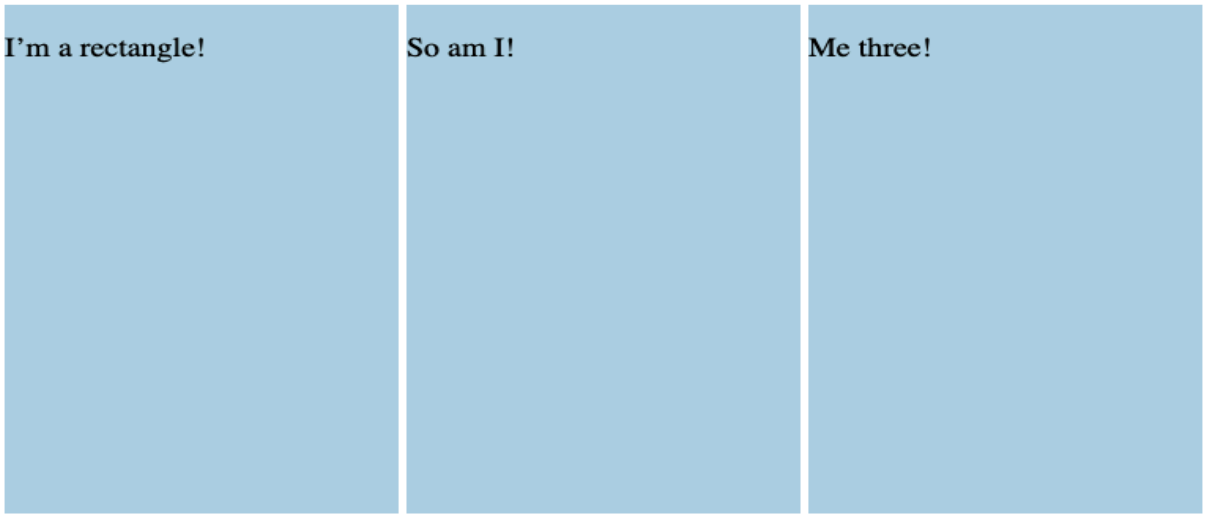
No exemplo acima, todos os elementos serão exibidos em sua própria linha, sem conteúdo diretamente em ambos os lados, mesmo que seu conteúdo não preencha a largura da maioria das telas de computador.

Display: Inline-Block

O terceiro valor para a propriedade display é inline-block. A exibição de bloco em linha combina recursos de elementos de bloco e em linha.

Elementos de bloco inline podem aparecer um ao lado do outro e podemos especificar suas dimensões usando as propriedades width e height. As imagens são o melhor exemplo de elementos de bloco inline padrão.

Por exemplo, os s abaixo serão exibidos na mesma linha e com as dimensões especificadas:



Vamos dar uma olhada no código:

```
<div class="rectangle">
  <p>I'm a rectangle!</p>
</div>

<div class="rectangle">
  <p>So am I!</p>
</div>

<div class="rectangle">
  <p>Me three!</p>
</div>

.rectangle {
  display: inline-block;
```

```
  width: 200px;
  height: 300px;
}
```

Existem três divs retangulares, cada uma contendo um parágrafo de texto. Os `.rectangle` `<div>`s aparecerão todos em linha (desde que haja espaço suficiente da esquerda para a direita) com largura de 200 pixels e altura de 300 pixels, mesmo que o texto dentro deles não exija 200 pixels por 300 pixels de espaço.

Saiba mais:

https://www.w3schools.com/cssref/pr_class_position.asp

https://www.w3schools.com/css/css_z-index.asp

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

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

https://www.youtube.com/watch?v=Y7NeqpwLM2g

https://www.youtube.com/watch?v=E1tR7sYMEN0

https://youtu.be/pMlxfhahXW4

https://youtu.be/f8H-wstL0nU

https://youtu.be/VP1Aw-qMMnI

https://www.youtube.com/watch?v=5PS6ku8NzIE

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

https://www.youtube.com/watch?v=Yj9-N9BEVeM