MÓDULO XII

Introdução ao CSS3 e Estilização Básica

MÓDULO XII

Transições CSS

Ao interagir com um site, você pode notar que muitos elementos têm *estado*.
Por exemplo, os menus suspensos podem estar abertos ou fechados. Os botões podem mudar de cor quando estão em foco ou com o passar do cursor. Os modals aparecem e desaparecem.

Por padrão, o CSS alterna o estilo desses estados instantaneamente.

Usando transições CSS, podemos interpolar entre o estado inicial e o estado de destino do elemento. A transição entre os dois aprimora a experiência do usuário, fornecendo direção visual, suporte e dicas sobre a causa e o efeito da interação.

EDIT ON

MÓDULO XII

Propriedades de transição

transition-property

A propriedade **transition-property** especifica quais estilos vão fazer a transição.

```
.my-element {
  transition-property: background-color;
}
```

O **transition-property** aceita um ou mais nomes de propriedades CSS em uma lista separada por vírgulas.

MÓDULO XII

Você também pode usar **transition-property**: **all**
para indicar que todas as propriedades precisam fazer a transição.

EDIT ON

transition-duration

A propriedade **transition-duration** é usada para definir o tempo que uma transição leva para ser concluída.

EDIT ON

transition-duration aceita unidades de tempo, seja em segundos (s) ou milissegundos (ms) e assume os como padrão.

MÓDULO XII

transition-timing-function

Use a propriedade **transition-timing-function** para variar a velocidade de uma transição CSS ao longo do transition-duration

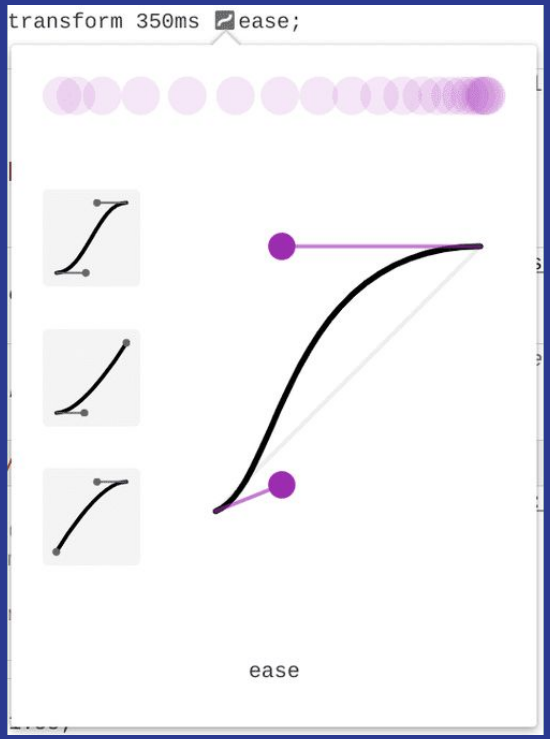
Por padrão, o CSS faz a transição dos elementos a uma velocidade constante (**transition-timing-function: linear**). No entanto, as transições lineares podem parecer um pouco artificiais: na vida real, os objetos têm peso e não podem parar e começar instantaneamente.
A transição suave pode tornar as transições mais animadas e naturais.

EDIT ON

MÓDULO XII

Nosso módulo sobre animação CSS tem uma boa visão geral das funções de tempo.

Você pode usar o **DevTools** para testar diferentes funções de temporização em tempo real.



MÓDULO XII

transition-delay

Use a propriedade **transition-delay** para especificar o momento em que uma transição vai começar. Se **transition-delay** não for especificado, as transições vão começar instantaneamente, porque o valor padrão é 0s. Essa propriedade aceita uma unidade de tempo, por exemplo, segundos (s) ou milissegundos (ms).

EDIT ON

Essa propriedade é útil para transições escalonadas, alcançadas definindo um **transition-delay** mais longo para cada elemento subsequente em um grupo.

EDIT ON

transition-delay também é útil para depuração. Definir o atraso como um valor negativo pode iniciar uma transição mais adiante na linha do tempo.

MÓDULO XII

Abreviação: transition

Como a maioria das propriedades do CSS, há uma versão abreviada. **transition** combina **transition-property**, **transition-duration, transition-timing-function** e **transition-delay**.

```
.longhand {
  transition-property: transform;
  transition-duration: 300ms;
  transition-timing-function: ease-in-out;
  transition-delay: 0s;
}

.shorthand {
  transition: transform 300ms ease-in-out 0s;
}
```

MÓDULO XII

Transformar

A propriedade CSS **transform** é comumente transferida porque é uma propriedade acelerada por GPU que resulta em uma animação mais suave e consome menos bateria. Essa propriedade permite redimensionar, girar, mover ou inclinar um elemento de forma arbitrária.

EDIT ON

Também é possível usar as propriedades scale, rotate ou translate para definir transições distintas para cada propriedade, fora de uma propriedade transform.

MÓDULO XII

Cor

Antes, durante e depois da interação, a cor pode ser um ótimo indicador de estado.

Por exemplo, um botão pode mudar de cor se estiver sendo apontado.

Essa mudança de cor pode fornecer feedback ao usuário de que o botão é clicável.

As propriedades color, **background-color** e **border-color** são apenas alguns lugares em que a cor pode ser transferida após a interação.

MÓDULO XII

Sombras

As sombras geralmente são transferidas para indicar a mudança de elevação, como do foco do usuário.

EDIT ON

MÓDULO XII

Filtros

filter é uma propriedade CSS poderosa que permite adicionar efeitos gráficos em tempo real. A transição entre diferentes estados de **filter** pode gerar resultados impressionantes.

EDIT ON

MÓDULO XII

Acionadores de transição

O CSS precisa incluir uma mudança de estado *e* um evento que aciona essa mudança para que as transições do CSS sejam ativadas. Um exemplo típico desse gatilho é a **pseudoclasse :hover**.

Essa **pseudoclasse** corresponde ao momento em que o usuário passa o cursor sobre um elemento.

Confira a seguir uma lista de algumas pseudoclasses e eventos que podem acionar mudanças de estado nos elementos:

MÓDULO XII

- **:hover:** corresponde se o cursor estiver sobre o elemento.
- **:focus:** corresponde se o elemento estiver com foco.
- **:focus-within :** corresponde se o elemento ou qualquer um dos descendentes dele estiver em foco.
- **:target:** corresponde quando o fragmento do **URL** atual corresponde ao valor do atributo **id** do elemento.
- **:active:** corresponde quando o elemento está sendo ativado (normalmente quando o mouse é pressionado sobre ele).
- Mudança de **class** do **JavaScript**: quando a **class** do CSS de um elemento muda usando **JavaScript**, o CSS faz a transição das propriedades qualificadas que mudaram.

MÓDULO XII

Diferentes transições para entrada ou saída

Ao definir diferentes propriedades transition no passar do cursor/foco, é possível criar alguns efeitos interessantes.

```
.my-element {
  background: red;

  /* This transition is applied on the "exit" transition */
  transition: background 2000ms ease-in;
}

.my-element:hover {
  background: blue;

  /* This transition is applied on the "enter" transition */
  transition: background 150ms ease;
}
```

EDIT ON

MÓDULO XII

Como animar um elemento com transição básica ao passar o ponteiro por cima dele?

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Static Template</title>
  </head>
  <style>
    .elem {
      background: blueviolet;
      width: 300px;
      height: 150px;
    }
    .elem:hover {
      opacity: 0.5;
    }
  </style>
  <body>
    <div class="elem"></div>
  </body>
</html>
```

MÓDULO XII

Esta é uma transição simples que pode ser acionada quando passamos o ponteiro do mouse sobre o elemento. Podemos adicionar mais de uma transição que será executada ao mesmo tempo.

Vamos adicionar uma propriedade de transformação da escala para aumentar a escala do elemento em transição.

```
.elem:hover {
    transform: scale(1.1);
  }
```

MÓDULO XII

A transição, no entanto, não parece suave, pois não definimos a duração da transição nem usamos alguma função de tempo.

Se adicionarmos a propriedade transition, ela fará o elemento se mover mais suavemente.

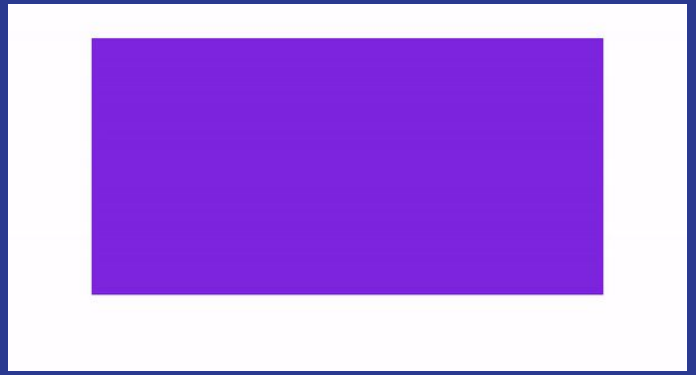
```
.elem {
    background: blueviolet;
    width: 300px;
    height: 150px;
    margin: 20px auto;
    transition: 500ms linear;
  }
```

MÓDULO XII

A transição, no entanto, não parece suave, pois não definimos a duração da transição nem usamos alguma função de tempo.

Se adicionarmos a propriedade transition, ela fará o elemento se mover mais suavemente.

```
.elem {
    background: blueviolet;
    width: 300px;
    height: 150px;
    margin: 20px auto;
    transition: 500ms linear;
  }
```



MÓDULO XII

Vamos dividir o código em partes para ver como a propriedade ***transition*** funciona:

```
transition: 500ms linear;
```

- 500ms: a duração da transição é em milissegundos
- linear: a função de tempo

```
div {
    transition: <propriedade> <duração> <função de tempo> <atraso>;
}
```

MÓDULO XII

Podemos adicionar mais opções como as que estão abaixo

```css
#delay {
  transition-property: font-size;
  transition-duration: 4s;
  transition-delay: 2s;
}
```

O que faz esse código?

- **transition-property:** a propriedade que você quer animar.
  Pode ser qualquer elemento do CSS, como **background, height, translateY, translateX** e assim por diante.
- **transition-duration:** à duração da transição
- **transition-delay:** o atraso antes de a transição começar

MÓDULO XII

Como tornar as transições mais interativas usando a propriedade animation e keyframes

Podemos fazer mais com as transições do CSS para tornar essa animação mais criativa e interativa.

Como mover um elemento com keyframes

um exemplo onde o elemento se move do ponto **A** para o ponto **B**. Usaremos **translateX** e **translateY**.

MÓDULO XII

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Static Template</title>
  </head>
  <style>
    .elem {
      background: orange;
      width: 300px;
      height: 150px;
      animation: moveToRight 2s ease-in-out;
      animation-delay: 1000ms;
    }

    @keyframes moveToRight {
      0% {
        transform: translateX(0px);
      }
      100% {
        transform: translateX(300px);
      }
    }
  </style>
  <body>
    <div class="elem"></div>
  </body>
</html>
```

MÓDULO XII

Desta vez, usamos novas propriedades, como **animation** e **keyframes**.
Usamos a propriedade **animation** para definir o nome da animação e a duração, e usamos **keyframes** para descrever como deve ser a movimentação do elemento.

```
animation: moveToRight 2s ease-in-out;
```

Chamei a animação de **moveToRight** – mas é possível usar o nome que você quiser.
A duração é de **2s** , e **ease-in-out** é uma função de tempo.
Existem outras funções de tempo que podem ser usadas, como **ease-in, linear, ease-out**, que, basicamente, tornam a animação mais suave.

MÓDULO XII

@keyframes recebe o nome da animação.
Neste caso, o nome é **moveToRight**.

```
@keyframes moveToRight {
    0% {
      transform: translateX(0px);
    }
    100% {
      transform: translateX(300px);
    }
  }
```

keyframes executará a animação em diversas etapas.
O exemplo acima usa a porcentagem para representar o intervalo ou a ordem das transições.

MÓDULO XII

Também podemos usar os métodos **from** e **to**, como vemos abaixo:

```
@keyframes moveToRight {
    from {
       transform: translateX(0px);
     }
    to {
       transform: translateX(300px);
     }
  }
```

from: representa o ponto de início ou a primeira parte da animação.
to: é o ponto final ou a última parte da animação a ser executada.

MÓDULO XII

Você pode querer usar uma porcentagem em alguns casos.
Por exemplo, digamos que você queira adicionar mais do que duas transições que serão executadas em sequência, como vemos abaixo:

```
@keyframes moveToRight {
    0% {
       transform: translateX(0px);
     }
    50% {
       transform: translateX(150px);
     }
     100% {
       transform: translateX(300px);
     }
  }
```

MÓDULO XII

Você pode ser mais criativo e animar muitas propriedades ao mesmo tempo, como no exemplo abaixo:

Há muito mais coisas que se pode fazer com os **keyframes**. Primeiro, vamos adicionar mais transições à nossa animação.

MÓDULO XII

Como animar mais propriedades e incluí-las na transição

Desta vez, vamos animar o segundo plano e faremos o elemento se mover em um padrão quadrado. Faremos a animação rodar indefinidamente usando a propriedade **infinite** como uma função de tempo.

Edit Sandbox

Vamos examinar o código por partes.
Primeiro, adicionamos **infinite** para fazer com que a animação rodasse indefinidamente.

MÓDULO XII

```
animation: moveToLeft 5s linear infinite;
```

Em seguida, dividimos a animação em quatro partes.
A cada parte, executamos uma transição diferente e todas as animações serão executadas em uma sequência.

- **Primeiro passo**: defina o elemento horizontalmente para **translateX(opx)** e altere o segundo plano de acordo com o gradiente.

```
0% {
        transform: translateX(0px);
        background: linear-gradient(
          to right,
          #ff8177 0%,
          #ff867a 0%,
          #ff8c7f 21%,
          #f99185 52%,
          #cf556c 78%,
          #b12a5b 100%
        );
      }
```

MÓDULO XII

- A **segunda** animação moverá o elemento da esquerda para a direita e alterará a cor do fundo.

```
25% {
        transform: translateX(400px);
        background: linear-gradient(120deg, #84fab0 0%, #8fd3f4 100%);
    }
```

- A **terceira** animação moverá o elemento para baixo usando **translateY** e alterará novamente a cor de fundo.

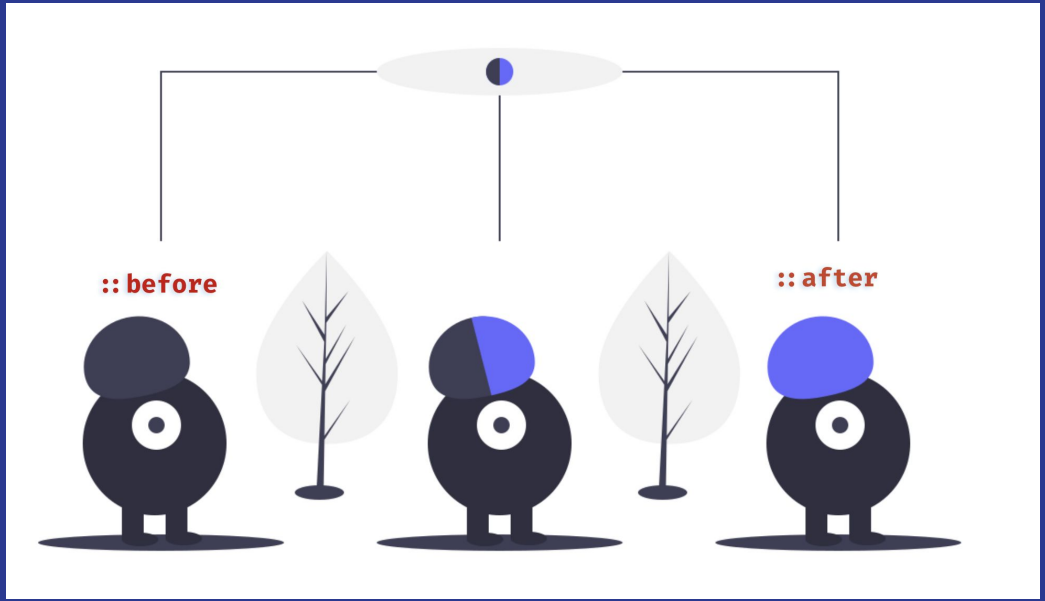
MÓDULO XII

- Na **quarta** parte, o elemento se moverá de volta para a esquerda e mudará sua cor de fundo.

- Na **quinta** animação, o elemento deve voltar ao seu local original.

MÓDULO XII

CSS: pseudo-elementos ::before e ::after



A Educação é o primeiro passo para um futuro melhor...

MÓDULO XII

Esses pseudo-elementos servem para "criar elementos" sem necessidade de cria-los no HTML.

Exemplo:

```
.meu-elemento {
  background-color: #7CB518;
}
```

```
<span class="meu-elemento">Olá mundo!</span>
```

MÓDULO XII

Se quisermos adicionar algo antes desse elemento, no CSS:

- Mantemos o estilo principal da classe "meu-elemento"
- Criamos uma outra estilização para "meu-elemento" adicionando ao lado direito do nome **::before**
- Usamos a propriedade content para adicionar um conteúdo para aquele elemento
- E adicionamos algum estilo, por exemplo o **background-color**

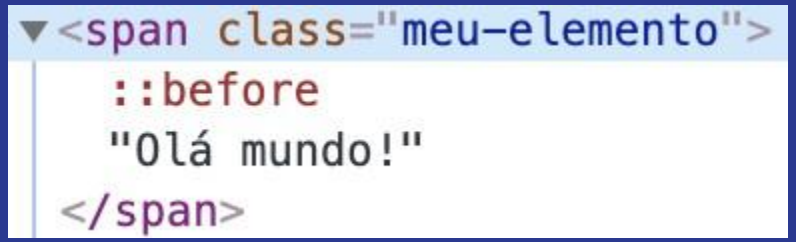
```
.meu-elemento {
  background-color: #7CB518;
}

.meu-elemento::before {
  content: ">>> ";
  background-color: #FB6107;
}
```

MÓDULO XII

O resultado:

E quando inspecionamos o HTML, é mostrado dessa forma:



MÓDULO XII

E agora se formos acrescentar algo depois do elemento utilizamos o **::after**

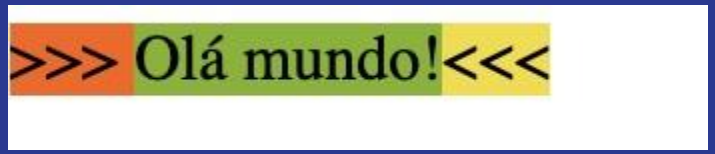
```
.meu-elemento {
  background-color: #7CB518;
}

.meu-elemento::before {
  content: ">>> ";
  background-color: #FB6107;
}

.meu-elemento::after {
  content: "<<<";
  background-color: #F3DE2C;
}
```

MÓDULO XII

E o resultado:

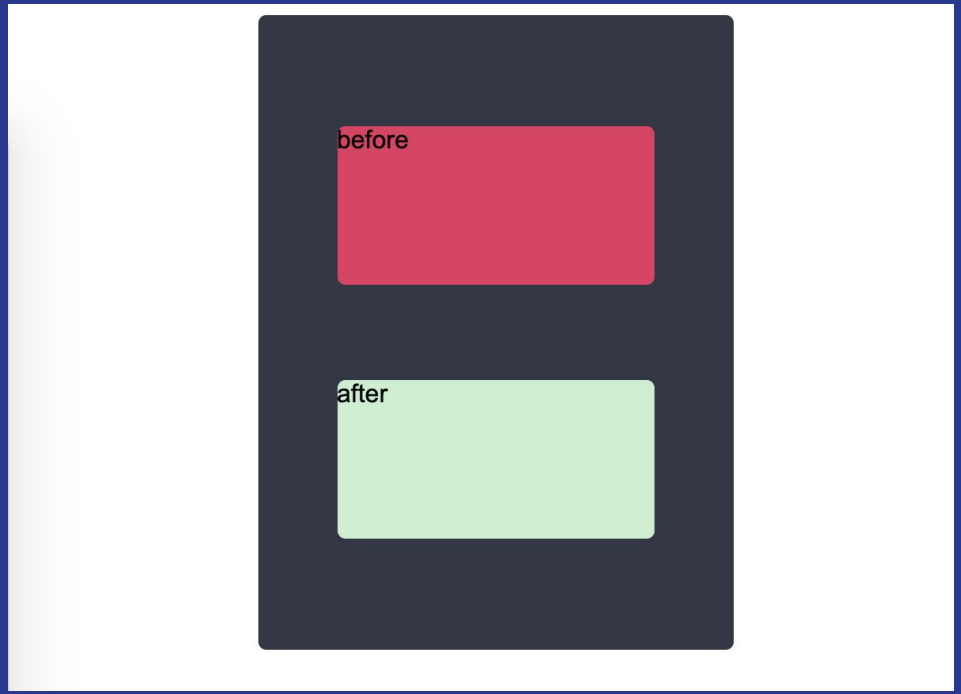


E inspecionando o HTML

```
▼<span class="meu-elemento">
    ::before
    "Olá mundo!"
    ::after
  </span>
```

MÓDULO XII

Nessa imagem, "vemos" pelo menos uns 3 elementos.



MÓDULO XII

```css
.retangulo{
  width: 300px;
  height: 400px;
  background-color: #333745;
  position: relative;
  border-radius: 5px;
  margin:auto;
}

.retangulo::before{
  content: 'before';
  position: absolute;
  width: 200px;
  height: 100px;
  background-color: #E63462;
  top: 70px;
  left: 50px;
  border-radius: 5px;
}
.retangulo::after{
  content: 'after';
  position: absolute;
  width: 200px;
  height: 100px;
  background-color: #C7EFCF;
  bottom: 70px;
  left: 50px;
  border-radius: 5px;
}
```

Porém tem apenas um no **HTML**:

```html
<div class="retangulo"></div>
```

Que foram adicionados o **::before** e **::after**

MÓDULO XII

O que fiz:

1. Na classe somente "**.retangulo**" acrescentei a propriedade position com valor relative para que o que tiver dentro dela eu consiga manipular melhor.
2. Nos pseudo-elementos **::before** e **::after**:

- coloquei um content (normalmente eu coloco vazio, pois não quero conteúdo, só quero um elemento extra, coloquei apenas para ficar visível)
- Coloquei um position: absolute, com isso eu consigo manipular usando as propriedades **top, botom, left, right** (com o position: absolute meus elementos perdem seus posicionamentos e ficam "**flutuando**" tendo como referência o elemento que está acima dele "hierarquicamente falando")
- defini tamanhos, cores, "**arredondamento**" de borda...

MÓDULO XII

Pseudo Elementos CSS

Um pseudo elemento CSS é uma palavra-chave que pode ser adicionada a um seletor, para estilizar um parte específica de um elemento.

Alguns usos comuns para pseudo elementos:

- Estilize a primeira letra ou primeira linha de um elemento
- Inserir conteúdo antes ou depois de um elemento
- Estilize os marcadores dos itens da lista
- Estilize a parte selecionada pelo usuário de um elemento
- Estilize a caixa de visualização atrás de uma caixa de diálogo

MÓDULO XII

Sintaxe

Os pseudo elementos são denotados por dois pontos duplos (**::**) seguidos pelo nome do pseudo elemento:

```
selector::pseudo-element-name {
  CSS properties
}
```

MÓDULO XII

O pseudo elemento CSS ::first-line

O **::first-line** pseudo elemento é usado para adicionar um estilo especial para a primeira linha de um texto.
Nota: O **::first-line** pseudo elemento só pode ser aplicado em nível de bloco elementos.

Exemplo

Formate a primeira linha de texto em todos os **<p>** elementos:

```
p::first-line {
  color: red;
  font-variant: small-caps;
  font-size: 19px;
}
```

Experimente você mesmo »

MÓDULO XII

O pseudo elemento CSS ::first-letter

O **::first-letter** pseudo elemento é usado para adicionar um estilo especial ao primeiro letra de um texto.

Nota: O **::first-letter** pseudo elemento só pode ser aplicado em nível de bloco elementos.

Formate a primeira letra do texto em todos os **<p>** elementos:

```
p::first-letter {
  color: red;
  font-size: xx-large;
}
```

Experimente você mesmo »

MÓDULO XII

O CSS **::before** do pseudo elemento

O **::before** pseudo elemento é usado para inserir algum conteúdo antes do conteúdo de um elemento especificado.

Use o **content** propriedade para especificar o conteúdo a inserir.

Insira uma imagem antes do conteúdo de cada elemento **<h3>**:

```
h3::before {
  content: url(smiley.gif);
}
```

Experimente você mesmo »

MÓDULO XII

O CSS **::after** Pseudo-elemento

O **::after** pseudo elemento é usado para inserir algum conteúdo após o conteúdo de um elemento especificado.

Use o **content** propriedade para especificar o conteúdo a inserir.

Insira uma imagem após o conteúdo de cada elemento **<h3>**:

```
h3::after {
  content: url(smiley.gif);
}
```

Experimente você mesmo »

A Educação é o primeiro passo para um futuro melhor...

MÓDULO XII

O pseudo elemento CSS **::marker**

O **::marker** pseudo elemento é usado para estilizar os marcadores de itens da lista.

Estilize os marcadores dos itens da lista:

```
::marker {
  color: red;
  font-size: 23px;
}
```

Experimente você mesmo »

MÓDULO XII

O pseudo elemento CSS **::selection**

O **::selection** pseudo elemento é usado para estilizar a parte de um texto selecionada por um usuário.

Estilize o texto selecionado pelo usuário com uma cor vermelha e um fundo amarelo:

```
::selection {
  color: red;
  background: yellow;
}
```

Experimente você mesmo »

MÓDULO XII

O pseudo elemento CSS **::backdrop**

O **::backdrop** pseudo elemento é usado para estilizar a caixa de visualização atrás de uma caixa de diálogo ou elemento **popover**.

Estilize a caixa de visualização atrás de uma caixa de diálogo:

```
dialog::backdrop {
  background-color: lightgreen;
}
```

Experimente você mesmo »

MÓDULO XII

Pseudo Elementos e classes HTML

Pseudo Elementos podem ser facilmente combinados com classes HTML.

Exiba a primeira letra dos elementos **<p>** com **class="intro"**, em vermelho e em tamanho maior:

```
p.intro::first-letter {
  color: #ff0000;
  font-size: 200%;
}
```

Experimente você mesmo »

MÓDULO XII

Pseudo Elementos múltiplos

Vários pseudo elementos também podem ser combinados.

No exemplo a seguir, a primeira letra dos elementos **<p>** será vermelha e em um tamanho de fonte **xx-large**. O resto da primeira linha será azul e em letras maiúsculas pequenas.

O restante do parágrafo estará no tamanho e cor de fonte padrão:

```
p::first-letter {
  color: red;
  font-size: xx-large;
}

p::first-line {
  color: blue;
  font-variant: small-caps;
}
```

Experimente você mesmo »

MÓDULO XII

Opacidade da imagem CSS

O **opacity** propriedade específica a *opacidade/transparência* de um elemento.
O **opacity** a propriedade pode assumir um valor de *0,0* a *1,0*:

- **0,0** - O elemento será totalmente transparente
- **0,5** - O elemento será **50%** transparente
- **1.0** - Padrão. O elemento será totalmente opaco

MÓDULO XII

opacidade 0,2

opacidade 0,5

opacidade 1.0 (padrão)

```
img {
  opacity: 0.5;
}
```

Experimente você mesmo »

MÓDULO XII

Opacidade e :hover

O **opacity** a propriedade é frequentemente usada com para alterar a opacidade ao passar o mouse**: :hover**

```css
img {
  opacity: 0.5;
}

img:hover {
  opacity: 1.0;
}
```

Experimente você mesmo »

MÓDULO XII

Efeito de pairar invertido

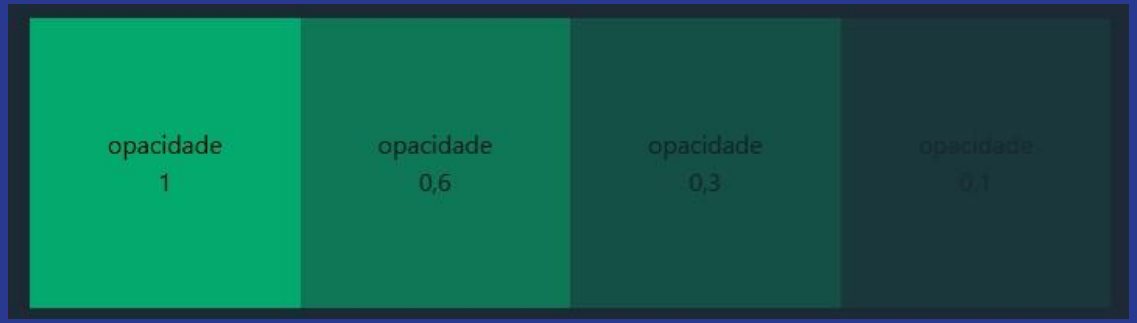
```
img:hover {
  opacity: 0.5;
}
```

Experimente você mesmo »

MÓDULO XII

Caixas transparentes

Ao usar o propriedade para adicionar transparência ao fundo de um elemento, todos os elementos filhos herdar a mesma transparência. Isso pode dificultar a leitura do texto dentro de um elemento transparente: **opacity**




```
div {
  opacity: 0.3;
}
```

Try it Yourself »

MÓDULO XII

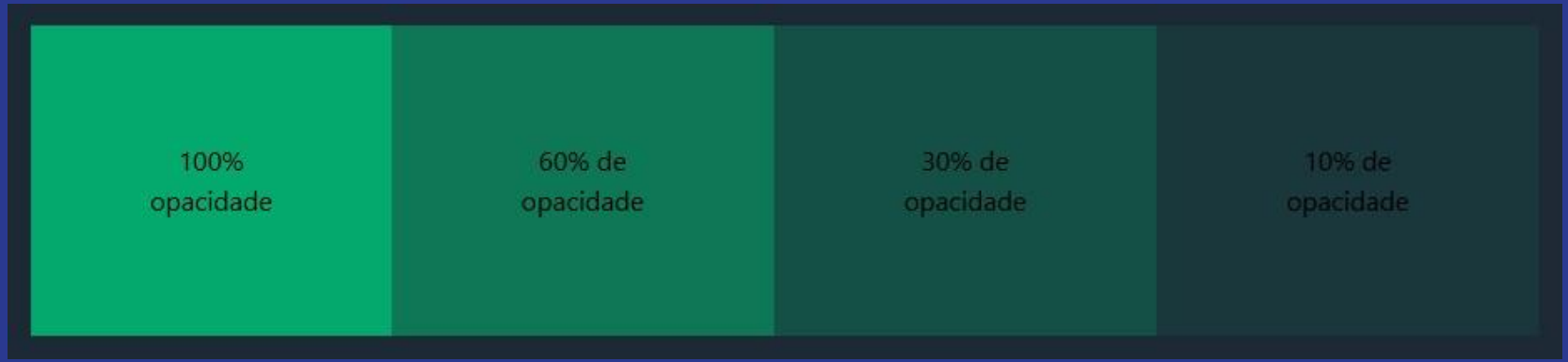
Transparência usando cor de fundo

Para NÃO aplicar a transparência aos elementos filhos, você pode usar o propriedade com um valor **RGBA**. **background-color**
Os valores de cor **RGBA** são uma extensão dos valores de cor **RGB** com um canal **alfa** - que especifica a **opacidade** de uma cor.

Um valor de cor **RGBA** é especificado com: (*vermelho, verde, azul, alfa*).
Onde o parâmetro **alfa** é um número entre **0,0** (*totalmente transparente*) e **1,0** (*totalmente opaco*).

MÓDULO XII

O exemplo a seguir define a opacidade da cor de fundo e não do texto:




```
div {
  background: rgba(4, 170, 109, 0.3) /* Green background with 30% opacity */
}
```

Experimente você mesmo »

MÓDULO XII

```
<html>
<head>
<style>
div.background {
  background: url(klematis.jpg) repeat;
  border: 2px solid black;
}

div.transbox {
  margin: 30px;
  background-color: rgba(255, 255, 255, 0.6);
  border: 1px solid black;
}

div.transbox p {
  margin: 5%;
  font-weight: bold;
  color: #000000;
}
</style>
</head>
<body>

<div class="background">
  <div class="transbox">
    <p>This is some text that is placed in the transparent box.</p>
  </div>
</div>

</body>
</html>
```

Texto em caixa transparente

Experimente você mesmo »

MÓDULO XII

Box Shadow - Propriedade CSS

Interação com a luz é a maneira como vemos as coisas, nessa interação com a luz é comum a geração de sombras sobre os objetos, o uso de sombras em interfaces Web trás uma sensação de volume e elevação

Conceito

A propriedade **box-shadow** do **CSS** projeta uma cópia do elemento com uma cor única

```
box-shadow: <deslocamento-horizontal> <deslocamento-vertical> <raio de desfoque>
  <raio de espalhamento> <cor>;
```

MÓDULO XII

Detalhamento dos atributos

deslocamento horizontal: Quando a sombra está projetada para **direita** quando positivo e esquerda para valores negativas;
deslocamento vertical: Quando a sombra está projetada para **baixa** quando positivo e cima para valores negativas;
raio de desfoque: Quando a sombra possui desfoque devido a difusão da luz, a irregularidade de incidência dela;
raio de espalhamento (opcional): Tamanho adicional para a sombra, somando medida a partir do centro ou reduzindo da borda quando negativo, uma boa indicação de ângulo da luz sobre o objeto;
cor: Cor da sombra, lembrado a possibilidade de controlar opacidade pela cor.

MÓDULO XII

Incidência de luz

Podemos usar para projetar uma sombra e indicar luz

MÓDULO XII

Luz Ambiente

É a luz vindo sem direcionamento de todos os ângulos criando uma sombra difusa, fazemos ela sem colocar deslocamento

```
box-shadow: 0 0 8px 1px #000003f;
```

Luz Chave

Uma luz chave é direcionada, com deslocamento na direção oposta da luz, no caso vindo de cima e do lado direito para fazer essa vamos colocar o deslocamento da direita (**4px**) e para baixo (**4px**)

```
box-shadow: 4px 4px 8px 1px #000003f;
```

MÓDULO XII

Exemplos da direção das luzes

Várias fontes de luzes

MÓDULO XII

Luz Combinada

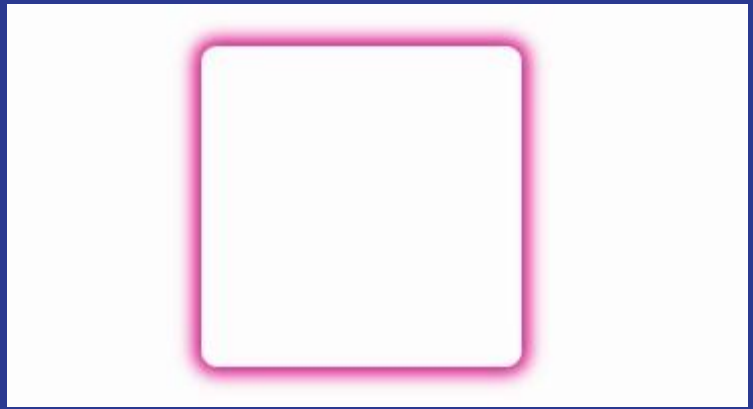
Junção das luzes ambientes e luz chave, adicionando as duas sombras como uma lista separado por vírgula

```
box-shadow:
  0 0 8px 1px #000003f,
  4px 4px 8px 1px #000003f;
```

MÓDULO XII

Luz de fundo

Um objeto com uma fonte de luz uniforme vindo de trás, como uma sanca no teto



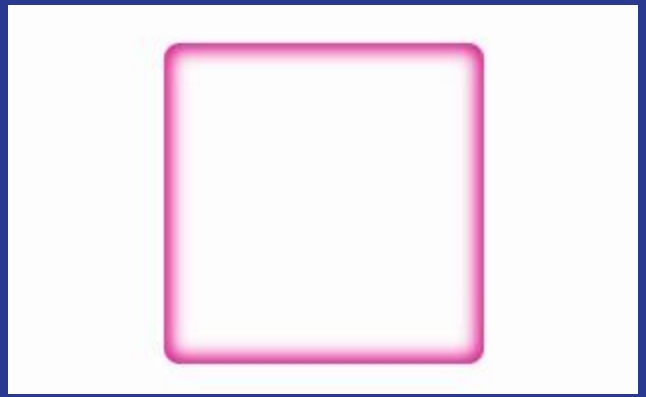
Usamos uma sombra sem deslocamentos com desfoque e espalhamento

```
box-shadow: 0 0 8px 4px #e0138c;
```

MÓDULO XII

Iluminação interna

Com as luzes vindo de forma regular internamente das margens



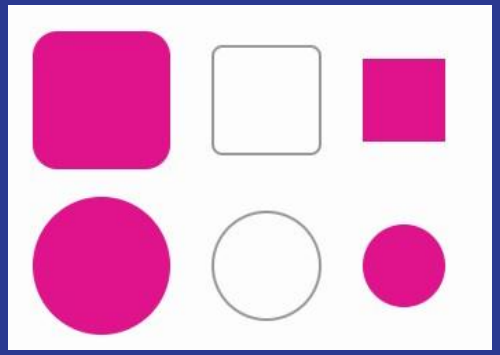
Basta incluir o inset para frente dos atributos

```
box-shadow: inset 0 0 8px 4px #e0138c;
```

MÓDULO XII

Objetos espelhos

As sombras podem ser usadas para desenhar novos objetos com o mesmo formato



Podemos fazer o espalhamento um como positivo para aumentar e outro como negativo para reduzir o tamanho da sombra

```
box-shadow:
  -120px 0 0 10px #e0138c,
  100px 0 0 -10px #e0138c;
```

MÓDULO XII

Efeito de elevação com mouse

Para mudança de elevação podemos aumentar o desfoque da luz ambiente e deslocamento e desfoque para a luz chave, como no exemplo ao passar o mouse

Elemento HTML com a classe **card-box-shadow**

```
<div class="card-box-shadow"></div>
```

MÓDULO XII

Estilo com **transition** de *300ms* **hover** para um efeito suave passando a impressão que está subindo

```css
.card-box-shadow {
  width: 160px;
  height: 160px;
  border-radius: 8px;
  box-shadow:
    0 0 8px 1px #000003f,
    8px 8px 16px 2px #000003f;
  transition: box-shadow 0.3s;
}

.card-box-shadow:hover {
  box-shadow:
    0 0 16px 2px #000003f,
    16px 16px 32px 4px #000003f;
}
```

MÓDULO XII

Pequeno exemplo

Uma pequena chama com sombras, ou luzes :)

```css
.fire {
  width: 50px;
  height: 50px;
  background: #ffb726;
  border-radius: 15% 75% 50% 75%;
  transform: rotate(45deg);
  box-shadow:
    -10px -10px 0 15px #ff7d17,
    -20px -20px 0 30px #ff6236,
    -30px -30px 10px 45px #ff62343f;
}
```

MÓDULO XII

EXERCÍCIOS...

A Educação é o primeiro passo para um futuro melhor...

MÓDULO XII

FIM...

A Educação é o primeiro passo para um futuro melhor...