

VOL
02



Guia

Descomplica
FRONT - END

Guia

Descomplica

FRONT - END



Autor: Artur Aguilar

Propriedade: +Education

ISENÇÃO DE RESPONSABILIDADE

Todos os direitos autorais das obras criadas por nossa empresa são de propriedade exclusiva de +education. Isso inclui, sem limitação, textos, imagens, logotipos, vídeos, design gráfico, software, e quaisquer outras produções intelectuais desenvolvidas no âmbito de nossos projetos e serviços. A utilização, reprodução ou distribuição de qualquer conteúdo sem a devida autorização prévia e por escrito da +education é estritamente proibida e estará sujeita às penalidades legais cabíveis.

EDUCATION

EDUCATION

+ EDUCATION

+ EDUCATION

+ EDUCATION

+ EDUCATION

+ EDUCATION

+ EDUCATION

EDUCATION



CAPÍTULO

01

O PODER DAS CORES E DA TIPOGRAFIA

A CIÊNCIA DAS CORES E DA TIPOGRAFIA

AS CORES E A TIPOGRAFIA SÃO ELEMENTOS ESSENCIAIS NO DESIGN, CAPAZES DE TRANSMITIR EMOÇÕES, CRIAR IDENTIDADES VISUAIS MARCANTES E FACILITAR A COMUNICAÇÃO. A ESCOLHA CUIDADOSA DESSES ELEMENTOS PODE IMPACTAR DIRETAMENTE A EXPERIÊNCIA DO USUÁRIO, TORNANDO AS MENSAGENS MAIS ATRAENTES E EFICAZES.

AS CORES TÊM O PODER DE EVOCAR SENTIMENTOS E ASSOCIAR SIGNIFICADOS, SENDO AMPLAMENTE UTILIZADAS EM ESTRATÉGIAS VISUAIS PARA DESTACAR INFORMAÇÕES, GUIAR A ATENÇÃO E REFORÇAR A IDENTIDADE DE MARCAS. DO CALOR DO VERMELHO À TRANQUILIDADE DO AZUL, CADA COR POSSUI UM PAPEL PSICOLÓGICO QUE INFLUENCIA A PERCEPÇÃO DO PÚBLICO.

A TIPOGRAFIA, POR SUA VEZ, É A ARTE DE ORGANIZAR OS TEXTOS DE FORMA CLARA E ESTETICAMENTE AGRADÁVEL. FONTES, TAMANHOS, ESPAÇAMENTO E ALINHAMENTO SÃO ASPECTOS QUE DETERMINAM A LEGIBILIDADE E A PERSONALIDADE DO DESIGN. ENQUANTO FONTES SERIFADAS TRANSMITEM TRADIÇÃO E ELEGÂNCIA, FONTES SEM SERIFA SÃO ASSOCIADAS A MODERNIDADE E SIMPLICIDADE.

COMBINANDO CORES HARMONIOSAS E TIPOGRAFIAS BEM PLANEJADAS, É POSSÍVEL CRIAR COMPOSIÇÕES VISUAIS QUE NÃO APENAS INFORMAM, MAS TAMBÉM ENCANTAM E INSPIRAM. ENTENDER OS PRINCÍPIOS BÁSICOS DESSAS ÁREAS É ESSENCIAL PARA QUALQUER PROFISSIONAL QUE DESEJE SE DESTACAR NO UNIVERSO CRIATIVO.

PSICOLOGIA DAS CORES

A PSICOLOGIA DAS CORES ESTUDA COMO AS CORES INFLUENCIAM O COMPORTAMENTO HUMANO, NOSSAS EMOÇÕES E PERCEPÇÕES. CADA COR PODE DESENCADEAR DIFERENTES REAÇÕES E É USADA ESTRATEGICAMENTE EM VÁRIAS ÁREAS, COMO MARKETING, DESIGN E ARTE, PARA EVOCAR SENSAÇÕES ESPECÍFICAS E INFLUENCIAR A DECISÃO DE COMPRA OU A PERCEPÇÃO DE UMA MARCA. AQUI ESTÃO ALGUNS EXEMPLOS DE ASSOCIAÇÕES COMUNS COM CADA COR:

- VERMELHO:** ASSOCIADO À ENERGIA, PAIXÃO E URGÊNCIA. ESTIMULA O APETITE, RAZÃO PELA QUAL É COMUM EM RESTAURANTES. TAMBÉM PODE TRANSMITIR SENTIMENTOS DE ALERTA OU PERIGO.
- AZUL:** TRANSMITE CALMA, CONFIANÇA E SEGURANÇA. É MUITO USADO EM MARCAS QUE QUEREM PASSAR UMA IMAGEM DE ESTABILIDADE E PROFISSIONALISMO, COMO BANCOS E EMPRESAS DE TECNOLOGIA.
- AMARELO:** EVOCÀ FELICIDADE, OTIMISMO E ENERGIA. PORÉM, EM EXCESSO, PODE CAUSAR ANSIEDADE. MARCAS INFANTIS E EMPRESAS QUE QUEREM TRANSMITIR POSITIVIDADE O UTILIZAM BASTANTE.
- VERDE:** SIMBOLIZA NATUREZA, SAÚDE E TRANQUILIDADE. TAMBÉM ESTÁ ASSOCIADO AO DINHEIRO E PROSPERIDADE. É MUITO USADO EM EMPRESAS DE PRODUTOS NATURAIS, DE SUSTENTABILIDADE OU DE BEM-ESTAR.
- LARANJA:** COMUNICA ENTUSIASMO, CRIATIVIDADE E DETERMINAÇÃO. ASSIM COMO O VERMELHO, PODE ESTIMULAR O APETITE E É COMUM EM CAMPANHAS QUE QUEREM ATRAIR A ATENÇÃO DE FORMA AMIGÁVEL E ACESSÍVEL.
- ROXO:** ASSOCIADO AO LUXO, MISTÉRIO E CRIATIVIDADE. É COMUMENTE USADO EM PRODUTOS DE BELEZA OU MARCAS QUE QUEREM TRANSMITIR SOFISTICAÇÃO E ORIGINALIDADE.
- PRETO:** SUGERE ELEGÂNCIA, PODER E AUTORIDADE. EMBORA SEJA VERSÁTIL, EM EXCESSO PODE PARECER OPPRESSOR. É POPULAR EM MARCAS DE LUXO E DESIGN MINIMALISTA.
- BRANCO:** REPRESENTA PUREZA, SIMPLICIDADE E FRESCOR. MUITAS VEZES USADO NO DESIGN PARA CRIAR UM VISUAL LIMPO E MODERNO, ALÉM DE DAR SENSAÇÃO DE ESPAÇO.



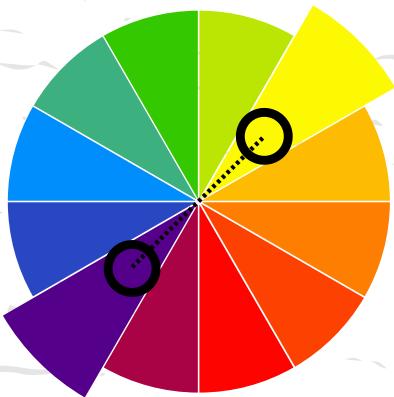
CÍRCULO CROMÁTICO

HARMONIA DE CORES

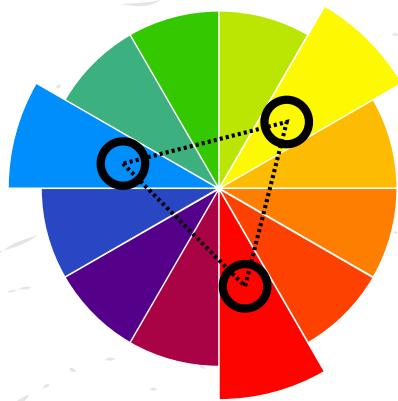
A HARMONIA DE CORES REFERE-SE À COMBINAÇÃO DE CORES DE MANEIRA QUE ELAS SE COMPLEMENTEM, CRIANDO UM EFEITO ESTÉTICO AGRADÁVEL E EQUILIBRADO. EXISTEM VÁRIAS TEORIAS E ESQUEMAS DE HARMONIA DE CORES QUE PODEM SER UTILIZADOS EM DESIGN, ARTE E DECORAÇÃO. AQUI ESTÃO ALGUNS DOS PRINCIPAIS:



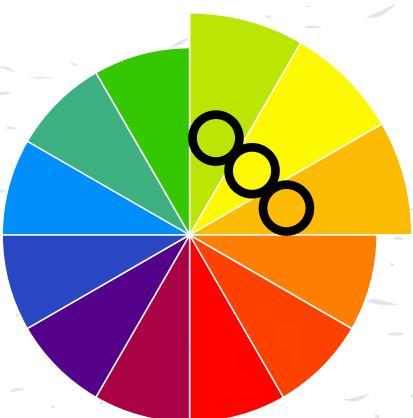
CORES COMPLEMENTARES



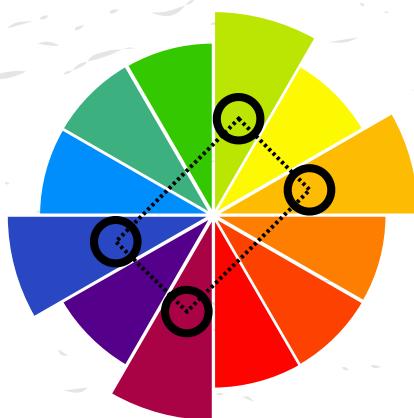
CORES TRIÁDICAS

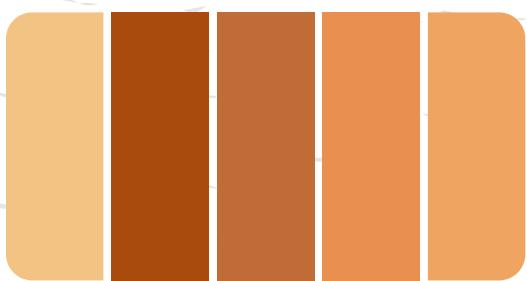
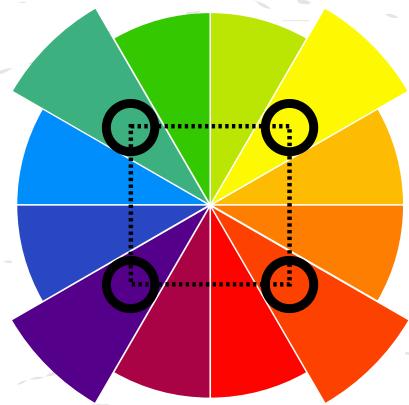
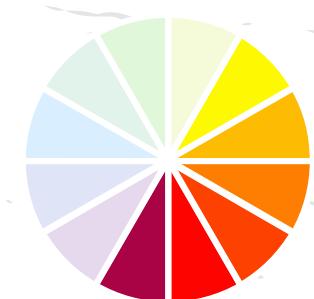


CORES ANÁLOGAS



CORES TETRÁDICAS



CORES MONOCROMÁTICAS**CORES TRIÁDICAS****CORES PRIMÁRIAS****CORES SECUNDÁRIAS****CORES TERCIÁRIAS****TEMPERATURA DAS CORES**

TIPOGRAFIA

ELEMENTOS BÁSICOS DA TIPOGRAFIA

- **FONTE:** CONJUNTO DE CARACTERES COM O MESMO DESIGN (EX: ARIAL, TIMES NEW ROMAN).
- **FAMÍLIA TIPOGRÁFICA:** CONJUNTO DE VARIAÇÕES DE UMA MESMA FONTE (EX: ARIAL REGULAR, ARIAL BOLD).
- **PESO:** ESPESSURA DA FONTE (EX: LEVE, REGULAR, NEGRITO).
- **ALTURA DA LINHA (LINE-HEIGHT):** ESPAÇAMENTO VERTICAL ENTRE LINHAS DE TEXTO, QUE AJUDA NA LEGIBILIDADE.
- **ESPAÇAMENTO ENTRE LETRAS (LETTER-SPACING):** ESPAÇAMENTO HORIZONTAL ENTRE LETRAS, QUE AFETA A DENSIDADE DO TEXTO.

- **VW E VH (VIEWPORT WIDTH E VIEWPORT HEIGHT):** RELATIVAS AO TAMANHO DA JANELA DE VISUALIZAÇÃO. EX: WIDTH: 100VW; OCUPA TODA A LARGURA DA TELA.
- **VMIN E VMAX:** RELATIVAS AO MENOR OU MAIOR LADO DA VIEWPORT. EX: WIDTH: 50VMIN; ADAPTA-SE AUTOMATICAMENTE.

QUANDO USAR CADA UMA

- MEDIDAS ABSOLUTAS (COMO PX) SÃO ÚTEIS PARA ELEMENTOS QUE PRECISAM DE UM TAMANHO FIXO, COMO UM ÍCONE ESPECÍFICO.
-
- MEDIDAS RELATIVAS SÃO IDEIAS PARA DESIGN RESPONSIVO E ACESSÍVEL. EX: REM PARA TAMANHOS DE FONTE QUE SE AJUSTAM COM A CONFIGURAÇÃO DO NAVEGADOR DO USUÁRIO, VW E VH PARA SE ADAPTAR À TELA DO DISPOSITIVO.

TAMANHO DE FONTE E SUAS MEDIDAS

MEDIDAS ABSOLUTAS

AS MEDIDAS ABSOLUTAS SÃO FIXAS E NÃO MUDAM COM O TAMANHO DA TELA OU COM AS CONFIGURAÇÕES DE DISPOSITIVO. SÃO ÚTEIS QUANDO VOCÊ QUER QUE ALGO TENHA UM TAMANHO ESPECÍFICO, MAS ELAS NÃO RESPONDEM BEM EM LAYOUTS RESPONSIVOS.

- **PT (PONTO) E PC (PICA):** MEDIDAS DE TIPOGRAFIA TRADICIONAL. EX: FONT-SIZE: 12PT;
- **IN (POLEGADA):** POUCO COMUM EM TELAS, MAIS PARA IMPRESSÃO. EX: WIDTH: 2IN;
- **CM, MM (CENTÍMETROS E MILÍMETROS):** USADAS MAIS EM IMPRESSÃO DO QUE EM TELAS. EX: WIDTH: 5CM;
- **PX (PIXEL):** UNIDADE MAIS COMUM. DEFINE O TAMANHO EM PIXELS. EX: WIDTH: 200PX;

MEDIDAS RELATIVAS

MEDIDAS RELATIVAS SE AJUSTAM DE ACORDO COM OUTROS ELEMENTOS NA PÁGINA, TORNANDO-AS IDEIAS PARA LAYOUTS RESPONSIVOS. ELAS DEPENDEM DO CONTEXTO, COMO O TAMANHO DA FONTE DO ELEMENTO PAI OU DA LARGURA DA VIEWPORT.

- **EM:** RELATIVA AO TAMANHO DA FONTE DO ELEMENTO PAI. EX: PADDING: 1.5EM;
- **REM:** RELATIVA AO TAMANHO DA FONTE RAIZ (0). MAIS CONSISTENTE EM LAYOUTS ESCALÁVEIS. EX: FONT-SIZE: 2REM;
- **%:** RELATIVA AO ELEMENTO PAI. EX: WIDTH: 50%;

PRINCÍPIOS DE TIPOGRAFIA NO DESIGN

- **LEGIBILIDADE:** ESCOLHER FONTES QUE SEJAM FÁCEIS DE LER, PRINCIPALMENTE EM TAMANHOS MENORES.
- **HIERARQUIA VISUAL:** USAR TAMANHOS, PESOS E CORES DIFERENTES PARA DIFERENCIAR TÍTULOS, SUBTÍTULOS E CORPO DO TEXTO.
- **CONSISTÊNCIA:** LIMITAR A QUANTIDADE DE FONTES E ESTILOS PARA MANTER A UNIDADE VISUAL.

TIPOGRAFIA NA WEB

NO DESENVOLVIMENTO WEB, O CSS OFERECE DIVERSAS PROPRIEDADES PARA MANIPULAR A TIPOGRAFIA:

- **FONT-FAMILY:** DEFINE A FONTE PRINCIPAL.
- **FONT-SIZE:** DEFINE O TAMANHO DO TEXTO.
- **FONT-WEIGHT:** DEFINE A ESPESSURA DA FONTE.
- **LINE-HEIGHT:** DEFINE A ALTURA DA LINHA.
- **LETTER-SPACING:** DEFINE O ESPAÇAMENTO ENTRE LETRAS.



ANOTAÇÕES



```
6    background-color: #f0f0f0;
7    border: 1px solid #ccc;
8    border-radius: 5px;
9    color: #333;
10   font-size: 14px;
11   padding: 10px;
12   width: 100px;
13 }
14 
15 .block-container {
16   margin: 7px;
17   padding: 5px;
18   background-color: rgba(0, 0, 0, 0.6);
19   border-radius: 4px;
20   color: $blank;
21   height: 90%;
22   height: calc(100% - 24px);
23   position: relative;
24   width: 85px;
25   height: 85px;
26   margin-left: calc(50% - 42.5px);
27 }
28 
29 h4.service-title {
30   font-size: 24px;
31   text-align: center;
32   margin: 0;
33 }
34 
35 .service-text {
36   font-size: 16px;
37   margin-top: 40px 0;
```

A CIÊNCIA DO CSS

O CSS (CASCADING STYLE SHEETS) É UMA DAS TECNOLOGIAS MAIS IMPORTANTES NO DESENVOLVIMENTO WEB MODERNO. ELE É RESPONSÁVEL PELA ESTILIZAÇÃO E APRESENTAÇÃO DE PÁGINAS, PERMITINDO CRIAR DESIGNS VISUALMENTE ATRAENTES E FUNCIONAIS PARA A MELHOR EXPERIÊNCIA DO USUÁRIO.

COM O CSS, É POSSÍVEL CONTROLAR CORES, FONTES, ESPAÇAMENTOS, LAYOUTS E ATÉ ANIMAÇÕES, TRANSFORMANDO PÁGINAS SIMPLES EM INTERFACES DINÂMICAS E PROFISSIONAIS. ELE TRABALHA EM CONJUNTO COM O HTML PARA ESTRUTURAR E ESTILIZAR O CONTEÚDO, DESEMPENHANDO UM PAPEL ESSENCIAL NA CRIAÇÃO DE WEBSITES RESPONSIVOS E ACESSÍVEIS.

ALÉM DE SUA SIMPLICIDADE, O CSS OFERECE RECURSOS AVANÇADOS, COMO FLEXBOX, GRID E MEDIA QUERIES, QUE PERMITEM DESENVOLVER DESIGNS ADAPTÁVEIS A DIFERENTES DISPOSITIVOS E TAMANHOS DE TELA. ESSES RECURSOS TORNARAM-SE INDISPENSÁVEIS EM UM MUNDO ONDE A NAVEGAÇÃO MÓVEL CRESCE CONTINUAMENTE.

O CONHECIMENTO EM CSS É UM PONTO DE PARTIDA FUNDAMENTAL PARA QUALQUER PESSOA QUE DESEJA INGRESSAR NO MUNDO DO DESENVOLVIMENTO WEB. ELE É A CHAVE PARA CRIAR EXPERIÊNCIAS DIGITAIS MEMORÁVEIS, COMBINANDO FUNCIONALIDADE COM ESTÉTICA DE FORMA HARMÔNICA.



CSS: A LINGUAGEM DO ESTILO

O CSS (CASCADING STYLE SHEETS) É A TECNOLOGIA RESPONSÁVEL POR DAR VIDA AO DESIGN E À APRESENTAÇÃO DE PÁGINAS DA WEB. ENQUANTO O HTML FORNECE A ESTRUTURA DO CONTEÚDO, O CSS É USADO PARA ESTILIZAR E ORGANIZAR ESSA ESTRUTURA, PERMITINDO CONTROLAR CORES, FONTES, MARGENS, ESPAÇAMENTOS, POSICIONAMENTOS E MUITO MAIS.

INTRODUZIDO OFICIALMENTE EM 1996, O CSS TRANSFORMOU A MANEIRA COMO OS SITES ERAM PROJETADOS, PERMITINDO QUE DESIGNERS E DESENVOLVEDORES CRIASSEM INTERFACES VISUAIS ATRAENTES E CONSISTENTES, SEPARANDO O CONTEÚDO (HTML) DO DESIGN (CSS).

ATRAVÉS DE SELETORES E PROPRIEDADES, O CSS APLICA ESTILOS DIRETAMENTE AOS ELEMENTOS HTML, SEJA DE FORMA SIMPLES, COMO ALTERAR A COR DE UM TEXTO, OU COMPLEXA, CRIANDO ANIMAÇÕES E LAYOUTS RESPONSIVOS. ALÉM DISSO, É ESSENCIAL PARA CRIAR EXPERIÊNCIAS OTIMIZADAS EM DISPOSITIVOS MÓVEIS E DESKTOPS.

COM O CSS, A WEB DEIXOU DE SER APENAS FUNCIONAL PARA TAMBÉM SER VISUALMENTE RICA, CONECTANDO USUÁRIOS A DESIGNS QUE ENCANTAM E PROMOVEM USABILIDADE. É UMA FERRAMENTA INDISPENSÁVEL PARA QUALQUER PESSOA INTERESSADA EM DESENVOLVIMENTO WEB E DESIGN.



SINTAXE DO CSS

ESTRUTURA BÁSICA

**SELETOR {
PROPRIEDADE: VALOR;
}**

- SELETOR:** DEFINE O(S) ELEMENTO(S) HTML QUE SERÁ(ÃO) ESTILIZADO(S).
- PROPRIEDADE:** ESPECIFICA O ESTILO A SER APLICADO (COMO COR, TAMANHO, FONTE, ETC.).
- VALOR:** O VALOR ASSOCIADO À PROPRIEDADE, QUE DEFINE O ESTILO EXATO (POR EXEMPLO, COR OU TAMANHO).

EXEMPLO PRÁTICO:

```
H1 {  
    COLOR: BLUE;  
    FONT-SIZE: 24px;  
    TEXT-ALIGN: CENTER;  
}
```

EXPLICAÇÃO:

- H1:** O SELETOR AQUI É O ELEMENTO <h1>.
- COLOR: BLUE:** A COR DO TEXTO É DEFINIDA COMO AZUL.
- FONT-SIZE: 24px:** O TAMANHO DA FONTE É 24 PIXELS.
- TEXT-ALIGN: CENTER:** O TEXTO SERÁ CENTRALIZADO.

SINTAXE COM SELETORES DE CLASSE E ID

- **SELETORES DE CLASSE:** USAM UM PONTO ANTES DO NOME DA CLASSE (.)
- **SELETORES DE ID:** USAM UM JOGO DA VELHA (#) ANTES DO NOME DO ID.

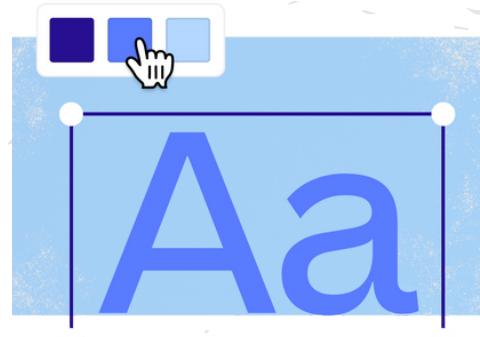
```
1. .BUTTON {  
2.   BACKGROUND-COLOR: RED;  
3.   PADDING: 10PX 20PX;  
4. }  
5.  
6. #HEADER {  
7.   BACKGROUND-COLOR: BLACK;  
8.   COLOR: WHITE;  
9. }
```

- **.BUTTON:** SELETOR DE CLASSE, ESTILIZANDO TODOS OS ELEMENTOS COM A CLASSE "BUTTON".
- **#HEADER:** SELETOR DE ID, ESTILIZANDO O ELEMENTO COM O ID "HEADER".

COMENTÁRIOS NO CSS

- OS COMENTÁRIOS SÃO ÚTEIS PARA ADICIONAR ANOTAÇÕES E EXPLICAÇÕES NO CÓDIGO. ELES SÃO IGNORADOS PELO NAVEGADOR E SÃO ESCRITOS ENTRE /* E */.

```
1. /* ESTE É UM COMENTÁRIO */  
2. H2 {  
3.   COLOR: GREEN; /* MUDANDO A COR PARA VERDE */  
4. }
```



CORES

CORES HEXADECIMais

CORES PODEM SER REPRESENTADAS POR UM CÓDIGO HEXADECIMAL. O FORMATO É #RRGGBB, ONDE RR É O VALOR HEXADECIMAL PARA O VERMELHO, GG É O VALOR PARA O VERDE E BB É O VALOR PARA O AZUL.

EXEMPLO:

```
1. H1 {  
2.   COLOR: #FF5733; /* UM TOM DE LARANJA */  
3. }
```

RGB (RED, GREEN, BLUE)

O MODELO DE CORES RGB PERMITE QUE VOCÊ DEFINA AS CORES COM VALORES NUMÉRICOS PARA VERMELHO, VERDE E AZUL (DE 0 A 255).

EXEMPLO:

```
1. H1 {  
2.   COLOR: RGB(255, 87, 51); /* UM TOM DE LARANJA */  
3. }
```

HSLA (HUE, SATURATION, LIGHTNESS, ALPHA)

HSLA É UMA VARIAÇÃO DO HSL COM O COMPONENTE ALPHA PARA A TRANSPARÊNCIA, DA MÉSMA FORMA QUE O RGBA.

EXEMPLO:

```
1. H1 {  
2.   COLOR: HSLA(9, 100%, 60%, 0.7); /* UM TOM DE LARANJA COM 70% DE OPACIDADE */  
3. }
```

RGBA (RED, GREEN, BLUE, ALPHA)

RGBA É UMA VARIAÇÃO DO MODELO RGB, MAS COM UM VALOR ADICIONAL PARA A OPACIDADE (ALPHA). O VALOR ALPHA VARIA DE 0 (TOTALMENTE TRANSPARENTE) A 1 (TOTALMENTE OPACO).

EXEMPLO:

```
1. H1 {  
2.   COLOR: RGBA(255, 87, 51, 0.7); /* TOM DE LARANJA COM 70% DE OPACIDADE */  
3. }
```



HSL (HUE, SATURATION, LIGHTNESS)

HSL USA TRÊS COMPONENTES:

- **HUE** (MATIZ) É O ÂNGULO DE 0 A 360 GRAUS, REPRESENTANDO A COR NO CÍRCULO CROMÁTICO (POR EXEMPLO, 0° É VERMELHO, 120° É VERDE, 240° É AZUL).
- **SATURATION** (SATURAÇÃO) É A INTENSIDADE DA COR, DE 0% (SEM COR) A 100% (CORES VIVAS).
- **LIGHTNESS** (LUMINOSIDADE) É A QUANTIDADE DE BRANCO OU PRETO NA COR, DE 0% (PRETO) A 100% (BRANCO).

EXEMPLO:

```
1. H1 {  
2.   COLOR: HSL(9, 100%, 60%); /* UM TOM DE LARANJA */  
3. }
```

CORES PERSONALIZADAS COM VARIÁVEIS (CSS VARIABLES)

VOCÊ PODE CRIAR E REUTILIZAR CORES PERSONALIZADAS COM VARIÁVEIS CSS:

EXEMPLO:

```
1. :ROOT {  
2.   --PRIMARY-COLOR: #3498DB;  
3. }  
4.  
5. H1 {  
6.   COLOR: VAR(--PRIMARY-COLOR);  
7. }
```

COM ESSA ABORDAGEM, VOCÊ PODE ALTERAR A COR EM UM ÚNICO LUGAR E TODAS AS REFERÊNCIAS A --PRIMARY-COLOR SERÃO ATUALIZADAS AUTOMATICAMENTE.

CORES COM OPACIDADE (TRANSPARENT)

VOCÊ TAMBÉM PODE USAR TRANSPARENT PARA UM FUNDO OU COR TOTALMENTE TRANSPARENTE.

EXEMPLO:

```
1. H1 {  
2.   COLOR: TRANSPARENT;  
3. }
```

CORES DE GRADIENTE

VOCÊ PODE CRIAR GRADIENTES (TRANSIÇÕES DE CORES) COM A PROPRIEDADE BACKGROUND OU BACKGROUND-IMAGE.

EXEMPLO:

```
1. H1 {  
2.   BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED, YELLOW);  
3. }
```

NESSE CASO, O FUNDO DO TÍTULO SERÁ UM GRADIENTE indo do VERMELHO PARA O AMARELO, DA ESQUERDA PARA A DIREITA.



FUNDOS

COR DE FUNDO

DEFINE UMA COR SÓLIDA COMO FUNDO.

EXEMPLO:

```
1. BODY {  
2.   BACKGROUND-COLOR: #F0F0F0; /* CINZA CLARO */  
3. }
```

IMAGEM DE FUNDO

DEFINE UMA IMAGEM COMO FUNDO.

EXEMPLO:

```
1. BODY {  
2.   BACKGROUND-IMAGE: URL('IMAGEM.JPG');  
3. }
```

- USE BACKGROUND-REPEAT PARA CONTROLAR A REPETIÇÃO:

EXEMPLO:

```
1. BACKGROUND-REPEAT: NO-REPEAT; /* A IMAGEM NÃO SE REPETIRÁ */
   */

```

- USE BACKGROUND-SIZE PARA AJUSTAR O TAMANHO:

EXEMPLO:

```
1. BACKGROUND-SIZE: COVER; /* FAZ COM QUE A IMAGEM CUBRA
   TODO O ELEMENTO */
2. BACKGROUND-SIZE: CONTAIN; /* AJUSTA A IMAGEM PARA CABER
   NO ELEMENTO */
```

FUNDO TRANSPARENTE

VOCÊ PODE USAR TRANSPARÊNCIA NO FUNDO COM RGBA OU HSLA.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND-COLOR: RGBA(0, 0, 0, 0.5); /* PRETO COM 50% DE
   TRANSPARÊNCIA */
3. }
```

FUNDO COM VÁRIOS ELEMENTOS

VOCÊ PODE COMBINAR VÁRIAS CAMADAS DE IMAGENS, CORES E GRADIENTES.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND:
3.   URL('IMAGEM.JPG') NO-REPEAT CENTER,
4.   LINEAR-GRADIENT(TO BOTTOM, RGBA(0, 0, 0, 0.5), RGBA(255, 255,
   255, 0.5));
5.   BACKGROUND-SIZE: COVER;
6. }
```

POSIÇÃO DO FUNDO

CONTROLA ONDE O FUNDO APARECE NO ELEMENTO.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND-POSITION: CENTER; /* CENTRALIZA O FUNDO */
3.   BACKGROUND-POSITION: TOP LEFT; /* POSICIONA NO CANTO
   SUPERIOR ESQUERDO */ }
```

GRADIENTES DE FUNDO

GRADIENTE LINEAR

- A TRANSIÇÃO OCORRE EM LINHA RETA.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED, YELLOW); /**
   VERMELHO À ESQUERDA E AMARELO À DIREITA */ }
```

GRADIENTE RADIAL

- A TRANSIÇÃO OCORRE A PARTIR DE UM PONTO CENTRAL.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND: RADIAL-GRADIENT(CIRCLE, BLUE, WHITE); /* AZUL
   NO CENTRO, BRANCO NAS BORDAS */ }
```

GRADIENTE CÔNICO

- A TRANSIÇÃO OCORRE EM TORNO DE UM PONTO CENTRAL.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND: CONIC-GRADIENT(FROM ODEG, RED, YELLOW, GREEN,
   RED); /* FUNDO EM FORMATO DE RODA DE CORES */ }
```

FIXAR O FUNDO

DEFINE SE O FUNDO SE MOVE COM O CONTEÚDO OU PERMANECE FIXO.

EXEMPLO:

```
1. BODY {
2.   BACKGROUND-ATTACHMENT: FIXED; /* O FUNDO FICARÁ FIXO NA
   TELA */
3. }
```

BORDAS DO FUNDO

O FUNDO PODE SER LIMITADO À BORDA INTERNA DO ELEMENTO USANDO BACKGROUND-CLIP.

EXEMPLO:

```
1. DIV {
2.   BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED, BLUE);
3.   BACKGROUND-CLIP: TEXT; /* APlica o fundo apenas no texto
   */
4.   -WEBKIT-BACKGROUND-CLIP: TEXT;
5.   COLOR: TRANSPARENT;
6. }
```



BORDAS

COMPONENTES DA BORDA

ESPESSURA DA BORDA

DEFINA A LARGURA DA BORDA.

EXEMPLO:

```
1. DIV {  
2.   BORDER-WIDTH: 5PX; /* BORDA COM 5PX */  
3. }
```

ESTILO DA BORDA

CONTROLE O ESTILO DA BORDA. VALORES COMUNS:

- **SOLID:** BORDA SÓLIDA.
- **DASHED:** TRAÇOS CURTOS.
- **DOTTED:** PONTILHADA.
- **DOUBLE:** DUAS LINHAS.
- **GROOVE:** BORDAS EM RELEVO.
- **RIDGE:** BORDAS EM ALTO RELEVO.
- **INSET:** PARECEM EMBUTIDOS.
- **OUTSET:** PARECEM SALTADAS.

EXEMPLO:

```
1. DIV {  
2.   BORDER-STYLE: DASHED; /* BORDA COM TRAÇOS */  
3. }
```

COR DA BORDA

DEFINA A COR DA BORDA.

EXEMPLO:

```
1. DIV {  
2.   BORDER-COLOR: RED; /* BORDA VERMELHA */  
3. }
```

BORDAS COM GRADIENTE

CREIE BORDAS COM GRADIENTES USANDO BORDER-IMAGE.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 5PX SOLID;  
3.   BORDER-IMAGE: LINEAR-GRADIENT(TO RIGHT, RED, YELLOW) 1; /*  
   GRADIENTE HORIZONTAL */  
4. }
```

BORDAS INDIVIDUAIS

VOÇÊ PODE DEFINIR BORDAS PARA LADOS ESPECÍFICOS:

EXEMPLO:

```
1. DIV {  
2.   BORDER-TOP: 3PX SOLID BLUE; /* BORDA SUPERIOR AZUL */  
3.   BORDER-RIGHT: 2PX DOTTED RED; /* BORDA DIREITA PONTILHADA  
   VERMELHA */  
4.   BORDER-BOTTOM: 4PX DOUBLE GREEN; /* BORDA INFERIOR DUPLA  
   VERDE */  
5.   BORDER-LEFT: 1PX DASHED BLACK; /* BORDA ESQUERDA  
   TRACEJADA PRETA */  
6. }
```

BORDAS ARREDONDADAS

PERMITE CRIAR BORDAS ARREDONDADAS. VALORES EM PXOU %.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 2PX SOLID #000;  
3.   BORDER-RADIUS: 10PX; /* BORDAS ARREDONDADAS DE 10PX */  
4. }
```

CIRCULAR DE BORDA

PARA BORDAS TOTALMENTE CIRCULARES, USE BORDER-RADIUS: 50%;

EXEMPLO:

```
1. DIV {  
2.   BORDER: 2PX SOLID BLUE;  
3.   BORDER-RADIUS: 50%; /* ELEMENTO REDONDO */  
4. }
```

BORDAS ESPECÍFICAS

VOÇÊ PODE ARREDONDAR APENAS ALGUNS CANTOS:

EXEMPLO:

```
1. DIV {  
2.   BORDER-TOP-LEFT-RADIUS: 20PX;  
3.   BORDER-TOP-RIGHT-RADIUS: 10PX;  
4.   BORDER-BOTTOM-RIGHT-RADIUS: 5PX;  
5.   BORDER-BOTTOM-LEFT-RADIUS: 0;  
6. }
```

BORDAS INVISÍVEIS

DEFINA O ESTILO COMO NONE PARA OCULTAR BORDAS.

EXEMPLO:

```
1. DIV {  
2.   BORDER: none; /* SEM BORDAS */  
3. }
```

BORDAS E SOMBRA

COMBINE BORDAS COM BOX-SHADOW PARA CRIAR EFEITOS VISUAIS.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 3px solid #333;  
3.   BOX-SHADOW: 0px 4px 8px rgba(0, 0, 0, 0.2); /* SOMBRA  
4.   EXTERNA */  
4. }
```

BORDAS DUPLAS OU COM IMAGENS

BORDA DUPLA

USAR BORDER-STYLE: DOUBLE.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 5px double blue;  
3. }
```

BORDA COM IMAGEM

ADICIONE UMA IMAGEM PARA FORMAR A BORDA.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 10px solid;  
3.   BORDER-IMAGE: url('BORDA.PNG') 30 round; /* IMAGEM COMO  
4.   BORDA */  
4. }
```

BORDAS COM ANIMAÇÕES

USE ANIMAÇÕES PARA CRIAR BORDAS DINÂMICAS.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 2px solid transparent;  
3.   BORDER-RADIUS: 15px;  
4.   ANIMATION: BORDERGLOW 2s infinite;  
5. }  
6.  
7. @KEYFRAMES BORDERGLOW {  
8.   0% {  
9.     BORDER-COLOR: red;  
10. }  
11. 50% {  
12.     BORDER-COLOR: blue;  
13. }  
14. 100% {  
15.     BORDER-COLOR: red;  
16. }  
17. }
```

BORDAS INTERNAS

USE OUTLINE PARA CRIAR BORDAS INTERNAS QUE NÃO AFETEM O LAYOUT.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE: 3px dotted orange;  
3. }
```





MARGENS

MARGENS COLAPSADAS

AS MARGENS VERTICais DE DOIS ELEMENTOS ADJACENTES PODEM "COLAPSAR" EM UM ÚNICO VALOR (O MAIOR DAS DUAS). ISSO OCORRE APENAS COM MARGIN-TOP E MARGIN-BOTTOM.

EXEMPLO:

```
1. DIV1 {  
2.   MARGIN-BOTTOM: 20px;  
3. }  
4.  
5. DIV2 {  
6.   MARGIN-TOP: 30px;  
7. }  
8.  
9./* A MARGEM TOTAL SERÁ DE 30px, NÃO 50px */
```

LAYOUTS RESPONSIVOS

USE MARGIN EM CONJUNTO COM UNIDADES FLEXÍVEIS, COMO % OU VW, PARA CRIAR DESIGNS RESPONSIVOS.

EXEMPLO:

```
1. DIV {  
2.   MARGIN: 5%; /* 5% DO TAMANHO DO CONTÊINER */  
3. }
```

VALORES NEGATIVOS

MARGENS PODEM TER VALORES NEGATIVOS PARA CRIAR SOBREPOSIÇÕES.

EXEMPLO:

```
1. DIV {  
2.   MARGIN: -10px; /* MOVE O ELEMENTO 10px MAIS PRÓXIMO DE  
3.   OUTROS ELEMENTOS */  
4. }
```

INLINE E BLOCK

- EM ELEMENTOS BLOCK (COMO <DIV>), AS MARGENS AFETAM OS QUATRO LADOS.
- EM ELEMENTOS INLINE (COMO), APENAS AS MARGENS HORIZONTAIS (ESQUERDA E DIREITA) AFETAM O LAYOUT.

EXEMPLO:

```
1. SPAN {  
2.   MARGIN-LEFT: 10px;  
3.   MARGIN-RIGHT: 10px;  
4. }
```

MARGEM AUTOMÁTICA

A PROPRIEDADE MARGIN: AUTO É USADA PRINCIPALMENTE PARA CENTRALIZAR ELEMENTOS HORIZONTALMENTE DENTRO DE UM CONTÊINER.

EXEMPLO:

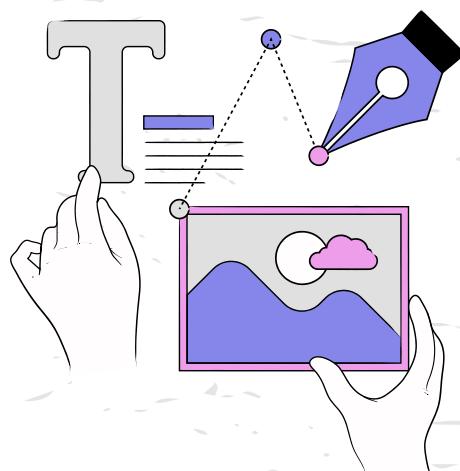
```
1. DIV {  
2.   WIDTH: 50%; /* DEFINE A LARGURA */  
3.   MARGIN: 0 auto; /* CENTRALIZA HORIZONTALMENTE */  
4. }
```

MARGEM COM FLEXBOX

QUANDO SE TRABALHA COM FLEXBOX, MARGENS PODEM SER USADAS PARA AJUSTAR ESPAÇOS ENTRE OS ITENS.

EXEMPLO:

```
1. CONTAINER {  
2.   DISPLAY: flex;  
3.   JUSTIFY-CONTENT: space-between; /* ESPAÇO ENTRE ITENS */  
4. }  
5.  
6. ITEM {  
7.   MARGIN: 0 10px; /* ESPAÇO HORIZONTAL ENTRE ITENS */  
8. }
```



MARGENS INDIVIDUAIS

VOCÊ PODE DEFINIR MARGENS ESPECÍFICAS PARA CADA LADO:

- **MARGIN-TOP:** MARGEM SUPERIOR.
- **MARGIN-RIGHT:** MARGEM DIREITA.
- **MARGIN-BOTTOM:** MARGEM INFERIOR.
- **MARGIN-LEFT:** MARGEM ESQUERDA.

EXEMPLO:

```
1. DIV {  
2.   MARGIN-TOP: 10px; /* MARGEM SUPERIOR DE 10px */  
3.   MARGIN-RIGHT: 15px; /* MARGEM DIREITA DE 15px */  
4.   MARGIN-BOTTOM: 20px; /* MARGEM INFERIOR DE 20px */  
5.   MARGIN-LEFT: 25px; /* MARGEM ESQUERDA DE 25px */  
6. }
```

VALORES COMBINADOS

QUATRO VALORES

```
1. DIV {  
2.   MARGIN: 10px 15px 20px 25px; /* TOPO, DIREITA, BAIXO,  
3.     ESQUERDA */  
4. }
```

TRÊS VALORES

```
1. DIV {  
2.   MARGIN: 10px 15px 20px; /* TOPO, LATERAIS (DIREITA E  
3.     ESQUERDA), BAIXO */  
4. }
```

DOIS VALORES

```
1. DIV {  
2.   MARGIN: 10px 15px; /* TOPO E BAIXO, LATERAIS */  
3. }
```

UM VALOR

```
1. DIV {  
2.   MARGIN: 10px; /* MARGEM IGUAL EM TODOS OS LADOS */  
3. }
```

PREENCHIMENTO

MARGENS INDIVIDUAIS

VOCÊ PODE DEFINIR O ESPAÇAMENTO INTERNO PARA CADA LADO SEPARADAMENTE:

- **PADDING-TOP:** ESPAÇAMENTO NA PARTE SUPERIOR.
- **PADDING-RIGHT:** ESPAÇAMENTO NA DIREITA.
- **PADDING-BOTTOM:** ESPAÇAMENTO NA PARTE INFERIOR.
- **PADDING-LEFT:** ESPAÇAMENTO NA ESQUERDA.

EXEMPLO:

```
1. DIV {  
2.   PADDING-TOP: 10px; /* ESPAÇAMENTO SUPERIOR */  
3.   PADDING-RIGHT: 15px; /* ESPAÇAMENTO DIREITO */  
4.   PADDING-BOTTOM: 20px; /* ESPAÇAMENTO INFERIOR */  
5.   PADDING-LEFT: 25px; /* ESPAÇAMENTO ESQUERDO */  
6. }
```

UNIDADES DE MEDIDA

O PADDING ACEITA DIFERENTES UNIDADES:

- **PIXELS (PX):** VALOR FIXO.
- **PORCENTAGEM (%):** RELATIVO À LARGURA DO ELEMENTO PAI.
- **EM/REM:** RELATIVO AO TAMANHO DA FONTE.
- **VIEWPORT WIDTH/HEIGHT (VW/VH):** RELATIVO AO TAMANHO DA JANELA DO NAVEGADOR.

EXEMPLO:

```
1. DIV {  
2.   PADDING: 10%; /* 10% DA LARGURA DO ELEMENTO PAI */  
3. }
```



VALORES COMBINADOS

QUATRO VALORES

```
1. DIV {  
2.   PADDING: 10px 15px 20px 25px; /* topo, direita, baixo,  
   esquerda */  
3. }
```

TRÊS VALORES

```
1. DIV {  
2.   PADDING: 10px 15px 20px; /* topo, laterais (direita e  
   esquerda), baixo */  
3. }
```

DOIS VALORES

```
1. DIV {  
2.   PADDING: 10px 15px; /* topo e baixo, laterais */  
3. }
```

UM VALOR

```
1. DIV {  
2.   PADDING: 10px; /* aplica o mesmo espaçamento em todos os  
   lados */  
3. }
```

BORDAS E BACKGROUND

O padding afeta diretamente o espaço onde o fundo e as bordas de um elemento são aplicados.

EXEMPLO:

```
1. DIV {  
2.   PADDING: 20px;  
3.   BORDER: 2px solid #000;  
4.   BACKGROUND-COLOR: #f4f4f4;  
5. }
```

BOX MODEL E PADDING

No modelo de caixa padrão, o padding aumenta o tamanho total de um elemento. Porém, ao usar box-sizing: border-box;, o padding é incluído dentro do tamanho total.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 100px;  
3.   HEIGHT: 100px;  
4.   PADDING: 20px; /* adiciona 20px a cada lado */  
5. }  
6.  
7. /* com box-sizing */  
8. DIV {  
9.   BOX-SIZING: BORDER-BOX;  
10.  WIDTH: 100px;  
11.  PADDING: 20px; /* espaçamento interno já incluso nos  
   100px */  
12. }
```

ALTURA E LARGURA

PROPRIEDADES BÁSICAS

- **width:** Define a largura de um elemento.
- **height:** Define a altura de um elemento.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 300px; /* define a largura como 300px */  
3.   HEIGHT: 150px; /* define a altura como 150px */  
4. }
```

UNIDADES DE MEDIDA

Você pode usar diferentes unidades para definir altura e largura:

- **pixels (px):** Um valor fixo e absoluto.
- **porcentagem (%):** Relativo ao tamanho do elemento pai.
- **em/rem:** Relativo ao tamanho da fonte.
- **viewports (vw/vh):** Relativo à largura ou altura da janela do navegador.
- **auto:** Calcula automaticamente com base no conteúdo.

ALTURA E LARGURA MÍNIMA/MÁXIMA

PARA LIMITAR O TAMANHO DE UM ELEMENTO:

- MIN-WIDTH E MAX-WIDTH: DEFINE A LARGURA MÍNIMA E MÁXIMA.
- MIN-HEIGHT E MAX-HEIGHT: DEFINE A ALTURA MÍNIMA E MÁXIMA.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 80%; /* LARGURA PADRÃO */  
3.   MAX-WIDTH: 500PX; /* LARGURA MÁXIMA */  
4.   MIN-WIDTH: 200PX; /* LARGURA MÍNIMA */  
5.  
6.   HEIGHT: 50VH; /* ALTURA PADRÃO */  
7.   MAX-HEIGHT: 400PH; /* ALTURA MÁXIMA */  
8.   MIN-HEIGHT: 100PH; /* ALTURA MÍNIMA */  
9. }
```

VIEWPORT (VW E VH)

- VW: 1 UNIDADE EQUIVALE A 1% DA LARGURA DA JANELA.
- VH: 1 UNIDADE EQUIVALE A 1% DA ALTURA DA JANELA.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 100VW; /* OCUPA TODA A LARGURA DA JANELA */  
3.   HEIGHT: 100VH; /* OCUPA TODA A ALTURA DA JANELA */  
4. }
```

ALTURA E LARGURA AUTOMÁTICA

- WIDTH: AUTO; O ELEMENTO OCUPA TODO O ESPAÇO DISPONÍVEL HORIZONTALMENTE.
- HEIGHT: AUTO; A ALTURA É AJUSTADA AUTOMATICAMENTE COM BASE NO CONTEÚDO.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: AUTO;  
3.   HEIGHT: AUTO;  
4. }
```

BOX MODEL

ESTRUTURA DO BOX MODEL

- **CONTENT (CONTEÚDO):** É A ÁREA ONDE O TEXTO, AS IMAGENS OU OUTROS ELEMENTOS SÃO EXIBIDOS. O TAMANHO DO CONTEÚDO É DEFINIDO PELAS PROPRIEDADES WIDTH E HEIGHT.
- **PADDING (PREENCHIMENTO INTERNO):** ESPAÇO ENTRE O CONTEÚDO E A BORDA. CRIA UM AFASTAMENTO INTERNO PARA QUE O CONTEÚDO NÃO TOQUE DIRETAMENTE NAS BORDAS.
- **BORDER (BORDA):** A BORDA ENVOLVE O CONTEÚDO E O PREENCHIMENTO (PADDING). ELA PODE SER CONFIGURADA COM PROPRIEDADES COMO LARGURA, ESTILO E COR.
- **MARGIN (MARGEM):** ESPAÇO ENTRE O ELEMENTO E OUTROS ELEMENTOS AO REDOR. É USADO PARA CRIAR AFASTAMENTO EXTERNO.

PROPRIEDADE BOX-SIZING

POR PADRÃO, O TAMANHO DE UM ELEMENTO NO CSS É CALCULADO APENAS COM BASE NO WIDTH E HEIGHT DO CONTENT. ISSO PODE CAUSAR PROBLEMAS AO ADICIONAR PADDING E BORDER, JÁ QUE ELES NÃO SÃO INCLUÍDOS NO TAMANHO TOTAL.

- **BOX-SIZING: CONTENT-BOX (PADRÃO):** APENAS O CONTEÚDO É INCLUÍDO NO CÁLCULO DO TAMANHO. O PADDING E A BORDA SÃO ADICIONADOS EXTERNAMENTE, AUMENTANDO O TAMANHO TOTAL.
- **BOX-SIZING: BORDER-BOX:** INCLUI O CONTEÚDO, PADDING E BORDA NO CÁLCULO DO TAMANHO TOTAL DO ELEMENTO. É MAIS PREVISÍVEL E RECOMENDADO PARA LAYOUTS MODERNOS.



PROPRIEDADES DO BOX MODEL

1. CONTENT (WIDTH E HEIGHT)

- DEFINE O TAMANHO DA ÁREA DE CONTEÚDO.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 200px;  
3.   HEIGHT: 100px;  
4. }
```

2. PADDING

- CRIA ESPAÇAMENTO INTERNO ENTRE O CONTEÚDO E A BORDA.

EXEMPLO:

```
1. DIV {  
2.   PADDING: 20px; /* 20px EM TODOS OS LADOS */  
3.   PADDING-TOP: 10px; /* APENAS NO TOPO */  
4.   PADDING-RIGHT: 15px; /* APENAS À DIREITA */  
5.   PADDING-BOTTOM: 20px; /* APENAS NA PARTE INFERIOR */  
6.   PADDING-LEFT: 10px; /* APENAS À ESQUERDA */  
7. }
```

3. BORDER

- ADICIONA UMA BORDA AO REDOR DO ELEMENTO.

EXEMPLO:

```
1. DIV {  
2.   BORDER: 5px solid black; /* BORDA PRETA SÓLIDA DE 5px */  
3.   BORDER-RADIUS: 10px; /* BORDA ARREDONDADA */  
4. }
```

4. MARGIN

- CRIA ESPAÇAMENTO EXTERNO ENTRE O ELEMENTO E OUTROS AO REDOR.

EXEMPLO:

```
1. DIV {  
2.   MARGIN: 20px; /* 20px EM TODOS OS LADOS */  
3.   MARGIN-TOP: 10px; /* APENAS NO TOPO */  
4.   MARGIN-RIGHT: 15px; /* APENAS À DIREITA */  
5.   MARGIN-BOTTOM: 20px; /* APENAS NA PARTE INFERIOR */  
6.   MARGIN-LEFT: 10px; /* APENAS À ESQUERDA */  
7. }
```

OUTLINE

O OUTLINE NO CSS É UMA LINHA DESENHADA AO REDOR DE UM ELEMENTO FORA DE SUA BORDA (BORDER), USADA PRINCIPALMENTE PARA DESTACAR O ELEMENTO VISUALMENTE. ELE É SEMELHANTE À PROPRIEDADE BORDER, MAS NÃO AFETA O FLUXO DO LAYOUT E NÃO OCUPA ESPAÇO ADICIONAL.

CARACTERÍSTICAS DO OUTLINE

1. O OUTLINE NÃO OCUPA ESPAÇO, OU SEJA, NÃO "EMPURRA" OUTROS ELEMENTOS NA PÁGINA.
2. É GERALMENTE USADO PARA ACESSIBILIDADE, COMO FOCO EM CAMPOS DE FORMULÁRIO OU ELEMENTOS INTERATIVOS.
3. A PROPRIEDADE OUTLINE PODE SER CONFIGURADA COM ESTILO, LARGURA E COR, DE MANEIRA SEMELHANTE À PROPRIEDADE BORDER.

PROPRIEDADES DO OUTLINE

1. OUTLINE-STYLE

DEFINE O ESTILO DA LINHA DO OUTLINE.

VALORES COMUNS:

- **SOLID**: LINHA CONTÍNUA.
- **DOTTED**: LINHA PONTILHADA.
- **DASHED**: LINHA TRACEJADA.
- **DOUBLE**: DUAS LINHAS PARALELAS.
- **GROOVE**: LINHA COM APARÊNCIA DE SULCO.
- **RIDGE**: LINHA COM APARÊNCIA DE ELEVAÇÃO.
- **NONE**: NENHUM OUTLINE.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE-STYLE: SOLID;  
3. }
```

2. OUTLINE-WIDTH

DEFINE A LARGURA DA LINHA DO OUTLINE.

VALORES:

- **THIN, MEDIUM, THICK**: VALORES PREDEFINIDOS.
- **PX, EM, REM**: VALORES ESPECÍFICOS.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE-WIDTH: 2PX;  
3. }
```

3. OUTLINE-COLOR

DEFINE A COR DA LINHA DO OUTLINE.

VALORES:

- RED, BLUE, TRANSPARENT, ETC.
- TAMBÉM ACEITA VALORES HEXADECIMais, RGB E HSL.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE-COLOR: RED;  
3. }
```

4. OUTLINE

DEFINE O STYLE, WIDTH E COLOR EM UMA ÚNICA LINHA.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE: 3PX DOTTED BLUE;  
3. }
```

5. OUTLINE-OFFSET

DEFINE O DESLOCAMENTO ENTRE O OUTLINE E A BORDA (BORDER) DO ELEMENTO.

EXEMPLO:

```
1. DIV {  
2.   OUTLINE: 3PX SOLID GREEN;  
3.   OUTLINE-OFFSET: 10PX;  
4. }
```

TEXTO

1. COR DO TEXTO (COLOR)

DEFINE A COR DO TEXTO:

EXEMPLO:

```
1. P {  
2.   COLOR: BLUE;  
3. }
```

2. ALINHAMENTO DO TEXTO (TEXT-ALIGN)

CONTROLA O ALINHAMENTO DO TEXTO DENTRO DE UM ELEMENTO:

- **LEFT** (ESQUERDA),
- **RIGHT** (DIREITA),
- **CENTER** (CENTRALIZADO),
- **JUSTIFY** (JUSTIFICADO).

EXEMPLO:

```
1. P {  
2.   TEXT-ALIGN: CENTER;  
3. }
```

3. DECORAÇÃO DO TEXTO (TEXT-DECORATION)

APLICA OU REMOVE EFEITOS COMO SUBLINHADO, LINHA SOBRE O TEXTO OU LINHA NO MEIO:

- **NONE** (SEM DECORAÇÃO),
- **UNDERLINE** (SUBLINHADO),
- **OVERLINE** (LINHA ACIMA DO TEXTO),
- **LINE-THROUGH** (RISCADO).

EXEMPLO:

```
1. A {  
2.   TEXT-DECORATION: UNDERLINE; }
```

4. TRANSFORMAÇÃO DO TEXTO (TEXT-TRANSFORM)

CONTROLA A CAPITALIZAÇÃO DO TEXTO:

- **UPPERCASE** (TUDO EM MAIÚSCULAS),
- **LOWERCASE** (TUDO EM MINÚSCULAS),
- **CAPITALIZE** (PRIMEIRA LETRA EM MAIÚSCULA DE CADA PALAVRA).

EXEMPLO:

```
1. H1 {  
2.   TEXT-TRANSFORM: UPPERCASE; }
```



5. ESPAÇAMENTO ENTRE LETRAS (LETTER-SPACING)

CONTROLA O ESPAÇO ENTRE AS LETRAS:

EXEMPLO:

```
1. p {  
2.   LETTER-SPACING: 2px;  
3. }
```

6. ESPAÇAMENTO ENTRE LINHAS (LINE-HEIGHT)

AJUSTA O ESPAÇO VERTICAL ENTRE AS LINHAS:

EXEMPLO:

```
1. p {  
2.   LINE-HEIGHT: 1.5;  
3. }
```

7. RECUO DO TEXTO (TEXT-INDENT)

DEFINE O RECUO DA PRIMEIRA LINHA DO TEXTO:

EXEMPLO:

```
1. p {  
2.   TEXT-INDENT: 50px;  
3. }
```

8. SOMBRA DO TEXTO (TEXT-SHADOW)

APLICA SOMBRAIS AO TEXTO:

EXEMPLO:

```
1. h1 {  
2.   TEXT-SHADOW: 2px 2px 5px gray;  
3. }
```

9. QUEBRA DE PALAVRAS (WORD-WRAP OU OVERFLOW-WRAP)

CONTROLA COMO AS PALAVRAS SE COMPORTAM EM LIMITES:

- **BREAK-WORD:** PERMITE A QUEBRA DE PALAVRAS LONGAS.

EXEMPLO:

```
1. p {  
2.   WORD-WRAP: break-word;  
3. }
```

FONTE

1. DEFININDO O TIPO DE FONTE (FONT-FAMILY)

ESPECIFICA AS FONTES A SEREM USADAS NO TEXTO. GERALMENTE, É RECOMENDÁVEL INCLUIR UMA FONTE GENÉRICA COMO FALLBACK:

EXEMPLO:

```
1. p {  
2.   FONT-FAMILY: 'arial', 'helvetica', sans-serif;  
3. }
```

2. TAMANHO DA FONTE (FONT-SIZE)

CONTROLA O TAMANHO DO TEXTO. PODE SER ESPECIFICADO EM:

- **PIXEIS (PX):** TAMANHO FIXO.
- **EM/REM:** TAMANHO RELATIVO.
- **PERCENTUAL (%):** RELATIVO AO ELEMENTO PAI.

EXEMPLO:

```
1. h1 {  
2.   FONT-SIZE: 24px;  
3. }  
4.  
5. p {  
6.   FONT-SIZE: 1.5em;  
7. }
```

3. ESTILO DA FONTE (FONT-STYLE)

DEFINE O ESTILO DO TEXTO:

- **NORMAL:** TEXTO PADRÃO.
- **ITALIC:** TEXTO EM ITÁLICO.
- **OBLIQUE:** TEXTO INCLINADO.

EXEMPLO:

```
1. p {  
2.   font-style: italic;  
3. }
```

4. PESO DA FONTE (font-weight)

CONTROLA A ESPESSURA DAS LETRAS:

- NORMAL: PESO PADRÃO.
- BOLD: NEGRITO.
- NÚMEROS DE 100 A 900: CONTROLE MAIS PRECISO DA ESPESSURA.

EXEMPLO:

```
1. h1 {  
2.   font-weight: bold;  
3. }  
4.  
5. p {  
6.   font-weight: 300; /* LEVE */  
7. }
```

5. ALTURA DA LINHA (line-height)

DEFINE O ESPAÇO ENTRE AS LINHAS DE TEXTO:

EXEMPLO:

```
1. p {  
2.   line-height: 1.6;  
3. }
```

6. VARIANTE DA FONTE (font-variant)

USADA PARA EXIBIR TEXTOS EM SMALL CAPS (LETROS MAIÚSCULAS PEQUENAS):

EXEMPLO:

```
1. p {  
2.   font-variant: small-caps;  
3. }
```

7. COMBINAÇÃO COM A PROPRIEDADE FONT

VOCÊ PODE DEFINIR VÁRIAS PROPRIEDADES DE FONTE EM UMA ÚNICA LINHA COM A SEGUINTE SINTAXE:

EXEMPLO:

```
1. p {  
2.   font: italic small-caps bold 16px/1.5 'arial', sans-serif;  
3. }
```

8. IMPORTANDO FONTES EXTERNAS

PARA USAR FONTES PERSONALIZADAS (COMO DO GOOGLE FONTS):

EXEMPLO:

```
1. @import url('https://fonts.googleapis.com/css2?  
family=roboto:wght@400;700&display=swap');  
2.  
3. body {  
4.   font-family: 'roboto', sans-serif;  
5. }
```

9. AJUSTANDO O ESPAÇAMENTO DAS LETRAS (letter-spacing)

CONTROLA O ESPAÇO ENTRE OS CARACTERES:

EXEMPLO:

```
1. p {  
2.   letter-spacing: 2px;  
3. }
```

10. AJUSTANDO O ESPAÇAMENTO DAS PALAVRAS (word-spacing)

CONTROLA O ESPAÇO ENTRE AS PALAVRAS:

EXEMPLO:

```
1. p {  
2.   word-spacing: 5px;  
3. }
```





LINKS

1. SELETOR DE LINKS

VOCÊ PODE ESTILIZAR LINKS COM DIFERENTES ESTADOS USANDO OS SELETORES DE PSEUDO-CLASSES:

- A: O LINK PADRÃO.
- A:LINK: UM LINK NÃO VISITADO.
- A:VISITED: UM LINK QUE JÁ FOI VISITADO.
- A:HOVER: O LINK QUANDO O CURSOR ESTÁ SOBRE ELE.
- A:ACTIVE: O LINK NO MOMENTO EM QUE É CLICADO.

EXEMPLO:

```
1. a {  
2.   TEXT-DECORATION: none; /* REMOVE O SUBLINHADO PADRÃO */  
3.   COLOR: blue; /* DEFINE A COR DO TEXTO */  
4. }  
5.  
6. a:hover {  
7.   TEXT-DECORATION: underline; /* ADICIONA SUBLINHADO AO PASSAR O MOUSE */  
8.   COLOR: red; /* MUDA A COR */  
9. }
```

2. ALTERANDO CORES

USE A PROPRIEDADE COLOR PARA ALTERAR A COR DOS LINKS EM DIFERENTES ESTADOS.

EXEMPLO:

```
1. a {  
2.   COLOR: #4CAF50; /* VERDE PARA LINKS NORMAIS */  
3. }  
4.  
5. a:visited {  
6.   COLOR: #9C27B0; /* ROXO PARA LINKS VISITADOS */  
7. }  
8.  
9. a:hover {  
10.  COLOR: #FF5722; /* LARANJA PARA O HOVER */  
11. }  
12.  
13. a:active {  
14.  COLOR: #F44336; /* VERMELHO AO CLICAR */  
15. }
```

3. REMOVER SUBLINHADO

A PROPRIEDADE TEXT-DECORATION É USADA PARA CONTROLAR O SUBLINHADO DO LINK.

EXEMPLO:

```
1. a {  
2.   TEXT-DECORATION: none;  
3. }
```

4. ADICIONANDO FUNDO E BORDAS

VOCÊ PODE USAR ESTILOS ADICIONAIS PARA DESTACAR OS LINKS.

EXEMPLO:

```
1. a {  
2.   BACKGROUND-COLOR: yellow; /* FUNDO AMARELO */  
3.   BORDER: 1px solid black; /* ADICIONA UMA BORDA */  
4.   PADDING: 5px; /* ADICIONA ESPAÇO INTERNO */  
5. }  
6.  
7. a:hover {  
8.   BACKGROUND-COLOR: orange; /* FUNDO LARANJA NO HOVER */  
9. }
```

5. EFEITO DE TRANSIÇÃO NO HOVER

ADICIONAR TRANSIÇÕES SUAVES PARA CRIAR UM EFEITO AO PASSAR O MOUSE:

EXEMPLO:

```
1. a {  
2.   COLOR: blue;  
3.   TEXT-DECORATION: none;  
4.   TRANSITION: color 0.3s ease;  
5. }  
6.  
7. a:hover {  
8.   COLOR: green;  
9. }
```

6. LINKS BOTÕES

TRANSFORME LINKS EM BOTÕES ESTILIZADOS:

EXEMPLO:

```
1. a {  
2.   DISPLAY: inline-block;  
3.   PADDING: 10px 20px;  
4.   BACKGROUND-COLOR: #007bff;  
5.   COLOR: white;  
6.   TEXT-DECORATION: none;  
7.   BORDER-RADIUS: 5px; /* BORDAS ARREDONDADAS */  
8. }  
9.  
10. a:hover {  
11.   BACKGROUND-COLOR: #0056b3; /* TON MAIS ESCURO NO HOVER */  
12. }
```

7. LINKS DESATIVADOS

USANDO A CLASSE DISABLED PARA DESATIVAR LINKS (NÃO CLICÁVEIS):

EXEMPLO:

```
1. a.disabled {  
2.   POINTER-EVENTS: none; /* DESABILITA O CLIQUE */  
3.   COLOR: gray; /* COR PARA INDICAR QUE ESTÁ DESABILITADO */  
4.   TEXT-DECORATION: none;  
5. }
```

8. LINKS COM ÍCONES

COMBINE LINKS COM ÍCONES PARA MELHORAR A APARÊNCIA (USANDO BIBLIOTECAS COMO FONT AWESOME OU REMIXICON):

EXEMPLO NO HTML:

```
1. <a href="#">  
2.   <i class="ri-home-line"></i> HOME  
3. </a>
```

EXEMPLO NO CSS:

```
1. a {  
2.   DISPLAY: flex;  
3.   ALIGN-ITEMS: center;  
4.   GAP: 8px; /* ESPAÇO ENTRE O TEXTO E O ÍCONE */  
5. }
```

9. LINKS COM SUBSTITUIÇÃO DE TEXTO NO HOVER

CRIE UM EFEITO ONDE O TEXTO MUDA AO PASSAR O MOUSE:

EXEMPLO:

```
1. a:hover::after {  
2.   CONTENT: " →"; /* ADICIONA UMA SETA AO LADO DO TEXTO */  
3. }
```

LISTAS

LIST-STYLE

2.1. TIPOS DE MARCADORES

USE LIST-STYLE-TYPE PARA DEFINIR O ESTILO:

EXEMPLO:

```
1. ul {  
2.   LIST-STYLE-TYPE: square; /* MARCADOR QUADRADO */  
3. }  
4.  
5. ol {  
6.   LIST-STYLE-TYPE: upper-roman; /* NÚMEROS ROMANOS MAIÚSCULOS */  
7. }
```

VALORES COMUNS:

- DISC (PADRÃO PARA ``)
- CIRCLE
- SQUARE
- NONE (REMOVE MARCADORES)
- DECIMAL (PADRÃO PARA ``)
- UPPER-ROMAN, LOWER-ROMAN
- UPPER-ALPHA, LOWER-ALPHA

2.2. IMAGEM COMO MARCADOR

USE LIST-STYLE-IMAGE PARA SUBSTITUIR O MARCADOR POR UMA IMAGEM PERSONALIZADA:



EXEMPLO:

```
1. UL {  
2.   LIST-STYLE-IMAGE: URL('BULLET.PNG'); /* CAMINHO DA IMAGEM */  
3. }
```

REMOVER ESPAÇAMENTO PADRÃO

LISTAS POSSUEM MARGENS E PREENCHIMENTOS PADRÕES. VOCÊ PODE REMOVÉ-LOS:

EXEMPLO:

```
1. UL, OL {  
2.   MARGIN: 0;  
3.   PADDING: 0;  
4.   LIST-STYLE: NONE; /* REMOVE OS MARCADORES */  
5. }
```

LISTAS HORIZONTAIS

TRANSFORME LISTAS VERTICIAIS EM HORIZONTAIS:

EXEMPLO:

```
1. UL {  
2.   DISPLAY: FLEX;  
3.   GAP: 10PX; /* ESPAÇAMENTO ENTRE OS ITENS */  
4.   LIST-STYLE: NONE; /* REMOVE OS MARCADORES */  
5. }  
6.  
7. LI {  
8.   PADDING: 10PX;  
9.   BACKGROUND-COLOR: #F0F0F0;  
10.  BORDER-RADIUS: 5PX;  
11. }
```

LISTAS COM ÍCONES

ADICIONE ÍCONES PERSONALIZADOS USANDO ::BEFORE:

EXEMPLO:

```
1. LI::BEFORE {  
2.   CONTENT: ":"; /* ADICIONA UM MARCADOR */  
3.   COLOR: RED; /* COR DO MARCADOR */  
4.   MARGIN-RIGHT: 8PX;  
5. }
```

ESTILIZANDO LISTAS ANIMADAS

EXEMPLO:

```
1. LI {  
2.   OPACITY: 0;  
3.   TRANSFORM: TRANSLATEX(-20PX);  
4.   ANIMATION: FADEIN 0.5S EASE FORWARDS;  
5. }  
6.  
7. LI:NTH-CHILD(1) { ANIMATION-DELAY: 0S; }  
8. LI:NTH-CHILD(2) { ANIMATION-DELAY: 0.2S; }  
9. LI:NTH-CHILD(3) { ANIMATION-DELAY: 0.4S; }  
10.  
11. @KEYFRAMES FADEIN {  
12.   TO {  
13.     OPACITY: 1;  
14.     TRANSFORM: TRANSLATEX(0);  
15.   }  
16. }
```

LISTAS EM MENUS

EXEMPLO NO HTML:

```
1. <UL CLASS="MENU">  
2.   <LI><A HREF="#">HOME</A></LI>  
3.   <LI><A HREF="#">SOBRE</A></LI>  
4.   <LI><A HREF="#">CONTATO</A></LI>  
5. </UL>
```

EXEMPLO NO CSS:

```
1. MENU {  
2.   DISPLAY: FLEX;  
3.   GAP: 15PX;  
4.   LIST-STYLE: NONE;  
5. }  
6.  
7. MENU A {  
8.   TEXT-DECORATION: NONE;  
9.   COLOR: #007BFF;  
10.  PADDING: 10PX 15PX;  
11.  BORDER-RADIUS: 5PX;  
12. }  
13.  
14. MENU A:HOVER {  
15.   BACKGROUND-COLOR: #007BFF;  
16.   COLOR: WHITE;  
17. }
```

ESTILIZANDO ITENS DA LISTA

VOCÊ PODE ESTILIZAR OS ITENS INDIVIDUAIS USANDO O SELETOR :

EXEMPLO:

```
1. LI {  
2.   COLOR: BLUE;  
3.   FONT-WEIGHT: BOLD;  
4.   MARGIN-BOTTOM: 10PX; /* ESPAÇO ENTRE ITENS */  
5. }
```

POSICIONAMENTO DO MARCADOR

A PROPRIEDADE LIST-STYLE-POSITION DEFINE A POSIÇÃO DOS MARCADORES:

- OUTSIDE [PADRÃO]: O MARCADOR FICA FORA DO CONTEÚDO.
- INSIDE: O MARCADOR É ALINHADO DENTRO DO CONTEÚDO.

EXEMPLO:

```
1. UL {  
2.   LIST-STYLE-POSITION: INSIDE;  
3. }
```

TABELAS

ESTRUTURA BÁSICA

EXEMPLO:

```
1. <TABLE>  
2. <THEAD>  
3.   <TR>  
4.     <TH>NOME</TH>  
5.     <TH>IDADE</TH>  
6.     <TH>CIDADE</TH>  
7.   </TR>  
8. </THEAD>  
9. <TBODY>  
10.  <TR>  
11.    <TD>ANA</TD>  
12.    <TD>25</TD>  
13.    <TD>SÃO PAULO</TD>  
14.  </TR>  
15.  <TR>  
16.    <TD>BRUNO</TD>  
17.    <TD>30</TD>  
18.    <TD>RIO DE JANEIRO</TD>  
19.  </TR>  
20. </TBODY>  
21. </TABLE>
```

ADICIONANDO BORDAS

VOCÊ PODE ADICIONAR BORDAS ÀS TABELAS USANDO A PROPRIEDADE BORDER:

EXEMPLO:

```
1. TABLE, TH, TD {  
2.   BORDER: 1PX SOLID BLACK;  
3.   BORDER-COLLAPSE: COLLAPSE; /* REMOVE BORDAS DUPLICADAS */  
4. }  
5.  
6. TH, TD {  
7.   PADDING: 10PX; /* ESPAÇO INTERNO */  
8.   TEXT-ALIGN: LEFT; /* ALINHAMENTO DO TEXTO */  
9. }
```

HOVER NAS LINHAS

ADICIONE EFEITOS DE DESTAQUE AO PASSAR O MOUSE:

EXEMPLO:

```
1. TBODY TR:HOVER {  
2.   BACKGROUND-COLOR: #D1E7DD; /* COR DE FUNDO AO PASSAR O  
   MOUSE */  
3.   CURSOR: POINTER; /* MOSTRA O CURSOR COMO INDICADOR */  
4. }
```



TABELAS RESPONSIVAS

USE CSS PARA CRIAR TABELAS ADAPTÁVEIS A TELAS MENORES:

EXEMPLO NO CSS:

```

1. TABLE {
2.   WIDTH: 100%;
3.   BORDER-COLLAPSE: COLLAPSE;
4. }
5.
6. TH, TD {
7.   PADDING: 10px;
8.   TEXT-ALIGN: LEFT;
9. }
10.
11. @MEDIA (MAX-WIDTH: 600px) {
12.   TABLE, THEAD, TBODY, TH, TD, TR {
13.     DISPLAY: BLOCK; /* CADA CÉLULA VIRA UM BLOCO */
14.   }
15.
16.   TH {
17.     DISPLAY: NONE; /* OCULTA O CABEÇALHO */
18.   }
19.
20.   TD {
21.     PADDING: 10px;
22.     BORDER: NONE;
23.     POSITION: RELATIVE;
24.   }
25.
26. TD::before {
27.   CONTENT: ATTR(DATA-LABEL); /* ADICIONA RÓTULOS ÀS CÉLULAS */
28.   POSITION: ABSOLUTE;
29.   LEFT: 0;
30.   FONT-WEIGHT: BOLD;
31. }
32. }
```

EXEMPLO NO HTML:

```

1. <TABLE>
2. <THEAD>
3. <TR>
4.   <TH>NOME</TH>
5.   <TH>IDADE</TH>
6.   <TH>CIDADE</TH>
7. </TR>
8. </THEAD>
9. <TBODY>
10. <TR>
11.   <TD DATA-LABEL="NOME">ANA</TD>
12.   <TD DATA-LABEL="IDADE">25</TD>
13.   <TD DATA-LABEL="CIDADE">SÃO PAULO</TD>
14. </TR>
15. <TR>
16.   <TD DATA-LABEL="NOME">BRUNO</TD>
17.   <TD DATA-LABEL="IDADE">30</TD>
18.   <TD DATA-LABEL="CIDADE">RIO DE JANEIRO</TD>
19. </TR>
20. </TBODY>
21. </TABLE>
```

ADICIONANDO BORDAS

USE O SELETOR <THEAD> OU <TH> PARA PERSONALIZAR O CABEÇALHO DA TABELA:

EXEMPLO:

```

1. THEAD {
2.   BACKGROUND-COLOR: #007BFF; /* COR DE FUNDO */
3.   COLOR: WHITE; /* COR DO TEXTO */
4.   FONT-WEIGHT: BOLD;
5. }
6.
7. TH {
8.   TEXT-TRANSFORM: UPPERCASE; /* LETRAS MAIÚSCULAS */ }
```

REMOVENDO BORDAS PADRÃO

PARA CRIAR TABELAS SEM BORDAS:

EXEMPLO:

```

1. TABLE {
2.   BORDER-COLLAPSE: COLLAPSE;
3.   BORDER: NONE;
4. }
```

ALINHAMENTO DO TEXTO

CONTROLE O ALINHAMENTO DO TEXTO EM CÉLULAS:

EXEMPLO:

```

1. TH, TD {
2.   TEXT-ALIGN: CENTER; /* ALINHA O TEXTO AO CENTRO */
3.   VERTICAL-ALIGN: MIDDLE; /* ALINHA O TEXTO VERTICALMENTE */ }
```

ANOTAÇÕES



ZEBRA STRIPES (LINHAS ALTERNADAS)

DESTAQUE LINHAS ALTERNADAS PARA FACILITAR A LEITURA:

EXEMPLO:

```
1. TBODY TR:NOT-CHILD(ODD) {  
2. BACKGROUND-COLOR: #F9F9F9; /* COR PARA LINHAS ÍMPARES */  
3.}  
4.  
5. TBODY TR:NOT-CHILD(EVEN) {  
6. BACKGROUND-COLOR: #E9E9E9; /* COR PARA LINHAS PARES */  
7.}
```

LARGURA E ALTURA DA TABELA

DEFINA A LARGURA DA TABELA E AJUSTE AS COLUNAS:

EXEMPLO:

```
1. TABLE {  
2. WIDTH: 100%; /* TABELA OCUPA 100% DO CONTAINER */  
3. TABLE-LAYOUT: FIXED; /* DISTRIBUI COLUNAS IGUALMENTE */  
4.}  
5.  
6. TH, TD {  
7. WORD-WRAP: BREAK-WORD; /* QUEBRA DE TEXTO EM CÉLULAS */  
8.}
```

DISPLAY

VALORES COMUNS DE DISPLAY

1. BLOCK

UM ELEMENTO DE BLOCO OCUPA TODA A LARGURA DISPONÍVEL E COMEÇA EM UMA NOVA LINHA.

- EXEMPLO DE ELEMENTOS PADRÃO COM DISPLAY: BLOCK: <DIV>, <P>, <H1>, <SECTION>.

EXEMPLO NO CSS:

```
1. DIV {  
2. DISPLAY: BLOCK;  
3.}
```

EXEMPLO NO HTML:

```
1. <DIV>PRIMEIRO BLOCO</DIV>  
2. <DIV>SEGUNDO BLOCO</DIV>
```

2. INLINE

UM ELEMENTO INLINE NÃO COMEÇA EM UMA NOVA LINHA E OCUPA APENAS O ESPAÇO NECESSÁRIO PARA SEU CONTEÚDO.

- EXEMPLO DE ELEMENTOS PADRÃO COM DISPLAY: INLINE: , <A>, .

EXEMPLO NO CSS:

```
1. SPAN {  
2. DISPLAY: INLINE;  
3. }
```

EXEMPLO NO HTML:

- TEXTO 1TEXTO 2

3. INLINE-BLOCK

COMPORTA-SE COMO UM ELEMENTO INLINE (NÃO QUEBRA A LINHA), MAS PERMITE DEFINIR LARGURA, ALTURA, MARGENS E PADDINGS.

EXEMPLO NO CSS:

```
1. DIV {  
2. DISPLAY: INLINE-BLOCK;  
3. WIDTH: 100px;  
4. HEIGHT: 50px;  
5. BACKGROUND-COLOR: LIGHTBLUE;  
6.}
```

EXEMPLO NO HTML:

```
1. <DIV>CAIXA 1</DIV>  
2. <DIV>CAIXA 2</DIV>
```

4. NONE

ESCONDE O ELEMENTO COMPLETAMENTE, COMO SE ELE NÃO EXISTISSE NO DOCUMENTO (SEM OCUPAR ESPAÇO).

EXEMPLO:

```
1. DIV {  
2. DISPLAY: NONE;  
3.}
```

5. FLEX

ATIVA O MODELO DE LAYOUT FLEXÍVEL (FLEXBOX) PARA O ELEMENTO, PERMITINDO ALINHAR E DISTRIBUIR ITENS DENTRO DE UM CONTÊINER.

EXEMPLO:

```
1. DIV {  
2.   DISPLAY: FLEX;  
3.   JUSTIFY-CONTENT: CENTER;  
4.   ALIGN-ITEMS: CENTER;  
5. }
```

6. GRID

ATIVA O MODELO DE LAYOUT EM GRADE (CSS GRID), PERMITINDO CRIAR LAYOUTS BASEADOS EM LINHAS E COLUNAS.

EXEMPLO:

```
1. DIV {  
2.   DISPLAY: GRID;  
3.   GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR);  
4. }
```

OUTROS VALORES

- **INLINE-FLEX**: UM CONTÊINER FLEXÍVEL QUE SE COMPORTA COMO INLINE.
- **INLINE-GRID**: UM CONTÊINER DE GRADE QUE SE COMPORTA COMO INLINE.
- **TABLE**: FAZ O ELEMENTO SE COMPORTAR COMO UMA TABELA.
- **TABLE-ROW, TABLE-CELL**: PARA SIMULAR PARTES DE UMA TABELA.
- **INHERIT**: HERDA O VALOR DE DISPLAY DO ELEMENTO PAI.
- **INITIAL**: DEFINE O VALOR INICIAL PADRÃO DE DISPLAY.

EXEMPLO COMPARATIVO

EXEMPLO:

```
1. <STYLE>  
2. .BLOCK {  
3.   DISPLAY: BLOCK;  
4.   BACKGROUND-COLOR: LIGHTBLUE;  
5.   MARGIN: 10px;  
6. }  
7. .INLINE {  
8.   DISPLAY: INLINE;  
9.   BACKGROUND-COLOR: LIGHTGREEN;  
10.  MARGIN: 10px;  
11. }  
12. .INLINE-BLOCK {  
13.   DISPLAY: INLINE-BLOCK;  
14.   BACKGROUND-COLOR: LIGHTCORAL;  
15.   MARGIN: 10px;  
16.   WIDTH: 100px;  
17.   HEIGHT: 50px;  
18. }  
19. </STYLE>  
20.  
21. <DIV CLASS="BLOCK">BLOCO 1</DIV>  
22. <DIV CLASS="BLOCK">BLOCO 2</DIV>  
23. <DIV CLASS="INLINE">INLINE 1</DIV>  
24. <DIV CLASS="INLINE">INLINE 2</DIV>  
25. <DIV CLASS="INLINE-BLOCK">INLINE-BLOCK 1</DIV>  
26. <DIV CLASS="INLINE-BLOCK">INLINE-BLOCK 2</DIV>
```

POSITION

VALORES DE POSITION

1. STATIC (PADRÃO)

- O ELEMENTO É POSICIONADO DE ACORDO COM O FLUXO NORMAL DO DOCUMENTO.
- NÃO ACEITA PROPRIEDADES DE DESLOCAMENTO (COMO TOP, RIGHT, BOTTOM, LEFT).

EXEMPLO NO CSS:

```
1. DIV {  
2.   POSITION: STATIC;  
3. }
```

2. RELATIVE

- O ELEMENTO É POSICIONADO RELATIVO À SUA POSIÇÃO ORIGINAL NO FLUXO DO DOCUMENTO.
- PERMITE USAR AS PROPRIEDADES DE DESLOCAMENTO (TOP, RIGHT, BOTTOM, LEFT) PARA AJUSTAR SUA POSIÇÃO.

EXEMPLO NO CSS:

```
1. DIV {  
2.   POSITION: RELATIVE;  
3.   TOP: 20px;  
4.   LEFT: 10px; }
```



3. ABSOLUTE

- O ELEMENTO É POSICIONADO RELATIVO AO SEU CONTÊINER DE REFERÊNCIA MAIS PRÓXIMO QUE TENHA POSITION: RELATIVE, ABSOLUTE OU FIXED.
- SE NENHUM CONTÊINER FOR ENCONTRADO, SERÁ POSICIONADO EM RELAÇÃO AO CORPO DO DOCUMENTO.
- REMOVE O ELEMENTO DO FLUXO NORMAL DO DOCUMENTO.

EXEMPLO:

```
1. DIV {  
2.   POSITION: ABSOLUTE;  
3.   TOP: 50PX;  
4.   LEFT: 100PX;  
5. }
```

4. FIXED

- O ELEMENTO É POSICIONADO RELATIVO À JANELA DO NAVEGADOR.
- NÃO SE MOVE AO ROLAR A PÁGINA.

EXEMPLO:

```
1. DIV {  
2.   POSITION: FIXED;  
3.   BOTTOM: 10PX;  
4.   RIGHT: 20PX;  
5. }
```

5. STICKY

- O ELEMENTO ALTERNA ENTRE RELATIVE E FIXED, DEPENDENDO DA ROLAGEM DA PÁGINA.
- FICA PRESO A UMA POSIÇÃO ESPECÍFICA (DEFINIDA PELAS PROPRIEDADES DE DESLOCAMENTO) ENQUANTO ESTÁ DENTRO DE SEU CONTÊINER.

EXEMPLO:

```
1. DIV {  
2.   POSITION: STICKY;  
3.   TOP: 0;  
4. }
```

COMPARATIVO VISUAL

EXEMPLO:

```
<STYLE>  
.STATIC {  
  POSITION: STATIC;  
  BACKGROUND: LIGHTGRAY;  
}  
.RELATIVE {  
  POSITION: RELATIVE;  
  TOP: 20PX;  
  LEFT: 20PX;  
  BACKGROUND: LIGHTBLUE;  
}  
.ABSOLUTE {  
  POSITION: ABSOLUTE;  
  TOP: 30PX;  
  LEFT: 40PX;  
  BACKGROUND: LIGHTCORAL;  
}  
.FIXED {  
  POSITION: FIXED;  
  BOTTOM: 10PX;  
  RIGHT: 10PX;  
  BACKGROUND: LIGHTGREEN;  
}  
.STICKY {  
  POSITION: STICKY;  
  TOP: 0;  
  BACKGROUND: YELLOW;  
}  
</STYLE>  
<DIV CLASS="STATIC">STATIC</DIV>  
<DIV CLASS="RELATIVE">RELATIVE</DIV>  
<DIV CLASS="ABSOLUTE">ABSOLUTE</DIV>  
<DIV CLASS="FIXED">FIXED</DIV>  
<DIV CLASS="STICKY">STICKY (SCROLL PARA TESTAR)</DIV>
```

PROPRIEDADES DE DESLOCAMENTO

AS PROPRIEDADES TOP, RIGHT, BOTTOM, E LEFT AJUSTAM A POSIÇÃO DE UM ELEMENTO QUANDO A PROPRIEDADE POSITION ESTÁ DEFINIDA COMO RELATIVE, ABSOLUTE, FIXED OU STICKY.

Z-INDEX

COMO FUNCIONA O Z-INDEX

1. PADRÃO:

- O VALOR PADRÃO É AUTO, O QUE SIGNIFICA QUE O ELEMENTO SEGUERÁ A ORDEM DE EMPILHAMENTO NATURAL NO FLUXO DO DOCUMENTO.
- ELEMENTOS QUE APARECEREM MAIS TARDE NO CÓDIGO SÃO RENDERIZADOS NA FRENTES DE OUTROS, A MENOS QUE O Z-INDEX SEJA ESPECIFICADO.

2. SOMENTE ELEMENTOS COM POSITION DIFERENTE DE STATIC:

- PARA QUE O Z-INDEX TENHA EFEITO, O ELEMENTO DEVE TER UMA PROPRIEDADE POSITION DEFINIDA COMO RELATIVE, ABSOLUTE, FIXED OU STICKY.

VALORES DE Z-INDEX

1. VALORES POSITIVOS

ELEMENTOS COM VALORES DE Z-INDEX MAIORES APARECERÃO NA FRENTES DE ELEMENTOS COM VALORES MENORES.

EXEMPLO:

```
1. DIV1 {  
2.   POSITION: RELATIVE;  
3.   Z-INDEX: 1;  
4. }  
5. DIV2 {  
6.   POSITION: RELATIVE;  
7.   Z-INDEX: 2; }
```

2. VALORES NEGATIVOS

ELEMENTOS COM VALORES DE Z-INDEX NEGATIVOS SÃO COLOCADOS ATRÁS DE OUTROS ELEMENTOS.

EXEMPLO:

```
1. DIV1 {  
2.   POSITION: RELATIVE;  
3.   Z-INDEX: -1;  
4. }  
5. DIV2 {  
6.   POSITION: RELATIVE;  
7.   Z-INDEX: 0; }
```

3. Z-INDEX: AUTO

O ELEMENTO SEGUERÁ A ORDEM PADRÃO DE EMPILHAMENTO, CONFORME A POSIÇÃO NO FLUXO DO DOCUMENTO.

ORDEM DE EMPILHAMENTO (STACKING CONTEXT)

A CRIAÇÃO DE UM CONTEXTO DE EMPILHAMENTO OCORRE AUTOMATICAMENTE EM DETERMINADAS SITUAÇÕES, COMO:

1. UM ELEMENTO COM POSITION: RELATIVE, ABSOLUTE, FIXED OU STICKY E COM UM VALOR DE Z-INDEX EXPLÍCITO.
2. ELEMENTOS RAÍZ DA PÁGINA (COMO <HTML> E <BODY>).
3. ELEMENTOS COM PROPRIEDADES COMO OPACITY MENOR QUE 1, TRANSFORM, FILTER OU PERSPECTIVE.

IMPORTANTE:

DENTRO DE UM CONTEXTO DE EMPILHAMENTO, OS ELEMENTOS FILHOS SÃO ORGANIZADOS DE ACORDO COM SEUS PRÓPRIOS VALORES DE Z-INDEX, INDEPENDENTEMENTE DE VALORES DE Z-INDEX FORA DESSE CONTEXTO.

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="en">  
<HEAD>  
  <META CHARSET="UTF-8">  
  <META NAME="viewport" CONTENT="width=device-width, initial-scale=1.0">  
<TITLE>Z-INDEX EXAMPLE</TITLE>  
<STYLE>  
  .BOX1 {  
    POSITION: RELATIVE;  
    Z-INDEX: 1;  
    BACKGROUND: LIGHTBLUE;  
    WIDTH: 150px;  
    HEIGHT: 150px;  
    MARGIN: -50px;  
  }  
  .BOX2 {  
    POSITION: RELATIVE;  
    Z-INDEX: 2;  
    BACKGROUND: LIGHTCORAL;  
    WIDTH: 150px;  
    HEIGHT: 150px;  
    MARGIN: -50px;  
  }
```



```
.BOX3 {
  position: relative;
  z-index: -1;
  background: lightgreen;
  width: 150px;
  height: 150px;
  margin: -50px;
}
</style>
</head>
<body>
<div class="BOX1">BOX 1</div>
<div class="BOX2">BOX 2</div>
<div class="BOX3">BOX 3</div>
</body>
</html>
```



OVERFLOW

A PROPRIEDADE **OVERFLOW** NO CSS CONTROLA COMO O CONTEÚDO QUE EXCDE O TAMANHO DE UM ELEMENTO É TRATADO. QUANDO O CONTEÚDO DE UM ELEMENTO ULTRAPASSA SUAS DIMENSÕES DEFINIDAS (ALTURA OU LARGURA), O OVERFLOW DEFINE SE ESSE CONTEÚDO SERÁ EXIBIDO, CORTADO OU SE BARRAS DE ROLAGEM SERÃO ADICIONADAS.

SINTAXE

OVERFLOW: VISIBLE | HIDDEN | SCROLL | AUTO | INHERIT;

VALORES DO OVERFLOW

VISIBLE (PADRÃO)

- O CONTEÚDO QUE EXCDE AS DIMENSÕES DO ELEMENTO SERÁ VISÍVEL FORA DO SEU CONTÊINER.

EXEMPLO:

```
1. DIV {
  2.   width: 200px;
  3.   height: 100px;
  4.   overflow: visible;
  5. }
```

HIDDEN

- O CONTEÚDO QUE EXCER AS DIMENSÕES SERÁ CORTADO, E NÃO SERÃO ADICIONADAS BARRAS DE ROLAGEM.

EXEMPLO:

```
1. DIV {
  2.   width: 200px;
  3.   height: 100px;
  4.   overflow: hidden;
  5. }
```

SCROLL

- SEMPRE ADICIONA BARRAS DE ROLAGEM, INDEPENDENTEMENTE DE O CONTEÚDO ULTRAPASSAR OU NÃO AS DIMENSÕES DO ELEMENTO.

EXEMPLO:

```
1. DIV {
  2.   width: 200px;
  3.   height: 100px;
  4.   overflow: scroll; }
```

AUTO

- ADICIONA BARRAS DE ROLAGEM AUTOMATICAMENTE SOMENTE SE O CONTEÚDO EXCEDER AS DIMENSÕES DO ELEMENTO.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: 200px;  
3.   HEIGHT: 100px;  
4.   OVERFLOW: AUTO;  
5. }
```

INHERIT

- HERDA O VALOR DO ELEMENTO PAI.

EXEMPLO:

```
1. DIV {  
2.   OVERFLOW: INHERIT;  
3. }
```

VALORES ESPECÍFICOS

- **OVERFLOW-X**: CONTROLA O OVERFLOW HORIZONTAL.
- **OVERFLOW-Y**: CONTROLA O OVERFLOW VERTICAL.

EXEMPLO:

```
1. DIV {  
2.   OVERFLOW-X: HIDDEN; /* OCULTA O CONTEÚDO QUE EXCDE NA LARGURA */  
3.   OVERFLOW-Y: SCROLL; /* SEMPRE ADICIONA BARRA DE ROLAGEM VERTICAL */  
4. }
```

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="en">  
<HEAD>  
<META CHARSET="UTF-8">  
<META NAME="viewport" CONTENT="width=device-width, initial-scale=1.0">  
<TITLE>OVERFLOW EXAMPLE</TITLE>  
<STYLE>  
.BOX {  
  width: 300px;  
  height: 150px;  
  border: 2px solid #000;  
  margin: 20px; }
```

.OVERFLOW-VISIBLE

```
OVERFLOW: VISIBLE;  
BACKGROUND: LIGHTBLUE;  
}
```

.OVERFLOW-HIDDEN

```
OVERFLOW: HIDDEN;  
BACKGROUND: LIGHTGREEN;  
}
```

.OVERFLOW-SCROLL

```
OVERFLOW: SCROLL;  
BACKGROUND: LIGHTCORAL;  
}
```

.OVERFLOW-AUTO

```
OVERFLOW: AUTO;  
BACKGROUND: LIGHTYELLOW;  
}
```

```
</STYLE>
```

```
</HEAD>
```

```
<BODY>
```

```
<DIV CLASS="BOX OVERFLOW-VISIBLE">
```

```
  <P>ESTE É UM EXEMPLO COM `OVERFLOW: VISIBLE`. O CONTEÚDO QUE EXCDE O TAMANHO NÃO SERÁ CORTADO.</P>
```

```
</DIV>
```

```
<DIV CLASS="BOX OVERFLOW-HIDDEN">
```

```
  <P>ESTE É UM EXEMPLO COM `OVERFLOW: HIDDEN`. O CONTEÚDO QUE EXCDE SERÁ CORTADO.</P>
```

```
</DIV>
```

```
<DIV CLASS="BOX OVERFLOW-SCROLL">
```

```
  <P>ESTE É UM EXEMPLO COM `OVERFLOW: SCROLL`. SEMPRE HAVERÁ BARRAS DE ROLAGEM.</P>
```

```
</DIV>
```

```
<DIV CLASS="BOX OVERFLOW-AUTO">
```

```
  <P>ESTE É UM EXEMPLO COM `OVERFLOW: AUTO`. AS BARRAS DE ROLAGEM APARECEM APENAS SE NECESSÁRIO.</P>
```

```
</DIV>
```

```
</BODY>
```

```
</HTML>
```



FLOAT E CLEAR

FLOAT

A PROPRIEDADE FLOAT É USADA PARA POSICIONAR UM ELEMENTO À ESQUERDA OU À DIREITA DENTRO DO CONTÊINER PAI, PERMITINDO QUE OUTROS ELEMENTOS "FLUTUEM" AO SEU LADO. GERALMENTE É USADA PARA CRIAR LAYOUTS SIMPLES OU ALINHAR IMAGENS E TEXTO.

SINTAXE:

1. **FLOAT: none | left | right | inherit;**

none (Padrão): O ELEMENTO NÃO FLUTUA.

left: O ELEMENTO FLUTUA PARA O LADO ESQUERDO DO CONTÊINER PAI.

right: O ELEMENTO FLUTUA PARA O LADO DIREITO DO CONTÊINER PAI.

inherit: HERDA O VALOR DA PROPRIEDADE FLOAT DO ELEMENTO PAI.

CLEAR

A PROPRIEDADE CLEAR É USADA PARA CONTROLAR O COMPORTAMENTO DE ELEMENTOS QUE Vêm DEPOIS DE UM ELEMENTO FLUTUANTE. ELA IMPIDE QUE O PRÓXIMO ELEMENTO "SUBA" AO LADO DO ELEMENTO FLUTUANTE.

SINTAXE:

1. **CLEAR: none | left | right | both | inherit;**

none (Padrão): NÃO IMPIDE QUE O ELEMENTO SEGUINTE FLUA AO LADO DO ELEMENTO FLUTUANTE.

left: O ELEMENTO SEGUINTE É MOVIDO PARA BAIXO, AFASTANDO-SE DE ELEMENTOS FLUTUANTES À ESQUERDA.

right: O ELEMENTO SEGUINTE É MOVIDO PARA BAIXO, AFASTANDO-SE DE ELEMENTOS FLUTUANTES À DIREITA.

both: O ELEMENTO SEGUINTE É MOVIDO PARA BAIXO, AFASTANDO-SE DE ELEMENTOS FLUTUANTES TANTO À ESQUERDA QUANTO À DIREITA.

inherit: HERDA O VALOR DA PROPRIEDADE CLEAR DO ELEMENTO PAI.

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>FLOAT E CLEAR</TITLE>
<STYLE>
.CONTAINER {
  width: 100%;
  border: 1px solid black;
  padding: 10px;
}

.BOX {
  width: 100px;
  height: 100px;
  margin: 5px;
}

.BOX1 {
  background-color: lightblue;
  float: left;
}

```

```
.BOX2 {
  background-color: lightgreen;
  float: right;
}

.BOX3 {
  background-color: lightcoral;
  clear: both; /* EVITA QUE ELEMENTOS FLUTUANTES INTERFERAM */
}

</STYLE>
</HEAD>
<BODY>
<DIV CLASS="CONTAINER">
<DIV CLASS="BOX BOX1">FLOAT LEFT</DIV>
<DIV CLASS="BOX BOX2">FLOAT RIGHT</DIV>
<DIV CLASS="BOX BOX3">CLEAR BOTH</DIV>
</DIV>
</BODY>
</HTML>
```

COMBINADORES

COMBINADOR DESCENDENTE (ESPAÇO)

SELEÇÃO TODOS OS ELEMENTOS DESCENDENTES (FILHOS, NETOS, ETC.) DE UM ELEMENTO PAI.

SINTAXE:

```
1. ELEMENTO1 ELEMENTO2 {  
2. /* ESTILO */  
3. }
```

EXEMPLO:

```
1. DIV P {  
2. COLOR: BLUE;  
3. }
```

COMBINADOR FILHO (>)

SELEÇÃO APENAS OS ELEMENTOS QUE SÃO FILHOS DIRETOS DE OUTRO ELEMENTO.

SINTAXE:

```
1. ELEMENTO1 > ELEMENTO2 {  
2. /* ESTILO */  
3. }
```

EXEMPLO:

```
1. DIV > P {  
2. COLOR: GREEN;  
3. }
```

COMBINADOR ADJACENTE (+)

SELEÇÃO O PRIMEIRO ELEMENTO QUE VEM IMEDIATAMENTE DEPOIS DE OUTRO ELEMENTO.

SINTAXE:

```
1. ELEMENTO1 + ELEMENTO2 {  
2. /* ESTILO */  
3. }
```

EXEMPLO:

```
1. H1 + P {  
2. COLOR: RED;  
3. }
```

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="EN">  
<HEAD>  
  <META CHARSET="UTF-8">  
  <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-  
  SCALE=1.0">  
  <TITLE>COMBINADORES CSS</TITLE>  
<STYLE>  
  /* DESCENDENTE */  
  DIV P {  
    COLOR: BLUE;  
  }  
  
  /* FILHO DIRETO */  
  DIV > P {  
    FONT-WEIGHT: BOLD;  
  }  
  
  /* ADJACENTE */  
  H1 + P {  
    FONT-STYLE: ITALIC;  
  }  
  
  /* GERAL DE IRMÃOS */  
  H1 ~ P {  
    TEXT-DECORATION: UNDERLINE;  
  }  
</STYLE>  
</HEAD>
```

COMBINADOR GERAL DE IRMÃOS (~)

SELEÇÃO TODOS OS ELEMENTOS QUE COMPARTILHAM O MESMO PAI E Vêm DEPOIS DE OUTRO ELEMENTO.

SINTAXE:

```
1. ELEMENTO1 ~ ELEMENTO2 {  
2. /* ESTILO */  
3. }
```

EXEMPLO:

```
1. H1 ~ P {  
2. COLOR: PURPLE;  
3. }
```



```
<BODY>
<DIV>
<P>PARÁGRAFO DENTRO DE UMA DIV.</P>
<P>OUTRO PARÁGRAFO DENTRO DA MESMA DIV.</P>
</DIV>

<H1>TÍTULO</H1>
<P>ESTE PARÁGRAFO É ADJACENTE AO H1.</P>
<P>ESTE PARÁGRAFO É IRMÃO GERAL DO H1.</P>
</BODY>
</HTML>
```



PSEUDO CLASSES

PSEUDO-CLASSES DE LINKS

CONTROLAM O ESTILO DE LINKS EM DIFERENTES ESTADOS.

- **:LINK** – APPLICA ESTILOS A LINKS AINDA NÃO VISITADOS.
- **:VISITED** – APPLICA ESTILOS A LINKS JÁ VISITADOS.
- **:HOVER** – APPLICA ESTILOS QUANDO O MOUSE PASSA SOBRE O ELEMENTO.
- **:ACTIVE** – APPLICA ESTILOS ENQUANTO O ELEMENTO ESTÁ SENDO CLICADO.

EXEMPLO:

```
1. a:link {
2.   color: blue;
3. }
4.
5. a:visited {
6.   color: purple;
7. }
8.
9. a:hover {
10.  color: green;
11. }
12.
13. a:active {
14.  color: red;
15. }
```

PSEUDO-CLASSES DE FORMULÁRIOS

APLICAM ESTILOS A ELEMENTOS DE FORMULÁRIO COM BASE EM SEU ESTADO.

- **:FOCUS** – QUANDO O ELEMENTO ESTÁ EM FOCO.
- **:CHECKED** – QUANDO CAIXAS DE SELEÇÃO OU BOTÕES DE RÁDIO ESTÃO MARCADOS.
- **:DISABLED** – QUANDO O ELEMENTO ESTÁ DESATIVADO.
- **:ENABLED** – QUANDO O ELEMENTO ESTÁ ATIVADO.
- **:REQUIRED** – PARA CAMPOS OBRIGATÓRIOS.
- **:OPTIONAL** – PARA CAMPOS NÃO OBRIGATÓRIOS.
- **:VALID** – QUANDO A ENTRADA É VÁLIDA.
- **:INVALID** – QUANDO A ENTRADA É INVÁLIDA.

EXEMPLO:

```
1. input:focus {
2.   border-color: blue;
3. }
4.
5. input:required {
6.   background-color: lightyellow;
7. }
8.
9. input:valid {
10.  border-color: green;
11. }
12.
13. input:invalid {
14.  border-color: red;
15. }
```

PSEUDO-CLASSES ESTRUTURAIS

ESTILIZAM ELEMENTOS COM BASE EM SUA POSIÇÃO NA ESTRUTURA HTML.

- **:FIRST-CHILD** – SELEÇÃO A O PRIMEIRO FILHO DE UM ELEMENTO.
- **:LAST-CHILD** – SELEÇÃO A O ÚLTIMO FILHO DE UM ELEMENTO.
- **:NTH-CHILD(n)** – SELEÇÃO O ENÉSIMO FILHO DE UM ELEMENTO.
- **:NTH-LAST-CHILD(n)** – SELEÇÃO O ENÉSIMO FILHO A PARTIR DO FINAL.
- **:ONLY-CHILD** – SELEÇÃO ELEMENTOS QUE SÃO OS ÚNICOS FILHOS DE SEUS PAIS.

EXEMPLO:

```
1. P:FIRST-CHILD {  
2.   FONT-WEIGHT: BOLD;  
3. }  
4.  
5. LI:NTH-CHILD(2) {  
6.   COLOR: RED;  
7. }  
8.  
9. DIV:LAST-CHILD {  
10.  BACKGROUND-COLOR: LIGHTBLUE;  
11. }
```

PSEUDO-CLASSES DINÂMICAS

ESTILIZAM ELEMENTOS COM BASE EM INTERAÇÕES OU ESTADOS DINÂMICOS.

- **:HOVER** – APPLICA ESTILOS QUANDO O MOUSE ESTÁ SOBRE O ELEMENTO.
- **:FOCUS** – QUANDO O ELEMENTO RECEBE FOCO.
- **:ACTIVE** – QUANDO O ELEMENTO É ATIVADO (CLICADO).

EXEMPLO:

```
1. BUTTON:HOVER {  
2.   BACKGROUND-COLOR: YELLOW;  
3. }  
4.  
5. BUTTON:ACTIVE {  
6.   TRANSFORM: SCALE(0.98);  
7. }
```

PSEUDO-CLASSES DE GRUPOS

SELECIONAM ELEMENTOS QUE PERTENCEM A CATEGORIAS ESPECÍFICAS.

- **:EMPTY** – SELEÇÃO ELEMENTOS QUE NÃO TÊM FILHOS (NEM TEXTO, NEM OUTROS ELEMENTOS).
- **:NOT(SELECTOR)** – SELEÇÃO TUDO QUE NÃO CORRESPONDE AO SELETOR.
- **:ROOT** – SELEÇÃO O ELEMENTO RAIZ (<HTML>).

EXEMPLO:

```
1. DIV:EMPTY {  
2.   DISPLAY: NONE;  
3. }  
4.  
5. P:NOT(HIGHLIGHT) {  
6.   COLOR: GRAY;  
7. }  
8.  
9. :ROOT {  
10.  --MAIN-COLOR: BLUE;  
11. }
```

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="EN">  
<HEAD>  
<META CHARSET="UTF-8">  
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-  
SCALE=1.0">  
<TITLE>PSEUDO-CLASSES NO CSS</TITLE>  
<STYLE>  
/* LINKS */  
A:LINK {  
  COLOR: BLUE;  
}  
A:VISITED {  
  COLOR: PURPLE;  
}  
A:HOVER {  
  COLOR: GREEN;  
}  
A:ACTIVE {  
  COLOR: RED;  
}  
  
/* FORMULÁRIOS */  
INPUT:FOCUS {  
  BORDER: 2PX SOLID BLUE;  
}  
INPUT:REQUIRED {  
  BACKGROUND-COLOR: LIGHTYELLOW;  
}
```



```

/* ESTRUTURAIS */
P::FIRST-CHILD {
  FONT-WEIGHT: BOLD;
}
LI::NTH-CHILD(ODD) {
  BACKGROUND-COLOR: LIGHTGRAY;
}

/* DINÂMICAS */
BUTTON::HOVER {
  BACKGROUND-COLOR: YELLOW;
}
</STYLE>
</HEAD>
<BODY>
<A HREF="#">LINK DE EXEMPLO</A>

```

→

```

<FORM>
<INPUT TYPE="TEXT" REQUIRED PLACEHOLDER="CAMPO OBRIGATÓRIO">
<INPUT TYPE="TEXT" PLACEHOLDER="CAMPO OPCIONAL">
</FORM>

<P>PRIMEIRO PARÁGRAFO.</P>
<P>SEGUNDO PARÁGRAFO.</P>

<UL>
<LI>ITEM 1</LI>
<LI>ITEM 2</LI>
<LI>ITEM 3</LI>
</UL>

<BUTTON>HOVER EM MIM</BUTTON>
</BODY>
</HTML>

```

PSEUDO ELEMENTOS

::FIRST-LINE

ESTILIZA APENAS A PRIMEIRA LINHA DE TEXTO DE UM ELEMENTO.

EXEMPLO:

1. P::FIRST-LINE {
2. FONT-WEIGHT: BOLD;
3. COLOR: RED;
4. }

::FIRST-LETTER

ESTILIZA APENAS A PRIMEIRA LETRA DE UM ELEMENTO.

EXEMPLO:

1. P::FIRST-LETTER {
2. FONT-SIZE: 2EM;
3. COLOR: BLUE;
4. FONT-WEIGHT: BOLD;
5. }

::BEFORE & ::AFTER

ESSES PSEUDO-ELEMENTOS SÃO USADOS PARA ADICIONAR CONTEÚDO ANTES OU DEPOIS DO CONTEÚDO REAL DE UM ELEMENTO. ELES GERALMENTE PRECISAM DA PROPRIEDADE CONTENT PARA FUNCIONAR.

EXEMPLO:

1. P::BEFORE {
2. CONTENT: "ANTES:";
3. COLOR: BLUE;
4. }
- 5.
6. P::AFTER {
7. CONTENT: "(DEPOIS)";
8. COLOR: GREEN;
9. }

::SELECTION

ESTILIZA A PARTE DO TEXTO SELECIONADA PELO USUÁRIO.

EXEMPLO:

1. ::SELECTION {
2. BACKGROUND-COLOR: YELLOW;
3. COLOR: BLACK;
4. }

::PLACEHOLDER

ESTILIZA O TEXTO DE ESPAÇO RESERVADO (PLACEHOLDER) EM CAMPOS DE ENTRADA.

EXEMPLO:

```
1. INPUT::PLACEHOLDER {  
2.   COLOR: GRAY;  
3.   FONT-STYLE: ITALIC;  
4. }
```

::BACKDROP

ESTILIZA O PLANO DE FONDO DE UM ELEMENTO DE DIÁLOGO EM TELA CHEIA.

EXEMPLO:

```
1. DIALOG::BACKDROP {  
2.   BACKGROUND-COLOR: RGBA(0, 0, 0, 0.5);  
3. }
```

::MARKER

ESTILIZA OS MARCADORES DE LISTAS (<U>).

EXEMPLO:

```
1. LI::MARKER {  
2.   COLOR: RED;  
3.   FONT-SIZE: 1.2EM;  
4. }
```

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="EN">  
<HEAD>  
<META CHARSET="UTF-8">  
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-  
SCALE=1.0">  
<TITLE>PSEUDO-ELEMENTOS NO CSS</TITLE>  
<STYLE>  
/* ANTES E DEPOIS */  
P::BEFORE {  
  CONTENT: "> ";  
  COLOR: BLUE;  
}
```

P::AFTER {

```
CONTENT: "<<";  
COLOR: GREEN;  
}
```

/* PRIMEIRA LINHA E PRIMEIRA LETRA */

```
P::FIRST-LINE {  
  FONT-WEIGHT: BOLD;  
  TEXT-TRANSFORM: UPPERCASE;  
}
```

```
P::FIRST-LETTER {  
  FONT-SIZE: 2EM;  
  COLOR: RED;  
}
```

/* SELEÇÃO */

```
::SELECTION {  
  BACKGROUND-COLOR: YELLOW;  
  COLOR: BLACK;  
}
```

/* PLACEHOLDER */

```
INPUT::PLACEHOLDER {  
  COLOR: GRAY;  
  FONT-STYLE: ITALIC;  
}  
</STYLE>  
</HEAD>  
<BODY>
```

<P>ESTE É UM TEXTO DE EXEMPLO PARA DEMONSTRAR PSEUDO-ELEMENTOS NO CSS.</P>

```
<FORM>  
<INPUT TYPE="TEXT" PLACEHOLDER="DIGITE ALGO AQUI">  
</FORM>  
</BODY>  
</HTML>
```



OPACIDADE

NO CSS, A PROPRIEDADE OPACITY CONTROLA A TRANSPARÊNCIA DE UM ELEMENTO. O VALOR VARIA DE 0 (COMPLETAMENTE TRANSPARENTE) A 1 (COMPLETAMENTE OPACO).

SINTAXE:

```
1. SELETOR {  
2.   OPACITY: VALUE;  
3. }
```

VALUE: UM NÚMERO ENTRE 0 E 1.

- 0: TOTALMENTE TRANSPARENTE.
- 1: TOTALMENTE OPACO.

TRANSPARÊNCIA SIMPLES

EXEMPLO NO CSS:

```
1. DIV {  
2.   OPACITY: 0.5; /* 50% OPACO */  
3. }
```

EXEMPLO NO HTML:

```
1. <DIV>ESTE ELEMENTO ESTÁ 50% TRANSPARENTE.</DIV>
```

DIFERENTES VALORES DE OPACIDADE

EXEMPLO NO CSS:

```
1. DIV.LOW-OPACITY {  
2.   OPACITY: 0.2; /* 20% OPACO */  
3. }  
4.  
5. DIV.MEDIUM-OPACITY {  
6.   OPACITY: 0.5; /* 50% OPACO */  
7. }  
8.  
9. DIV.HIGH-OPACITY {  
10.  OPACITY: 0.8; /* 80% OPACO */  
11. }
```

EXEMPLO NO HTML:

```
1. <DIV CLASS="LOW-OPACITY">20% OPACO</DIV>  
2. <DIV CLASS="MEDIUM-OPACITY">50% OPACO</DIV>  
3. <DIV CLASS="HIGH-OPACITY">80% OPACO</DIV>
```

FUNDO TRANSLÚCIDO COM RGBA

EXEMPLO NO CSS:

```
1. DIV {  
2.   BACKGROUND-COLOR: RGBA(0, 0, 255, 0.5); /* AZUL COM 50% DE  
3.     OPACIDADE */  
4.   COLOR: WHITE;  
5.   PADDING: 20PX;  
6. }
```

EXEMPLO NO HTML:

```
1. <DIV>TEXTO SEM TRANSPARÊNCIA, MAS O FUNDO É TRANSLÚCIDO.  
2. </DIV>
```

RGBA PARA MANTER O TEXTO OPACO

EXEMPLO NO CSS:

```
1. DIV {  
2.   BACKGROUND-COLOR: RGBA(0, 0, 0, 0.5); /* FUNDO 50%  
3.     TRANSPARENTE */  
4.   COLOR: WHITE; /* TEXTO OPACO */  
5.   PADDING: 20PX;  
6. }
```



SELETORES DE ATRIBUTOS

[ATRIBUTO]

SELEÇÃO ELEMENTOS QUE POSSUEM UM ATRIBUTO ESPECÍFICO, INDEPENDENTEMENTE DO VALOR.

EXEMPLO:

```
1. [TYPE] {  
2.   COLOR: RED;  
3. }
```

[ATRIBUTO="VALOR"]

SELEÇÃO ELEMENTOS CUJO ATRIBUTO CONTÉM UMA LISTA DE VALORES SEPARADOS POR ESPAÇOS, INCLUINDO O VALOR ESPECIFICADO.

EXEMPLO:

```
1. [CLASS~="BUTTON"] {  
2.   BACKGROUND-COLOR: GREEN;  
3. }
```

[ATRIBUTO^="VALOR"]

SELEÇÃO ELEMENTOS CUJO ATRIBUTO COMEÇA COM O VALOR ESPECIFICADO.

EXEMPLO:

```
1. [HREF^="HTTPS://"] {  
2.   COLOR: GREEN;  
3. }
```

[ATRIBUTO|= "VALOR"]

SELEÇÃO ELEMENTOS ONDE O ATRIBUTO COMEÇA COM O VALOR ESPECIFICADO OU É EXATAMENTE IGUAL A ELE. GERALMENTE USADO PARA VALORES DE IDIOMA.

EXEMPLO:

```
1. [LANG|= "En"] {  
2.   FONT-STYLE: ITALIC;  
3. }
```

[ATRIBUTO="VALOR"]

SELEÇÃO ELEMENTOS ONDE O ATRIBUTO POSSUI EXATAMENTE O VALOR ESPECIFICADO.

EXEMPLO:

```
1. [TYPE="TEXT"] {  
2.   BORDER: 1PX SOLID BLUE;  
3. }
```

[ATRIBUTO*="VALOR"]

SELEÇÃO ELEMENTOS CUJO ATRIBUTO CONTÉM O VALOR ESPECIFICADO EM QUALQUER POSIÇÃO.

EXEMPLO:

```
1. [TITLE*="INFO"] {  
2.   BACKGROUND-COLOR: YELLOW;  
3. }
```

[ATRIBUTO\$="VALOR"]

SELEÇÃO ELEMENTOS CUJO ATRIBUTO TERMINA COM O VALOR ESPECIFICADO.

EXEMPLO:

```
1. [SRC$=".JPG"] {  
2.   BORDER: 2PX SOLID BLACK;  
3. }
```



EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
    <META CHARSET="UTF-8">
    <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>SELETORES DE ATRIBUTOS</TITLE>
<STYLE>
    /* QUALQUER ELEMENTO COM O ATRIBUTO "TYPE" */
    [TYPE] {
        BACKGROUND-COLOR: LIGHTGRAY;
        MARGIN: 5PX;
        PADDING: 10PX;
    }

    /* INPUTS COM TYPE="TEXT" */
    [TYPE="TEXT"] {
        BORDER: 2PX SOLID BLUE;
    }

    /* LINKS QUE COMEÇAM COM "HTTPS://" */
    [HREF^="HTTPS://"] {
        COLOR: GREEN;
    }

    /* IMAGENS QUE TERMINAM COM ".PNG" */
    [SRC$=".PNG"] {
        BORDER: 2PX DASHED RED;
    }

```

```
/* ELEMENTOS COM UMA CLASSE QUE CONTÉM "HIGHLIGHT" */
[CLASS~="HIGHLIGHT"] {
    BACKGROUND-COLOR: YELLOW;
    FONT-WEIGHT: BOLD;
}

</STYLE>
</HEAD>
<BODY>
    <H1>EXEMPLOS DE SELETORES DE ATRIBUTOS</H1>

    <!-- ATRIBUTO TYPE -->
    <INPUT TYPE="TEXT" PLACEHOLDER="DIGITE ALGO">
    <INPUT TYPE="PASSWORD" PLACEHOLDER="SENHA">

    <!-- ATRIBUTO HREF -->
    <A HREF="HTTPS://EXAMPLE.COM">LINK SEGUNDO</A>
    <A HREF="HTTP://EXAMPLE.COM">LINK NÃO SEGUNDO</A>

    <!-- ATRIBUTO SRC -->
    <IMG SRC="IMAGEM.PNG" ALT="IMAGEM PNG">
    <IMG SRC="IMAGEM.JPG" ALT="IMAGEM JPG">

    <!-- CLASSE COM HIGHLIGHT -->
    <P CLASS="TEXT HIGHLIGHT">ESTE PARÁGRAFO CONTÉM A CLASSE "HIGHLIGHT".</P>
    <P CLASS="TEXT">ESTE PARÁGRAFO NÃO CONTÉM A CLASSE "HIGHLIGHT".</P>
</BODY>
</HTML>
```

FORMS

CAMPOS DE SELEÇÃO (<SELECT>)

EXEMPLO NO CSS:

1. SELECT {
2. WIDTH: 300PX;
3. PADDING: 10PX;
4. BORDER: 2PX SOLID #CCC;
5. BORDER-RADIUS: 5PX;
6. BACKGROUND-COLOR: WHITE;
7. FONT-SIZE: 16PX;
8. }

ÁREAS DE TEXTO (<TEXTAREA>)

EXEMPLO NO CSS:

1. TEXTAREA {
2. WIDTH: 100%;
3. HEIGHT: 150PX;
4. PADDING: 10PX;
5. BORDER: 2PX SOLID #CCC;
6. BORDER-RADIUS: 5PX;
7. FONT-SIZE: 16PX;
8. RESIZE: VERTICAL; /* PERMITE O REDIMENSIONAMENTO APENAS VERTICALMENTE */
9. }

CAIXA DE TEXTO (<input type="text">)

EXEMPLO NO CSS:

```
1.INPUT[type="text"] {  
2.    width: 300px;  
3.    padding: 10px;  
4.    margin: 10px 0;  
5.    border: 2px solid #ccc;  
6.    border-radius: 5px;  
7.    font-size: 16px;  
8.}  
9.INPUT[type="text"]:focus {  
10.   border-color: #007bff;  
11.   outline: none;  
12.}
```

BOTÕES (<button> e <input type="submit">)

EXEMPLO NO CSS:

```
1.BUTTON, INPUT[type="submit"] {  
2.    background-color: #007bff;  
3.    color: white;  
4.    padding: 10px 20px;  
5.    border: none;  
6.    border-radius: 5px;  
7.    cursor: pointer;  
8.    font-size: 16px;  
9.}  
10.BUTTON:hover, INPUT[type="submit"]:hover {  
11.   background-color: #0056b3;  
12.}
```

CHECKBOXES E RADIO BUTTONS

EXEMPLO NO CSS:

```
1.INPUT[type="checkbox"], INPUT[type="radio"] {  
2.    width: 20px;  
3.    height: 20px;  
4.    cursor: pointer;  
5.    accent-color: #007bff; /* DEFINE A COR DO MARCADOR  
6.    (COMPATÍVEL COM NAVEGADORES MODERNOS) */  
7.}
```

LABELS (<label>)

EXEMPLO NO CSS:

```
1.LABEL {  
2.    font-size: 16px;  
3.    color: #333;  
4.    margin-bottom: 5px;  
5.    display: block;  
6.}
```

AGRUPAMENTO COM FIELDSET E LEGEND

EXEMPLO NO CSS:

```
1.FIELDSET {  
2.    border: 2px solid #ccc;  
3.    padding: 20px;  
4.    border-radius: 5px;  
5.}  
6.LEGEND {  
7.    font-size: 18px;  
8.    font-weight: bold;  
9.    color: #007bff;  
10.}
```

ESTILO GLOBAL PARA FORMULÁRIOS

EXEMPLO NO CSS:

```
1.FORM {  
2.    max-width: 600px;  
3.    margin: 0 auto;  
4.    background-color: #f9f9f9;  
5.    padding: 20px;  
6.    border-radius: 10px;  
7.    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);  
8.}
```



EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
  META CHARSET="UTF-8"
  <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
  <TITLE>ESTILIZANDO FORMULÁRIOS</TITLE>
<STYLE>
  FORM {
    MAX-WIDTH: 600PX;
    MARGIN: 20PX AUTO;
    BACKGROUND-COLOR: #F9F9F9;
    PADDING: 20PX;
    BORDER-RADIUS: 10PX;
    BOX-SHADOW: 0 4PX 6PX RGBA(0, 0, 0, 0.1);
  }

  LABEL {
    FONT-SIZE: 16PX;
    COLOR: #333;
    MARGIN-BOTTOM: 5PX;
    DISPLAY: BLOCK;
  }

  INPUT[TYPE="TEXT"], TEXTAREA, SELECT {
    WIDTH: 100%;
    PADDING: 10PX;
    MARGIN: 10PX 0 20PX;
    BORDER: 2PX SOLID #CCC;
    BORDER-RADIUS: 5PX;
    FONT-SIZE: 16PX;
  }

  INPUT[TYPE="TEXT"]:FOCUS, TEXTAREA:FOCUS, SELECT:FOCUS {
    BORDER-COLOR: #007BFF;
    OUTLINE: NONE;
  }

  BUTTON, INPUT[TYPE="SUBMIT"] {
    BACKGROUND-COLOR: #007BFF;
    COLOR: WHITE;
    PADDING: 10PX 20PX;
    BORDER: NONE;
    BORDER-RADIUS: 5PX;
    CURSOR: POINTER;
    FONT-SIZE: 16PX;
  }

  BUTTON:HOVER, INPUT[TYPE="SUBMIT"]:HOVER {
    BACKGROUND-COLOR: #005683;
  }
}
```

FIELDSET {

```
  BORDER: 2PX SOLID #CCC;
  PADDING: 20PX;
  BORDER-RADIUS: 5PX;
  MARGIN-BOTTOM: 20PX;
}
```

LEGEND {

```
  FONT-SIZE: 18PX;
  FONT-WEIGHT: BOLD;
  COLOR: #007BFF;
}
```

</STYLE>

</HEAD>

<BODY>

<FORM ACTION="#">

<FIELDSET>

<LEGEND>INFORMAÇÕES PESSOAIS</LEGEND>

<LABEL FOR="NAME">NOME:</LABEL>

<INPUT TYPE="TEXT" ID="NAME" NAME="NAME"

PLACEHOLDER="DIGITE SEU NOME">

<LABEL FOR="EMAIL">EMAIL:</LABEL>

<INPUT TYPE="TEXT" ID="EMAIL" NAME="EMAIL"

PLACEHOLDER="DIGITE SEU EMAIL">

</FIELDSET>

<FIELDSET>

<LEGEND>MENSAGEM</LEGEND>

<LABEL FOR="MESSAGE">SUA MENSAGEM:</LABEL>

<TEXTAREA ID="MESSAGE" NAME="MESSAGE"

PLACEHOLDER="ESCREVA AQUI..."></TEXTAREA>

</FIELDSET>

<LABEL FOR="OPTIONS">ESCOLHA UMA OPÇÃO:</LABEL>

<SELECT ID="OPTIONS" NAME="OPTIONS">

<OPTION VALUE="OP1">OPÇÃO 1</OPTION>

<OPTION VALUE="OP2">OPÇÃO 2</OPTION>

<OPTION VALUE="OP3">OPÇÃO 3</OPTION>

</SELECT>

<LABEL>

<INPUT TYPE="CHECKBOX" NAME="AGREE"> EU ACEITO OS TERMOS.

</LABEL>

<BUTTON TYPE="SUBMIT">ENVIAR</BUTTON>

</FORM>

</BODY>

</HTML>

CONTADORES

EXEMPLO DE CONTADORES SIMPLES

```
<!DOCTYPE HTML>
1. <HTML LANG="EN">
2. <HEAD>
3.   <META CHARSET="UTF-8">
4.   <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH,
   INITIAL-SCALE=1.0">
5.   <TITLE>CONTADORES NO CSS</TITLE>
6.   <STYLE>
7.     /* REINICIA O CONTADOR */
8.   BODY {
9.     COUNTER-RESET: SECTION; /* INICIALIZA O CONTADOR
   CHAMADO 'SECTION' */
10.    }
11.
12.   /* INCREMENTA O CONTADOR PARA CADA ELEMENTO H2 */
13.   H2::BEFORE {
14.     COUNTER-INCREMENT: SECTION; /* INCREMENTA O CONTADOR
   */
15.     CONTENT: "SEÇÃO " COUNTER(SECTION) ":"; /* EXIBE O VALOR
   DO CONTADOR */
16.     FONT-WEIGHT: BOLD;
17.     COLOR: #007BFF;
18.   }
19.   </STYLE>
20. </HEAD>
21. <BODY>
22.   <H2>INTRODUÇÃO</H2>
23.   <H2>FUNDAMENTOS</H2>
24.   <H2>EXEMPLO AVANÇADO</H2>
25. </BODY>
26. </HTML>
```

SAÍDA:

SEÇÃO 1: INTRODUÇÃO
SEÇÃO 2: FUNDAMENTOS
SEÇÃO 3: EXEMPLO AVANÇADO

EXEMPLO DE CONTADORES SIMPLES

```
<!DOCTYPE HTML>
1. <HTML LANG="EN">
2. <HEAD>
3.   <META CHARSET="UTF-8">
4.   <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH,
   INITIAL-SCALE=1.0">
5.   <TITLE>CONTADORES ANINHADOS</TITLE>
6.   <STYLE>
7.     /* REINICIA OS CONTADORES */
8.   BODY {
9.     COUNTER-RESET: SECTION;
10.    }
11.   H2 {
12.     COUNTER-RESET: SUBSECTION; /* REINICIA O CONTADOR DE
   SUBSEÇÕES */
13.    }
14.   H2::BEFORE {
15.     COUNTER-INCREMENT: SECTION; /* INCREMENTA O CONTADOR
   DE SEÇÕES */
16.     CONTENT: COUNTER(SECTION) ". "; /* EXIBE O NÚMERO DA
   SEÇÃO */
17.     FONT-WEIGHT: BOLD;
18.     COLOR: #007BFF;
19.   }
20.   </STYLE>
21.   H3::BEFORE {
22.     COUNTER-INCREMENT: SUBSECTION; /* INCREMENTA O
   CONTADOR DE SUBSEÇÕES */
23.     CONTENT: COUNTER(SECTION) "." COUNTER(SUBSECTION) " "; /* EXIBE O NÚMERO DA SUBSEÇÃO */
24.     COLOR: #555;
25.   }
26.   </STYLE>
27. </HEAD>
28. <BODY>
29.   <H2>INTRODUÇÃO</H2>
30.   <H3>CONTEXTO</H3>
31.   <H3>IMPORTÂNCIA</H3>
32.
33.   <H2>FUNDAMENTOS</H2>
34.   <H3>CONCEITOS BÁSICOS</H3>
35.   <H3>ESTRUTURA</H3>
36. </BODY>
37. </HTML>
```

SAÍDA:

1. INTRODUÇÃO
 1.1 CONTEXTO
 1.2 IMPORTÂNCIA
2. FUNDAMENTOS
 2.1 CONCEITOS BÁSICOS
 2.2 ESTRUTURA



LISTAS ORDENADAS PERSONALIZADAS

```

1. <!DOCTYPE HTML>
2. <HTML LANG="en">
3. <HEAD>
4.   <META CHARSET="UTF-8">
5.   <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH,
   INITIAL-SCALE=1.0">
6.   <TITLE>LISTAS COM CONTADORES</TITLE>
7. <STYLE>
8.   OL {
9.     COUNTER-RESET: LIST-ITEM; /* INICIALIZA O CONTADOR PARA
   A LISTA */
10.    LIST-STYLE: NONE; /* REMOVE OS MARCADORES PADRÃO */
11.    PADDING-LEFT: 0;
12.  }
13.  LI::BEFORE {
14.    COUNTER-INCREMENT: LIST-ITEM; /* INCREMENTA O
   CONTADOR PARA CADA ITEM */
15.    CONTENT: COUNTER(LIST-ITEM) ". "; /* EXIBE O VALOR DO
   CONTADOR */
16.    COLOR: #007BFF;
17.    FONT-WEIGHT: BOLD;
18.  }

```

```

1. LI {
2.   MARGIN-BOTTOM: 10PX;
3. }
4. </STYLE>
5. </HEAD>
6. <BODY>
7. <OL>
8.   <LI>ITEM 1</LI>
9.   <LI>ITEM 2</LI>
10.  <LI>ITEM 3</LI>
11. </OL>
12. </BODY>
13. </HTML>

```

SAÍDA:

1. ITEM 1
2. ITEM 2
3. ITEM 3

ESTILO DE CONTADORES

1. CONTENT: COUNTER(SECTION, UPPER-ROMAN); /* EXIBE NÚMEROS ROMANOS */
2. CONTENT: COUNTER(SECTION, LOWER-ALPHA); /* EXIBE LETRAS MINÚSCULAS */
3. CONTENT: COUNTER(SECTION, UPPER-ALPHA); /* EXIBE LETRAS MAIÚSCULAS */

UNIDADES

UNIDADES ABSOLUTAS

AS UNIDADES ABSOLUTAS POSSUEM TAMANHOS FIXOS, INDEPENDENTES DE OUTROS ELEMENTOS OU DO DISPOSITIVO UTILIZADO.

UNIDADE	DESCRIÇÃO	EXEMPLO
px	PIXELS – UNIDADE FIXA EM PIXELS.	WIDTH: 100PX;
cm	CENTÍMETROS.	WIDTH: 5CM;
mm	MILÍMETROS.	WIDTH: 20MM;
in	POLEGADAS (1IN = 96PX).	WIDTH: 2IN;
pt	PONTOS (1PT = 1/72 DE POLEGADA).	FONT-SIZE: 12PT;
pc	PAICAS (1PC = 12PT).	WIDTH: 2PC;

UNIDADES RELATIVAS

AS UNIDADES RELATIVAS DEPENDEM DO TAMANHO DE OUTROS ELEMENTOS OU DA VIEWPORT (ÁREA VISÍVEL DO NAVEGADOR).

UNIDADE	DESCRIÇÃO	EXEMPLO
%	PORCENTAGEM – RELATIVO AO ELEMENTO PAI.	WIDTH: 50%;
EM	RELATIVO AO TAMANHO DA FONTE DO ELEMENTO PAI (1EM = TAMANHO DA FONTE DO ELEMENTO PAI).	FONT-SIZE: 2EM;
REM	RELATIVO AO TAMANHO DA FONTE RAIZ (DEFINIDO NO ELEMENTO <HTML>).	FONT-SIZE: 1.5REM;
VW	VIEWPORT WIDTH – 1% DA LARGURA DA JANELA DO NAVEGADOR.	WIDTH: 50VW;
VH	VIEWPORT HEIGHT – 1% DA ALTURA DA JANELA DO NAVEGADOR.	HEIGHT: 75VH;
VMIN	O MENOR VALOR ENTRE VW E VH.	FONT-SIZE: 10VMIN;
VMAX	O MAIOR VALOR ENTRE VW E VH.	WIDTH: 100VMAX;
EX	RELATIVO À ALTURA DA LETRA "X" NA FONTE ATUAL (GERALMENTE MENOR QUE EM).	LINE-HEIGHT: 2EX;
CH	RELATIVO À LARGURA DO CARACTERE "0" NA FONTE ATUAL.	WIDTH: 10CH;

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>UNIDADES NO CSS</TITLE>
<STYLE>
BODY {
    FONT-SIZE: 16PH;
}
.BOX {
    WIDTH: 50VW; /* 50% DA LARGURA DA JANELA */
    HEIGHT: 30VH; /* 30% DA ALTURA DA JANELA */
    BACKGROUND-COLOR: LIGHTBLUE;
    MARGIN: 5%; /* 5% DO ELEMENTO PAI */
}
.TEXT {
    FONT-SIZE: 2REM; /* 2X O TAMANHO DA FONTE RAIZ */
    PADDING: 1EM; /* RELATIVO AO TAMANHO DA FONTE DO ELEMENTO PAI */
}
</STYLE>
</HEAD>
<BODY>
<DIV CLASS="BOX">
    <P CLASS="TEXT">EXEMPLO DE UNIDADES RELATIVAS</P>
</DIV>
</BODY>
</HTML>
```

FUNÇÕES MATEMÁTICAS

CALC()

- REALIZA CÁLCULOS MATEMÁTICOS PARA PROPRIEDADES CSS QUE ACEITAM VALORES NUMÉRICOS.
- SUPORTA OPERAÇÕES COMO SOMA (+), SUBTRAÇÃO (-), MULTIPLICAÇÃO (*) E DIVISÃO (/).
- USE ESPAÇOS AO REDOR DOS OPERADORES (+, -, *, /) PARA EVITAR ERROS.
- O CALC() ACEITA COMBINAÇÕES DE UNIDADES, COMO %, PX, E EM.

EXEMPLO:

1. DIV {
2. WIDTH: CALC(100% - 50PX); /* SUBTRAI 50PX DA LARGURA TOTAL */
3. HEIGHT: CALC(50VH + 20PX); /* SOMA 20PX À ALTURA DE 50% DA VIEWPORT */
4. }



min()

- RETORNA O MENOR VALOR ENTRE OS FORNECIDOS.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: min(50%, 300px); /* SERÁ 50% DA LARGURA OU NO  
3.   MÁXIMO 300px */  
3. }
```

max()

- RETORNA O MAIOR VALOR ENTRE OS FORNECIDOS.

EXEMPLO:

```
1. DIV {  
2.   WIDTH: max(50%, 300px); /* SERÁ NO MÍNIMO 300px OU 50%, O  
3.   QUE FOR MAIOR */  
3. }
```

clamp()

- DEFINE UM VALOR DENTRO DE UM INTERVALO, COM UM VALOR MÍNIMO, PREFERENCIAL E MÁXIMO.

SINTAXE:

- clamp(min, preferred, max);

EXEMPLO:

```
1. DIV {  
2.   FONT-SIZE: clamp(16px, 5vw, 32px); /* MÍNIMO DE 16px,  
3.   AJUSTÁVEL ATÉ 32px */  
3. }
```

EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>  
<HTML LANG="en">  
<HEAD>  
  <META CHARSET="UTF-8">  
  <META NAME="viewport" CONTENT="width=device-width, initial-  
  scale=1.0">  
  <TITLE>FUNÇÕES MATEMÁTICAS NO CSS</TITLE>  
<STYLE>  
  BODY {  
    margin: 0;  
    font-family: Arial, sans-serif;  
  }  
  
  .CONTAINER {  
    width: calc(100% - 40px); /* LARGURA DINÂMICA COM 20px DE  
    MARGEM DE CADA LADO */  
    height: calc(100vh - 100px); /* ALTURA DINÂMICA MENOS 100px  
    */  
    margin: 20px;  
    background: lightblue;  
  }  
  
  .BOX {  
    width: clamp(200px, 50%, 400px); /* LARGURA MÍNIMA DE 200px,  
    MÁXIMA DE 400px */  
  }  
</STYLE>  
</HEAD>  
<BODY>  
  <DIV CLASS="CONTAINER">  
    <DIV CLASS="BOX">  
      <P CLASS="TEXT">TEXTO RESPONSIVO</P>  
    </DIV>  
  </DIV>  
</BODY>  
</HTML>
```

```
  height: min(50vh, 300px); /* NO MÁXIMO 300px OU 50% DA  
  ALTURA DA VIEWPORT */  
  background: lightcoral;  
  margin: auto;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
}<br>  
.TEXT {  
  font-size: max(16px, 2vw); /* TAMANHO MÍNIMO DE 16px OU 2%  
  DA LARGURA DA VIEWPORT */  
  color: white;  
}  
</style>  
</head>  
<body>  
  <div class="container">  
    <div class="box">  
      <p class="text">texto responsivo</p>  
    </div>  
  </div>  
</body>  
</html>
```

ANOTAÇÕES









```
<form>
  <input type="text" name="us
  Email address" value="{{use
  <input type="password" nam
  placeholder="Password" val
  <input type="checkbox" id=
  <label for="login-remember">
  <button class="btn-tall bt
</form>
{{#error}}
```

A CIÊNCIA DO CSS AVANÇADO

O CSS (CASCADING STYLE SHEETS) AVANÇADO É UMA FERRAMENTA PODEROSA QUE PERMITE TRANSFORMAR A APARIÊNCIA E A USABILIDADE DE PÁGINAS WEB, PROPORCIONANDO EXPERIÊNCIAS VISUAIS IMPACTANTES E RESPONSIVAS. MAIS DO QUE APENAS ESTILIZAR ELEMENTOS BÁSICOS, O CSS AVANÇADO OFERECE POSSIBILIDADES SOFISTICADAS PARA CRIAR DESIGNS MODERNOS E DINÂMICOS.

COM TÉCNICAS COMO ANIMAÇÕES, TRANSIÇÕES, E LAYOUTS FLEXÍVEIS (FLEXBOX E GRID), É POSSÍVEL DESENVOLVER INTERFACES QUE SE ADAPTAM A QUALQUER DISPOSITIVO E RESOLUÇÕES DE TELA. ALÉM DISSO, RECURSOS COMO VARIÁVEIS, PSEUDO-ELEMENTOS E FUNÇÕES MATEMÁTICAS AMPLIAM A FLEXIBILIDADE NO CONTROLE DO DESIGN, GARANTINDO MAIOR CONSISTÊNCIA E EFICIÊNCIA NO CÓDIGO.

O USO DE PREPROCESSADORES COMO SASS E LESS, ALIADO A BOAS PRÁTICAS DE ORGANIZAÇÃO DE ARQUIVOS E MODULARIZAÇÃO, FACILITA A MANUTENÇÃO E ESCALABILIDADE DOS PROJETOS. FERRAMENTAS MODERNAS COMO CUSTOM PROPERTIES (CSS VARIABLES) E MEDIA QUERIES AVANÇADAS TORNAM O CSS UM ALIADO FUNDAMENTAL NA CRIAÇÃO DE INTERFACES RESPONSIVAS E ACESSÍVEIS.

COM O CSS AVANÇADO, DESIGNERS E DESENVOLVEDORES CONSEGUEM TRANSFORMAR IDEIAS EM EXPERIÊNCIAS DIGITAIS ENVOLVENTES, EXPLORANDO A CRIATIVIDADE E ENTREGANDO SOLUÇÕES INOVADORAS PARA UM MUNDO CADA VEZ MAIS VISUAL E CONECTADO.



CANTOS ARREDONDADOS

O BORDER-RADIUS É UMA PROPRIEDADE CSS QUE PERMITE ARREDONDAR OS CANTOS DE UM ELEMENTO, COMO UMA CAIXA (DIV), BOTÃO, IMAGEM OU QUALQUER OUTRO BLOCO DE CONTEÚDO. ESSA PROPRIEDADE É ÚTIL PARA CRIAR ELEMENTOS COM BORDAS SUAVIZADAS, PROPORCIONANDO UM VISUAL MAIS MODERNO E AMIGÁVEL.

SINTAXE:

```
1. ELEMENTO {  
2.   BORDER-RADIUS: VALOR;  
3. }
```

VALORES ACEITOS

VALOR ÚNICO: QUANDO VOCÊ USA UM VALOR ÚNICO, TODOS OS QUATRO CANTOS SERÃO ARREDONDADOS COM O MESMO RAIO.

EXEMPLO:

```
1. DIV {  
2.   BORDER-RADIUS: 10px;  
3. }
```

DOIS VALORES: SE FOREM USADOS DOIS VALORES, O PRIMEIRO SE APLICA AOS CANTOS SUPERIOR ESQUERDO E INFERIOR DIREITO, E O SEGUNDO AOS CANTOS SUPERIOR DIREITO E INFERIOR ESQUERDO.

EXEMPLO:

```
1. DIV {  
2.   BORDER-RADIUS: 10px 20px;  
3. }
```

QUATRO VALORES: USANDO QUATRO VALORES, CADA VALOR SE APLICA A UM CANTO ESPECÍFICO, NA ORDEM: SUPERIOR ESQUERDO, SUPERIOR DIREITO, INFERIOR DIREITO E INFERIOR ESQUERDO.

EXEMPLO:

```
1. DIV {  
2.   BORDER-RADIUS: 10px 20px 30px 40px;  
3. }
```

VALORES COM DIFERENTES UNIDADES: VOCÊ PODE USAR QUALQUER UNIDADE DE MEDIDA, COMO PX, %, EM, REM, ETC.

EXEMPLO:

```
1. DIV {  
2.   BORDER-RADIUS: 50%;  
3. }
```

EFEITOS DE ARREDONDAMENTO

O VALOR DO BORDER-RADIUS PODE SER APLICADO TANTO A ELEMENTOS RETANGULARES QUANTO A ELEMENTOS QUADRADOS, DANDO UM EFEITO DE "CURVATURA" ÀS BORDAS. AQUI ESTÃO ALGUNS EFEITOS COMUNS:

- **CÍRCULO:** QUANDO O ELEMENTO TEM LARGURA E ALTURA IGUAIS E VOCÊ DEFINE BORDER-RADIUS COMO 50%, ELE SE TRANSFORMA EM UM CÍRCULO.
- **ELIPSE:** USANDO DOIS VALORES, PODE-SE CRIAR UMA FORMA OVAL, AJUSTANDO O RAIO HORIZONTAL E VERTICAL DE MANEIRA DIFERENTE.

EXEMPLO

```
<!DOCTYPE HTML>  
<HTML LANG="PT-BR">  
<HEAD>  
  <META CHARSET="UTF-8">  
  <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-  
  SCALE=1.0">  
<TITLE>BORDER RADIUS EXEMPLO</TITLE>  
<STYLE>  
  .CAIXA {  
    WIDTH: 200px;  
    HEIGHT: 200px;  
    BACKGROUND-COLOR: #4CAF50;  
    BORDER-RADIUS: 20px;  
  }  
  .BOTAO {  
    PADDING: 10px 20px;  
    BACKGROUND-COLOR: #008CBA;  
    COLOR: white;  
    BORDER: none;  
    BORDER-RADIUS: 12px;  
  }  
</STYLE>  
</HEAD>  
<BODY>  
  <DIV CLASS="CAIXA"></DIV>  
  <BUTTON CLASS="BOTAO">BOTÃO ARREDONDADO</BUTTON>  
</BODY>  
</HTML>
```

BORDER-IMAGE

A PROPRIEDADE BORDER-IMAGE EM CSS PERMITE APPLICAR UMA IMAGEM COMO BORDA DE UM ELEMENTO, SUBSTITUINDO O TRADICIONAL USO DE CORES SÓLIDAS OU ESTILOS COMO SOLID, DASHED E DOTTED. A IMAGEM É DIVIDIDA EM VÁRIAS SEÇÕES, QUE SÃO APLICADAS ÀS BORDAS SUPERIOR, DIREITA, INFERIOR E ESQUERDA DO ELEMENTO. ISSO PROPORCIONA UM EFEITO DE BORDA MAIS INTERESSANTE E ESTILIZADO.

SINTAXE:

```
1. ELEMENTO {  
2.   BORDER-IMAGE: <IMAGEM> <SLICE> <WIDTH> <OUTSET>  
     <REPEAT>;  
3. }
```

COMPONENTES DA PROPRIEDADE

IMAGEM (<IMAGEM>): O CAMINHO DA IMAGEM QUE SERÁ USADA COMO BORDA. PODE SER UMA URL PARA UMA IMAGEM OU UM VALOR NONE SE VOCÊ NÃO QUISER USAR UMA IMAGEM.

EXEMPLO:

```
1. BORDER-IMAGE: URL('BORDA.PNG');
```

SLICE (<SLICE>): DEFINE COMO A IMAGEM SERÁ DIVIDIDA. ELE ESPECIFICA A ÁREA DA IMAGEM QUE SERÁ USADA PARA AS BORDAS. ESSE VALOR DETERMINA A QUANTIDADE DE PIXEL DA BORDA A SER CORTADA DA IMAGEM. NORMALMENTE, É UMA UNIDADE DE MEDIDA (PX, %, ETC.).

EXEMPLO:

```
1. BORDER-IMAGE: URL('BORDA.PNG') 30%;
```

WIDTH (<WIDTH>): DEFINE A LARGURA DA BORDA CRIADA PELA IMAGEM. PODE SER ESPECIFICADO EM UNIDADES COMO PX, EM, %, ETC.

EXEMPLO:

```
1. BORDER-IMAGE: URL('BORDA.PNG') 30% 15PX;
```

OUTSET (<OUTSET>): ESPECIFICA O QUANTO A BORDA DA IMAGEM SE ESTENDE ALÉM DO ELEMENTO. ELE CRIA UM "AFASTAMENTO" DA BORDA PARA FORA DO CONTEÚDO.

EXEMPLO:

```
1. BORDER-IMAGE: URL('BORDA.PNG') 30% 15PX 10PX;
```

REPEAT (<REPEAT>): CONTROLA COMO A IMAGEM É REPETIDA AO LONGO DAS BORDAS. PODE SER:

- **STRETCH (PADRÃO):** A IMAGEM SERÁ ESTICADA PARA COBRIR TODA A BORDA.
- **REPEAT:** A IMAGEM SERÁ REPETIDA AO LONGO DA BORDA.
- **ROUND:** A IMAGEM SERÁ REPETIDA E AJUSTADA DE MODO QUE SE ENCAIXE PERFECTAMENTE.
- **SPACE:** A IMAGEM SERÁ REPETIDA, MAS COM ESPAÇAMENTO ENTRE AS REPETIÇÕES.

EXEMPLO:

```
1. BORDER-IMAGE: URL('BORDA.PNG') 30% 15PX 10PX REPEAT;
```

EXEMPLO

```
<!DOCTYPE HTML>  
<HTML LANG="PT-BR">  
<HEAD>  
  <META CHARSET="UTF-8">  
  <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-  
  SCALE=1.0">  
  <TITLE>BORDER IMAGE EXEMPLO</TITLE>  
<STYLE>  
  .CAIXA {  
    WIDTH: 300PX;  
    HEIGHT: 200PX;  
    BORDER: 10PX SOLID TRANSPARENT; /* DEFINE A LARGURA DA BORDA  
   */  
    BORDER-IMAGE: URL('HTTPS://VIA.PLACEHOLDER.COM/100') 30%  
    ROUND;  
    PADDING: 20PX;  
    BACKGROUND-COLOR: #F4F4F4;  
  }  
</STYLE>  
</HEAD>  
<BODY>  
  <DIV CLASS="CAIXA">  
    CONTEÚDO DA CAIXA COM BORDA DE IMAGEM.  
  </DIV>  
</BODY>  
</HTML>
```



COLOR KEYWORDS

OS CSS COLOR KEYWORDS SÃO PALAVRAS-CHAVE QUE REPRESENTAM CORES ESPECÍFICAS EM CSS. ELAS SÃO UMA FORMA CONVENIENTE DE DEFINIR CORES SEM PRECISAR ESPECIFICAR VALORES EM FORMATOS COMO HEXADECIMAL, RGB OU HSL. EMBORA O USO DE VALORES COMO #FF5733 OU RGB[255, 87, 51] SEJA COMUM, AS PALAVRAS-CHAVE PARA CORES OFERECEM UMA MANEIRA SIMPLES E LEGÍVEL DE APLICAR CORES.

CORES POR PALAVRA-CHAVE

- **PRETO**
 - BLACK (VALOR: #000000)
- **BRANCO**
 - WHITE (VALOR: #FFFFFF)
- **VERMELHO**
 - RED (VALOR: #FF0000)
- **VERDE**
 - GREEN (VALOR: #008000)
- **AZUL**
 - BLUE (VALOR: #0000FF)
- **CIANO**
 - CYAN (VALOR: #00FFFF)
- **MAGENTA**
 - MAGENTA (VALOR: #FF00FF)
- **AMARELO**
 - YELLOW (VALOR: #FFFF00)
- **CINZA**
 - GRAY (VALOR: #808080)
- **CINZA CLARO**
 - LIGHTGRAY (VALOR: #D3D3D3)
- **LARANJA**
 - ORANGE (VALOR: #FFA500)
- **ROSA**
 - PINK (VALOR: #FFC0CB)

PÚRPURA

- PURPLE (VALOR: #800080)

MARRON

- BROWN (VALOR: #A52A2A)

BEGE

- BEIGE (VALOR: #F5F5DC)

AZUL CLARO

- LIGHTBLUE (VALOR: #ADD8E6)

VERDE CLARO

- LIGHTGREEN (VALOR: #90EE90)

VERMELHO CLARO

- LIGHTCORAL (VALOR: #F08080)

AMARELO CLARO

- LIGHTYELLOW (VALOR: #FFFFE0)

AZUL ESCURO

- DARKBLUE (VALOR: #00008B)

VERDE ESCURO

- DARKGREEN (VALOR: #006400)

VERMELHO ESCURO

- DARKRED (VALOR: #8B0000)

CINZA ESCURO

- DARKGRAY (VALOR: #A9A9A9)

BRANCO NEVE

- SNOW (VALOR: #FFFFFA)

PÊSSEGO

- PEACHPUFF (VALOR: #FFDAB9)

GRADIENTES

OS GRADIENTES CSS SÃO TRANSIÇÕES SUAVES ENTRE DUAS OU MAIS CORES. ELES PERMITEM CRIAR EFEITOS DE COR FLUIDA E CONTÍNUA, SEM A NECESSIDADE DE USAR IMAGENS, E PODEM SER APLICADOS A FUNDOS DE ELEMENTOS, BORDAS E OUTROS ATRIBUTOS.

EM CSS, OS GRADIENTES SÃO CRIADOS UTILIZANDO AS FUNÇÕES `LINEAR-GRADIENT()` E `RADIAL-GRADIENT()`, ENTRE OUTRAS. A VANTAGEM DO GRADIENTE É A FLEXIBILIDADE PARA CRIAR EFEITOS VISUAIS SOFISTICADOS COM POUCO CÓDIGO.

TIPOS DE GRADIENTES

1. GRADIENTE LINEAR (LINEAR-GRADIENT)

O GRADIENTE LINEAR CRIA UMA TRANSIÇÃO DE CORES AO LONGO DE UMA LINHA RETA. VOCÊ PODE ESPECIFICAR A DIREÇÃO DO GRADIENTE (VERTICAL, HORIZONTAL, OU ANGULADA).

SINTAXE:

`BACKGROUND: LINEAR-GRADIENT(DIREÇÃO, COR1, COR2, ...);`

DIREÇÃO: DEFINE A DIREÇÃO DO GRADIENTE. PODE SER UM ÂNGULO OU UMA PALAVRA-CHAVE COMO TO LEFT, TO RIGHT, TO TOP, OU TO BOTTOM.
COR1, COR2, ...: CORES QUE VÃO SER MESCLADAS.

EXEMPLOS:

- **GRADIENTE HORIZONTAL (DA ESQUERDA PARA A DIREITA):**
 - `BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED, YELLOW);`
- **GRADIENTE VERTICAL (DE CIMA PARA BAIXO):**
 - `BACKGROUND: LINEAR-GRADIENT(TO BOTTOM, RED, YELLOW);`
- **GRADIENTE COM ÂNGULO DE 45 GRAUS:**
 - `BACKGROUND: LINEAR-GRADIENT(45deg, RED, YELLOW);`
- **GRADIENTE COM MÚLTIPLAS CORES:**
 - `BACKGROUND: LINEAR-GRADIENT(TO BOTTOM, RED, YELLOW, GREEN);`
- **GRADIENTE COM CORES E PONTOS DE PARADA (PARA CONTROLAR A DISTRIBUIÇÃO DAS CORES):**
 - `BACKGROUND: LINEAR-GRADIENT(TO BOTTOM, RED 0%, YELLOW 50%, GREEN 100%);`

2. GRADIENTE RADIAL (RADIAL-GRADIENT)

O GRADIENTE RADIAL CRIA UMA TRANSIÇÃO DE CORES A PARTIR DE UM PONTO CENTRAL, FORMANDO UM EFEITO CIRCULAR OU ELÍPTICO. VOCÊ PODE CONTROLAR O FORMATO E A POSIÇÃO DO PONTO CENTRAL.

SINTAXE:

`BACKGROUND: RADIAL-GRADIENT(TIPO DE FORMA, POSIÇÃO, COR1, COR2, ...);`

TIPO DE FORMA: PODE SER `CIRCLE` (CÍRCULO) OU `ELLIPSE` (ELÍPSE).

POSIÇÃO: DEFINE O PONTO CENTRAL DO GRADIENTE (PODE SER `CENTER`, `TOP`, `BOTTOM`, OU COORDENADAS ESPECÍFICAS).

COR1, COR2, ...: CORES QUE SERÃO MESCLADAS.

EXEMPLOS:

- **GRADIENTE CIRCULAR (COMEÇANDO DO CENTRO):**
 - `BACKGROUND: RADIAL-GRADIENT(CIRCLE, RED, YELLOW);`
- **GRADIENTE ELÍPTICO:**
 - `BACKGROUND: RADIAL-GRADIENT(ELLIPSE, RED, YELLOW);`
- **GRADIENTE COM MÚLTIPLAS CORES E UM CENTRO ESPECÍFICO:**
 - `BACKGROUND: RADIAL-GRADIENT(CIRCLE AT TOP LEFT, RED, YELLOW, GREEN);`

3. GRADIENTE RADIAL (CONIC-GRADIENT)

O GRADIENTE CÔNICO CRIA UMA TRANSIÇÃO DE CORES AO LONGO DE UM CÍRCULO, COM A POSSIBILIDADE DE DEFINIR A ROTAÇÃO DAS CORES AO REDOR DE UM PONTO CENTRAL.

SINTAXE:

`BACKGROUND: CONIC-GRADIENT(COR1, COR2, ...);`

EXEMPLO:

- `BACKGROUND: CONIC-GRADIENT(RED, YELLOW, GREEN);`



PROPRIEDADES AVANÇADAS

1. PONTOS DE PARADA DE COR (COLOR STOPS)

VOCÊ PODE DEFINIR OS PONTOS DE PARADA PARA CONTROLAR EXATAMENTE ONDE AS CORES COMEÇAM E TERMINAM AO LONGO DO GRADIENTE.

EXEMPLO:

```
BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED 0%, YELLOW 50%, GREEN 100%);
```

2. REPETIÇÃO DE GRADIENTES (REPEATING GRADIENTS)

VOCÊ PODE CRIAR GRADIENTES REPETIDOS USANDO A FUNÇÃO REPEATING-LINEAR-GRADIENT() OU REPEATING-RADIAL-GRADIENT().

EXEMPLO DE GRADIENTE LINEAR REPETIDO:

```
BACKGROUND: REPEATING-LINEAR-GRADIENT(45DEG, RED, YELLOW 10%, RED 20%);
```

EXEMPLO

GRADIENTE LINEAR DE FUNDO DE TELA

```
1. BODY {  
2.   BACKGROUND: LINEAR-GRADIENT(TO RIGHT, #FF5733, #FFC300);  
3.   HEIGHT: 100vh;  
4.   MARGIN: 0;  
5. }
```

GRADIENTE RADIAL DE FUNDO

```
1. DIV {  
2.   BACKGROUND: RADIAL-GRADIENT(CIRCLE, RGBA(255, 0, 0, 1) 0%,  
3.           RGBA(0, 0, 255, 1) 100%);  
4.   WIDTH: 300px;  
5.   HEIGHT: 300px;  
6.   MARGIN: 50px;  
7. }
```

GRADIENTE CÔNICO

```
1. DIV {  
2.   BACKGROUND: CONIC-GRADIENT(RED, YELLOW, GREEN, BLUE);  
3.   WIDTH: 300px;  
4.   HEIGHT: 300px;  
5.   BORDER-RADIUS: 50%;  
6. }
```

EFEITOS DE SOMBRA

BOX SHADOW

A PROPRIEDADE BOX-SHADOW PERMITE APPLICAR SOMBRA AOS ELEMENTOS, COMO CAIXAS, BOTÕES OU QUALQUER OUTRO BLOCO DE CONTEÚDO.

SINTAXE:

- ```
1. ELEMENTO {
2. BOX-SHADOW: DESLOCAMENTO-HORIZONTAL DESLOCAMENTO-
3. VERTICAL DESFOQUE ESPALHAMENTO COR;
```
- DESLOCAMENTO-HORIZONTAL:** DISTÂNCIA DA SOMBRA EM RELAÇÃO AO EIXO X (HORIZONTAL). PODE SER UM VALOR POSITIVO (PARA DIREITA) OU NEGATIVO (PARA ESQUERDA).
  - DESLOCAMENTO-VERTICAL:** DISTÂNCIA DA SOMBRA EM RELAÇÃO AO EIXO Y (VERTICAL). PODE SER UM VALOR POSITIVO (PARA BAIXO) OU NEGATIVO (PARA CIMA).

- DESFOQUE (OPCIONAL):** A QUANTIDADE DE SUAVIZAÇÃO DA SOMBRA. QUANTO MAIOR O VALOR, MAIS SUAVE E DIFUSA A SOMBRA SERÁ.
- ESPALHAMENTO (OPCIONAL):** DEFINE O QUANTO A SOMBRA SE ESPALHA EM RELAÇÃO AO ELEMENTO. UM VALOR NEGATIVO FAZ A SOMBRA ENCOLHER, E UM VALOR POSITIVO FAZ ELA SE EXPANDIR.
- COR:** A COR DA SOMBRA. PODE SER DEFINIDA POR QUALQUER VALOR DE COR VÁLIDO (HEXADECIMAL, RGB, RGBA, ETC.).

#### EXEMPLOS:

- SOMBRA SIMPLES (SEM DESFOQUE OU ESPALHAMENTO):**

```
1. BOX {
2. BOX-SHADOW: 10px 10px black;
3. }
```

- **SOMBRA COM DESFOQUE:**  
 1. .BOX {  
 2. BOX-SHADOW: 10px 10px 15px black;  
 3. }
- **SOMBRA COM ESPALHAMENTO E COR SEMITRASPARENTE (USANDO RGBA):**  
 1. .BOX {  
 2. BOX-SHADOW: 10px 10px 20px 5px rgba(0, 0, 0, 0.5);  
 3. }
- **VÁRIOS EFEITOS DE SOMBRA:**  
 1. .BOX {  
 2. BOX-SHADOW: 2px 2px 4px rgba(0, 0, 0, 0.3), -2px -2px 4px rgba(0, 0, 0, 0.2);  
 3. }

#### EXEMPLO

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>BOX SHADOW</TITLE>
<STYLE>
.BOX {
 WIDTH: 200px;
 HEIGHT: 200px;
 BACKGROUND-COLOR: #3498db;
 BOX-SHADOW: 15px 15px 25px rgba(0, 0, 0, 0.5);
 MARGIN: 50px auto;
}
</STYLE>
</HEAD>
<BODY>
<DIV CLASS="BOX"></DIV>
</BODY>
</HTML>
```

#### TEXT SHADOW

A PROPRIEDADE TEXT-SHADOW PERMITE APLICAR SOMBRA AO TEXTO, CRIANDO EFEITOS DE DESTAQUE E PROFUNDIDADE.

#### SINTAXE:

```
1. ELEMENTO {
2. TEXT-SHADOW: DESLOCAMENTO-HORIZONTAL DESLOCAMENTO-VERTICAL DESFOQUE COR;
3. }
```

- **DESLOCAMENTO-HORIZONTAL:** DISTÂNCIA DA SOMBRA EM RELAÇÃO AO EIXO X (HORIZONTAL). PODE SER UM VALOR POSITIVO (PARA DIREITA) OU NEGATIVO (PARA ESQUERDA).
- **DESLOCAMENTO-VERTICAL:** DISTÂNCIA DA SOMBRA EM RELAÇÃO AO EIXO Y (VERTICAL). PODE SER UM VALOR POSITIVO (PARA BAIXO) OU NEGATIVO (PARA CIMA).
- **DESFOQUE (OPCIONAL):** A QUANTIDADE DE SUAVIZAÇÃO DA SOMBRA. QUANTO MAIOR O VALOR, MAIS SUAVE A SOMBRA SERÁ.
- **COR:** A COR DA SOMBRA. ASSIM COMO EM BOX-SHADOW, VOCÊ PODE USAR QUALQUER VALOR DE COR VÁLIDO.

#### EXEMPLOS:

- **SOMBRA SIMPLES NO TEXTO:**

```
1. H1 {
2. TEXT-SHADOW: 2px 2px 4px #000000;
3. }
```

- **SOMBRA COM DESFOQUE MAIOR:**

```
1. H1 {
2. TEXT-SHADOW: 2px 2px 10px rgba(0, 0, 0, 0.7);
3. }
```

- **SOMBRA EM MÚLTIPLOS NÍVEIS (VÁRIAS SOMBRA NO TEXTO):**

```
1. H1 {
2. TEXT-SHADOW: 1px 1px 2px rgba(0, 0, 0, 0.3), 0 0 25px rgba(0, 0, 0, 0.1);
3. }
```



## EXEMPLO

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>TEXT SHADOW</TITLE>
<STYLE>
H1 {
 FONT-FAMILY: ARIAL, SANS-SERIF;
 FONT-SIZE: 48PX;
 COLOR: #2C3E50;
 TEXT-SHADOW: 3PX 3PX 6PX RGBA(0, 0, 0, 0.3);
}
</STYLE>
</HEAD>
<BODY>
<H1>TEXTO COM SOMBRA</H1>
</BODY>
</HTML>
```

## COMBINANDO EFEITOS DE SOMBRA

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>SOMBRA COMBINADA</TITLE>
<STYLE>
.BOX {
 WIDTH: 250PX;
 HEIGHT: 250PX;
 BACKGROUND-COLOR: #2ECC71;
 BOX-SHADOW: 10PX 10PX 15PX RGBA(0, 0, 0, 0.5);
 MARGIN: 50PX AUTO;
 TEXT-ALIGN: CENTER;
 LINE-HEIGHT: 250PX;
 FONT-SIZE: 24PX;
 COLOR: WHITE;
 TEXT-SHADOW: 2PX 2PX 5PX RGBA(0, 0, 0, 0.7);
}
</STYLE>
</HEAD>
<BODY>
<DIV CLASS="BOX">SOMBRA COMBINADA</DIV>
</BODY>
</HTML>
```

# EFEITOS DE TEXTO

## TRANSFORMAÇÕES DE TEXTO

VOCÊ PODE ALTERAR A APARÊNCIA DO TEXTO DE VÁRIAS MANEIRAS, COMO FAZER O TEXTO MAIÚSCULO, MINÚSCULO OU ESTILIZAR AS LETRAS.

### TRANSFORMAR EM MAIÚSCULAS OU MINÚSCULAS

- **TEXT-TRANSFORM:** CONTROLA A TRANSFORMAÇÃO DO TEXTO PARA MAIÚSCULAS, MINÚSCULAS OU CAPITALIZAÇÃO.

### SINTAXE:

1. ELEMENTO {
2. TEXT-TRANSFORM: UPPERCASE | LOWERCASE | CAPITALIZE | NONE;
3. }

- **UPPERCASE:** CONVERTE TODAS AS LETRAS EM MAIÚSCULAS.
- **LOWERCASE:** CONVERTE TODAS AS LETRAS EM MINÚSCULAS.
- **CAPITALIZE:** CONVERTE A PRIMEIRA LETRA DE CADA PALAVRA PARA MAIÚSCULA.
- **NONE:** NÃO APPLICA TRANSFORMAÇÃO.

### EXEMPLO:

```
1.P {
2. TEXT-TRANSFORM: UPPERCASE; /* TODAS AS LETRAS EM
 Maiúsculas */
3.}
```

## TRANSFORMAÇÕES DE TEXTO (ROTAÇÃO E ESCALA)

VOCÊ PODE ROTACIONAR OU ESCALAR O TEXTO UTILIZANDO A PROPRIEDADE TRANSFORM JUNTO COM ROTATE() E SCALE().

### SINTAXE:

```
1. ELEMENTO {
2. TRANSFORM: ROTATE(ÂNGULO) SCALE(FATOR);
3. }
```

### EXEMPLO DE ROTAÇÃO:

```
1. H1 {
2. TRANSFORM: ROTATE(15DEG); /* ROTACIONA O TEXTO EM 15
GRAUS */
3. }
```

### EXEMPLO DE ESCALA:

```
1. H1 {
2. TRANSFORM: SCALE(1.5); /* AUMENTA O TAMANHO DO TEXTO EM
1.5 VEZES */
3. }
```

## GRADIENTES NO TEXTO

VOCÊ PODE APPLICAR GRADIENTES DE COR AO TEXTO USANDO A PROPRIEDADE BACKGROUND-CLIP E DEFININDO UM GRADIENTE COMO FUNDO.

### SINTAXE:

```
1. ELEMENTO {
2. BACKGROUND: LINEAR-GRADIENT(COR1, COR2, ...);
3. -WEBKIT-BACKGROUND-CLIP: TEXT;
4. COLOR: TRANSPARENT;
5. }
```

- **BACKGROUND-CLIP: TEXT;** FAZ O GRADIENTE APARECER COMO SE FOSSE APPLICADO AO TEXTO.
- **COLOR: TRANSPARENT;** Torna a cor do texto transparente para permitir que o fundo seja visível.

### EXEMPLO:

```
1. H1 {
2. BACKGROUND: LINEAR-GRADIENT(TO RIGHT, RED, YELLOW, GREEN);
3. -WEBKIT-BACKGROUND-CLIP: TEXT;
4. COLOR: TRANSPARENT;
5. }
```

## ANIMAÇÕES NO TEXTO

### ANIMAR A COR DO TEXTO

USANDO A PROPRIEDADE @KEYFRAMES, VOCÊ PODE CRIAR ANIMAÇÕES PARA O TEXTO, COMO MUDANÇAS DE COR.

### SINTAXE:

```
1. @KEYFRAMES MUDARCOR {
2. 0% { COLOR: RED; }
3. 50% { COLOR: YELLOW; }
4. 100% { COLOR: GREEN; }
5. }
6.
7. ELEMENTO {
8. ANIMATION: MUDARCOR 2S INFINITE;
9. }
```

### EXEMPLO:

```
1. @KEYFRAMES CORDOTEXTO {
2. 0% { COLOR: BLUE; }
3. 50% { COLOR: RED; }
4. 100% { COLOR: GREEN; }
5. }
6.
7. H1 {
8. ANIMATION: CORDOTEXTO 3S INFINITE;
9. }
```

### ANIMAR A POSIÇÃO DO TEXTO

VOCÊ PODE MOVER O TEXTO PARA DIFERENTES DIREÇÕES.

### SINTAXE:

```
1. @KEYFRAMES MOVERTEXTO {
2. 0% { TRANSFORM: TRANSLATEX(0); }
3. 100% { TRANSFORM: TRANSLATEX(100PX); }
4. }
5.
6. ELEMENTO {
7. ANIMATION: MOVERTEXTO 2S EASE-IN-OUT;
8. }
```

### EXEMPLO:

```
1. @KEYFRAMES SLIDE {
2. 0% { TRANSFORM: TRANSLATEX(-100%); }
3. 100% { TRANSFORM: TRANSLATEX(0); }
4. }
5.
6. H1 {
7. ANIMATION: SLIDE 1S EASE-OUT;
8. }
```



## ANIMAÇÕES NO TEXTO

OS EFEITOS DE TEXTO TAMBÉM PODEM SER APLICADOS QUANDO O USUÁRIO INTERAGE COM O ELEMENTO, COMO AO PASSAR O MOUSE SOBRE ELE.

### EXEMPLO DE MUDANÇA DE COR NO HOVER:

```
1. H1 {
2. transition: color 0.3s ease;
3. }
4.
5. H1:hover {
6. color: red; /* MUDA A COR DO TEXTO AO PASSAR O MOUSE */
7. }
```

### EXEMPLO DE SOMBRA NO TEXTO COM HOVER:

```
1. H1 {
2. text-shadow: none;
3. transition: text-shadow 0.3s ease;
4. }
5.
6. H1:hover {
7. text-shadow: 3px 3px 5px rgba(0, 0, 0, 0.5); /* ADICIONA
 SOMBRA AO PASSAR O MOUSE */
8. }
```

## ANIMAÇÕES NO TEXTO

OS EFEITOS DE TEXTO TAMBÉM PODEM SER APLICADOS QUANDO O USUÁRIO INTERAGE COM O ELEMENTO, COMO AO PASSAR O MOUSE SOBRE ELE.

### EXEMPLO DE MUDANÇA DE COR NO HOVER:

```
1. H1 {
2. transition: color 0.3s ease;
3. }
4.
5. H1:hover {
6. color: red; /* MUDA A COR DO TEXTO AO PASSAR O MOUSE */
7. }
```

### EXEMPLO DE SOMBRA NO TEXTO COM HOVER:

```
1. H1 {
2. text-shadow: none;
3. transition: text-shadow 0.3s ease;
4. }
5.
6. H1:hover {
7. text-shadow: 3px 3px 5px rgba(0, 0, 0, 0.5); /* ADICIONA
 SOMBRA AO PASSAR O MOUSE */
8. }
```

## EFEITOS DE TEXTO EM TIPOGRAFIA

VOCÊ PODE USAR A PROPRIEDADE FONT-FAMILY PARA MUDAR A TIPOGRAFIA DO TEXTO, O QUE TAMBÉM PODE SER CONSIDERADO UM EFEITO DE TEXTO.

### EXEMPLO:

```
1. H1 {
2. font-family: 'Arial', sans-serif;
3. }
```

VOCÊ TAMBÉM PODE USAR FONTES PERSONALIZADAS COM O @FONT-FACE PARA CARREGAR E APPLICAR FONTES ESPECÍFICAS.

### EXEMPLO DE FONTE PERSONALIZADA:

```
1. @font-face {
2. font-family: 'minhafonte';
3. src: url('minhafonte.woff') format('woff');
4. }
5.
6. H1 {
7. font-family: 'minhafonte', sans-serif;
8. }
```

## SUBLINHA COM ANIMAÇÃO

VOCÊ PODE CRIAR UM EFEITO DE SUBLINHADO ANIMADO USANDO A PROPRIEDADE BORDER-BOTTOM JUNTO COM A ANIMAÇÃO.

### EXEMPLO:

```
1. H1 {
2. position: relative;
3. display: inline-block; }
4.
5. H1::after {
6. content: " ";
7. position: absolute;
8. width: 0;
9. height: 2px;
10. background-color: black;
11. bottom: 0;
12. left: 0;
13. transition: width 0.3s ease; }
14.
15. H1:hover::after {
16. width: 100%; /* A LINHA SE EXPANDE AO PASSAR O MOUSE */ }
```

# TRANSFORMAÇÕES 2D E 3D

## TRANSFORMAÇÕES 2D

AS TRANSFORMAÇÕES 2D NO CSS PERMITEM MANIPULAR ELEMENTOS NO PLANO BIDIMENSIONAL, OU SEJA, EM APENAS DUAS DIREÇÕES: HORIZONTAL (EIXO X) E VERTICAL (EIXO Y).

### PROPRIEDADE TRANSFORM

A PROPRIEDADE TRANSFORM É USADA PARA APLICAR DIFERENTES TIPOS DE TRANSFORMAÇÃO A UM ELEMENTO.

#### SINTAXE:

```
1. ELEMENTO {
2. TRANSFORM: TIPO(VALOR);
3. }
```

AQUI ESTÃO OS TIPOS DE TRANSFORMAÇÕES 2D QUE VOCÊ PODE APLICAR:

### TRANSLATE()

MOVE UM ELEMENTO AO LONGO DO EIXO X E/OU Y.

- **TRANSLATEX()**: MOVE HORIZONTALMENTE.
- **TRANSLATEY()**: MOVE VERTICALMENTE.
- **TRANSLATE()**: MOVE AMBOS OS EIXOS.

#### EXEMPLO:

```
1. BOX {
2. TRANSFORM: TRANSLATE(50px, 100px); /* MOVE A CAIXA 50px À
 DIREITA E 100px PARA BAIXO */
3. }
```

### ROTATE()

GIRA UM ELEMENTO EM TORNO DE UM PONTO FIXO (NORMALMENTE O CENTRO DO ELEMENTO).

- **ROTATE(ÂNGULO)**: O ÂNGULO PODE SER POSITIVO (SENTIDO HORÁRIO) OU NEGATIVO (SENTIDO ANTI-HORÁRIO).

#### EXEMPLO:

```
1. BOX {
2. TRANSFORM: ROTATE(45deg); /* ROTACIONA A CAIXA 45 GRAUS
 NO SENTIDO HORÁRIO */
3. }
```

### SCALE()

AJUSTA O TAMANHO DE UM ELEMENTO.

- **SCALEX()**: ESCALA AO LONGO DO EIXO X.
- **SCALEY()**: ESCALA AO LONGO DO EIXO Y.
- **SCALE()**: ESCALA UNIFORMEMENTE AO LONGO DE AMBOS OS EIXOS.

#### EXEMPLO:

```
1. BOX {
2. TRANSFORM: SCALE(1.5); /* AUMENTA O TAMANHO DA CAIXA EM
 1.5 VEZES */
3. }
```

### SKEW()

INCLINA OU DISTORCE UM ELEMENTO AO LONGO DO EIXO X E/OU Y.

- **SKEWX()**: INCLINA AO LONGO DO EIXO X.
- **SKEWY()**: INCLINA AO LONGO DO EIXO Y.
- **SKEW()**: INCLINA AMBOS OS EIXOS.

#### EXEMPLO:

```
1. BOX {
2. TRANSFORM: SKew(30deg, 20deg); /* INCLINA A CAIXA EM 30
 GRAUS NO EIXO X E 20 GRAUS NO EIXO Y */
3. }
```

### COMBINAÇÃO DE TRANSFORMAÇÕES

VOCÊ PODE COMBINAR VÁRIAS TRANSFORMAÇÕES APLICANDO-AS NA MESMA PROPRIEDADE TRANSFORM, SEPARANDO-AS POR ESPAÇOS.

#### EXEMPLO:

```
1. BOX {
2. TRANSFORM: TRANSLATE(50px, 100px) ROTATE(45deg)
 SCALE(1.5);
3. }
```

## TRANSFORMAÇÕES 3D

AS TRANSFORMAÇÕES 3D NO CSS PERMITEM MANIPULAR ELEMENTOS NO ESPAÇO TRIDIMENSIONAL, ADICIONANDO UM EIXO Z À MANIPULAÇÃO, O QUE DÁ A SENSAÇÃO DE PROFUNDIDADE E PERSPECTIVA.

### PROPRIEDADE TRANSFORM COM 3D

A SINTaxe BÁSICA PARA TRANSFORMAÇÕES 3D É A MESMA QUE AS 2D, MAS VOCÊ USA VALORES PARA O EIXO Z (PROFUNDIDADE).

#### SINTAXE:

```
1. ELEMENTO {
2. TRANSFORM: TIPO3D(VALOR);
3. }
```

AQUI ESTÃO OS TIPOS DE TRANSFORMAÇÕES 3D QUE VOCÊ PODE APLICAR:



## TRANSLATEZ()

MOVE UM ELEMENTO AO LONGO DO EIXO Z, CRIANDO O EFEITO DE PROFUNDIDADE.

- **VALOR POSITIVO** MOVE O ELEMENTO PARA FREnte (EM DIREÇÃO AO ESPECTADOR).
- **VALOR NEGATIVO** MOVE O ELEMENTO PARA TRÁS (AFASTA DO ESPECTADOR).

### EXEMPLO:

```
1..BOX {
 2. TRANSFORM: TRANSLATEZ(100PX); /* MOVE A CAIXA 100PX PARA FORA DA TELA */
 3.}
```

## ROTATEX() E ROTATEY()

ESSAS TRANSFORMAÇÕES GIRAM O ELEMENTO EM TORNO DOS EIXOS X E Y, RESPECTIVAMENTE, CRIANDO UM EFEITO DE ROTAÇÃO 3D.

### EXEMPLO:

```
1..BOX {
 2. TRANSFORM: ROTATEX(45DEG); /* GIRA A CAIXA 45 GRAUS EM TORNO DO EIXO X */
 3.
 4.
 5..BOX {
 6. TRANSFORM: ROTATEY(45DEG); /* GIRA A CAIXA 45 GRAUS EM TORNO DO EIXO Y */
 7.}
```

## PERSPECTIVE()

A PROPRIEDADE PERSPECTIVE CONTROLA A PROFUNDIDADE DO ESPAÇO 3D. QUANDO APlicADA A UM ELEMENTO PAI, ELA DEFINE A DISTÂNCIA DA "CÂMERA" EM RELAÇÃO AOS ELEMENTOS FILHOS, AFETANDO A MANEIRA COMO AS TRANSFORMAÇÕES 3D SÃO PERCEBIDAS.

### SINTAXE:

```
1.ELEMENTO {
 2. PERSPECTIVE: VALOR;
 3.}
```

**VALOR:** A DISTÂNCIA DA CÂMERA EM RELAÇÃO AO ELEMENTO. VALORES MENORES CRIAM UMA PERSPECTIVA MAIS DRAMÁTICA.

### EXEMPLO:

```
1..CONTAINER {
 2. PERSPECTIVE: 500PX; /* DEFINE A DISTÂNCIA DA CÂMERA */
 3.
 4..BOX {
 5. TRANSFORM: ROTATEY(45DEG); }
```

## ROTATE3D()

VOCÊ PODE ROTACIONAR UM ELEMENTO AO REDOR DE QUALQUER EIXO 3D (X, Y, Z).

### SINTAXE:

1. **TRANSFORM: ROTATE3D(EIXOX, EIXOV, EIXOZ, ÂNGULO);**
- **EIXOX, EIXOV, EIXOZ:** VALORES QUE DEFINEM A DIREÇÃO DO EIXO DE ROTAÇÃO.
- **ÂNGULO:** O ÂNGULO DE ROTAÇÃO.

### EXEMPLO:

```
1..BOX {
 2. TRANSFORM: ROTATE3D(1, 1, 0, 45DEG); /* ROTACIONA A CAIXA AO REDOR DOS EIXOS X E Y EM 45 GRAUS */ }
```

## EXEMPLO DE TRANSFORMAÇÃO 3D COMPLETA

```
1.<!DOCTYPE HTML>
2.<HTML LANG="PT-BR">
3.<HEAD>
4. <META CHARSET="UTF-8">
5. <META NAME="viewport" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
6. <TITLE>TRANSFORMAÇÃO 3D</TITLE>
7. <STYLE>
8. .CONTAINER {
9. PERSPECTIVE: 500PX; /* DEFINE A DISTÂNCIA DA CÂMERA */
10. WIDTH: 300PX;
11. HEIGHT: 300PX;
12. MARGIN: 50PX AUTO;
13. BORDER: 2PX SOLID #CCC;
14. POSITION: RELATIVE; }
15.
16. .BOX {
17. WIDTH: 100%;
18. HEIGHT: 100%;
19. BACKGROUND-COLOR: #3498DB;
20. TRANSFORM: ROTATEX(30DEG) ROTATEY(45DEG);
21. TRANSFORM-STYLE: PRESERVE-3D; /* MANTÉM O EFEITO 3D PARA OS FILHOS */
22. TRANSITION: TRANSFORM 1S; }
23.
24. .CONTAINER:HOVER .BOX {
25. TRANSFORM: ROTATEX(90DEG) ROTATEY(180DEG); /* ALTERAÇÃO NO HOVER */ }
26. </STYLE>
27. </HEAD>
28. <BODY>
29. <DIV CLASS="CONTAINER">
30. <DIV CLASS="BOX"></DIV>
31. </DIV>
32. </BODY>
33. </HTML>
```

# TRANSIÇÕES

## SINTAXE BÁSICA

A PROPRIEDADE TRANSITION É USADA PARA DEFINIR UMA TRANSIÇÃO EM CSS.

### SINTAXE:

```
1. ELEMENTO {
2. TRANSITION: PROPRIEDADE DURAÇÃO TIMING-FUNCTION DELAY;
3. }
```

- **PROPRIEDADE:** A PROPRIEDADE CSS A SER ANIMADA. PODE SER UMA ÚNICA PROPRIEDADE (EX: BACKGROUND-COLOR) OU ALL (APLICA A TRANSIÇÃO A TODAS AS PROPRIEDADES QUE MUDAM).
- **DURAÇÃO:** A DURAÇÃO DA TRANSIÇÃO. PODE SER DEFINIDA EM SEGUNDOS (S) OU MILISSEGUNDOS (MS).
- **TIMING-FUNCTION:** DEFINE O RITMO DA TRANSIÇÃO. AS OPÇÕES MAIS COMUNS SÃO:
  - **EASE:** INICIA DEVAGAR, ACELERA E DESACELERA.
  - **LINEAR:** A TRANSIÇÃO OCORRE COM UMA VELOCIDADE CONSTANTE.
  - **EASE-IN:** INICIA DEVAGAR E ACELERA.
  - **EASE-OUT:** ACELERA E DESACELERA.
  - **EASE-IN-OUT:** COMEÇA DEVAGAR, ACELERA E DESACELERA NO FINAL.
  - **CUBIC-BEZIER(0,0,0,0):** DEFINE UMA CURVA DE BEZIER PERSONALIZADA.
- **DELAY:** ATRASO ANTES DA TRANSIÇÃO COMEÇAR. PODE SER DEFINIDO EM SEGUNDOS OU MILISSEGUNDOS.

### EXEMPLO:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. TRANSITION: background-color 0.5s ease-in-out;
6. }
7.
8. .BOX:hover {
9. BACKGROUND-COLOR: blue; /* A COR DO FUNDO MUDARÁ1
 SUAVEMENTE AO PASSAR O MOUSE */
10. }
```

## PROPRIEDADES QUE PODEM SER ANIMADAS

A MAIORIA DAS PROPRIEDADES CSS PODE SER ANIMADA COM TRANSIÇÕES. ALGUMAS DAS MAIS COMUNS INCLuem:

- **CORES** (EX: COLOR, BACKGROUND-COLOR, BORDER-COLOR)
- **TAMANHO** (EX: WIDTH, HEIGHT, FONT-SIZE)
- **ESPAÇAMENTO** (EX: PADDING, MARGIN)
- **POSIÇÃO** (EX: LEFT, RIGHT, TOP, BOTTOM)
- **TRANSFORMAÇÕES** (EX: ROTATE, SCALE, TRANSLATE)
- **OPACIDADE** (EX: OPACITY)

## EXEMPLO DE TRANSIÇÃO SIMPLES:

NESTE EXEMPLO, AO PASSAR O MOUSE SOBRE A CAIXA, ELA AUMENTARÁ DE TAMANHO E MUDARÁ A COR SUAVEMENTE:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. TRANSITION: all 0.3s ease-in-out; /* APlica a transição a
 todas as propriedades */
6. }
7.
8. .BOX:hover {
9. WIDTH: 150px;
10. HEIGHT: 150px;
11. BACKGROUND-COLOR: blue;
12. }
```

## TRANSIÇÕES COM MÚLTIPHAS PROPRIEDADES

VOCÊ PODE ANIMAR VÁRIAS PROPRIEDADES AO MESMO TEMPO. PARA ISSO, BASTA SEPARAR CADA TRANSIÇÃO COM UMA VÍRGULA.

### EXEMPLO:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. TRANSITION: width 0.5s, height 0.5s, background-color 0.5s;
6. }
7.
8. .BOX:hover {
9. WIDTH: 200px;
10. HEIGHT: 200px;
11. BACKGROUND-COLOR: green;
12. }
```



## TRANSIÇÕES COM ATRASO

VOCÊ PODE ADICIONAR UM DELAY (ATRASO) ANTES QUE A TRANSIÇÃO COMECE A SER EXECUTADA.

### EXEMPLO:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. TRANSITION: all 0.5s ease-in-out 1s; /* A TRANSIÇÃO
 COMEÇARÁ APÓS 1 SEGUNDO */
6. }
7.
8. .BOX:hover {
9. WIDTH: 150px;
10. HEIGHT: 150px;
11. BACKGROUND-COLOR: blue;
12. }
```

## TRANSIÇÕES EM TRANSFORMAÇÕES

VOCÊ PODE USAR TRANSIÇÕES EM TRANSFORMAÇÕES CSS, COMO ROTAÇÃO, ESCALA, E MOVIMENTAÇÃO.

### EXEMPLO:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. TRANSITION: transform 0.5s ease-in-out;
6. }
7.
8. .BOX:hover {
9. TRANSFORM: rotate(45deg) scale(1.5); /* A CAIXA GIRA E
 AUMENTA DE TAMANHO SUAVEMENTE */
10. }
```

## TRANSIÇÕES EM OPACIDADE

UMA TRANSIÇÃO DE OPACIDADE É COMUMENTE USADA PARA CRIAR EFEITOS DE FADE-IN (APARECER) E FADE-OUT (DESAPARECER).

### EXEMPLO:

```
1. .BOX {
2. WIDTH: 100px;
3. HEIGHT: 100px;
4. BACKGROUND-COLOR: red;
5. OPACITY: 0;
6. TRANSITION: opacity 1s ease-in-out;
7. }
8.
9. .BOX:hover {
10. OPACITY: 1; /* TORNA A CAIXA COMPLETAMENTE VISÍVEL AO
 PASSAR O MOUSE */
11. }
```

## TRANSIÇÕES EM PSEUDO-CLASSES E PSEUDO-ELEMENTOS

AS TRANSIÇÕES TAMBÉM FUNCIONAM EM PSEUDO-CLASSES COMO :HOVER, :FOCUS E PSEUDO-ELEMENTOS COMO ::BEFORE E ::AFTER.

### EXEMPLO:

```
1. .BUTTON {
2. POSITION: relative;
3. PADDING: 10px 20px;
4. BACKGROUND-COLOR: #3498db;
5. COLOR: white;
6. BORDER: none;
7. FONT-SIZE: 16px; }
8.
9. .BUTTON::after {
10. CONTENT: " ";
11. POSITION: absolute;
12. BOTTOM: 0;
13. LEFT: 0;
14. WIDTH: 100%;
15. HEIGHT: 2px;
16. BACKGROUND-COLOR: #298089;
17. TRANSFORM: scale(0);
18. TRANSFORM-ORIGIN: bottom right;
19. TRANSITION: transform 0.4s ease; }
20.
21. .BUTTON:hover::after {
22. TRANSFORM: scale(1); /* ANIMA A LINHA DE BAIXO AO PASSAR
 O MOUSE */ }
```

# ANIMAÇÕES

## SINTAXE BÁSICA

PARA CRIAR ANIMAÇÕES CSS, USAMOS A PROPRIEDADE @KEYFRAMES PARA DEFINIR OS PONTOS DE INÍCIO E FIM DA ANIMAÇÃO, ALÉM DAS TRANSFORMAÇÕES INTERMEDIÁRIAS. A ANIMAÇÃO EM SI É APLICADA AO ELEMENTO COM A PROPRIEDADE ANIMATION.

### SINTAXE:

```
1. @KEYFRAMES NOME-DA-ANIMACAO {
2. 0% {
3. /* ESTADO INICIAL */
4. }
5. 50% {
6. /* ESTADO INTERMEDIÁRIO (OPCIONAL) */
7. }
8. 100% {
9. /* ESTADO FINAL */
10. }
11. }
12.
13. ELEMENTO {
14. ANIMATION: NOME-DA-ANIMACAO DURACAO TIMING-FUNCTION
 DELAY ITERATION-COUNT DIRECTION FILL-MODE;
15. }
```

- **@KEYFRAMES:** DEFINE OS DIFERENTES ESTÁGIOS DA ANIMAÇÃO.
  - 0%: INÍCIO DA ANIMAÇÃO.
  - 100%: FIM DA ANIMAÇÃO.
  - PODEMOS TAMBÉM USAR PORCENTAGENS INTERMEDIÁRIAS COMO 50% PARA TRANSIÇÕES PARCIAIS.
- **ANIMATION:** APPLICA A ANIMAÇÃO AO ELEMENTO.
  - **DURACAO:** TEMPO QUE A ANIMAÇÃO LEVA PARA SER CONCLUÍDA.
  - **TIMING-FUNCTION:** DEFINE A ACELERAÇÃO/DESACELERAÇÃO DA ANIMAÇÃO (EX: EASE, LINEAR).
  - **DELAY:** ATRASO ANTES DE A ANIMAÇÃO COMEÇAR.
  - **ITERATION-COUNT:** QUANTAS VEZES A ANIMAÇÃO SE REPETE.
  - **DIRECTION:** DEFINE A DIREÇÃO DA ANIMAÇÃO (EX: NORMAL, REVERSE).
  - **FILL-MODE:** DEFINE O COMPORTAMENTO DA ANIMAÇÃO APÓS A EXECUÇÃO (EX: FORWARDS, BACKWARDS).

## EXEMPLO BÁSICO DE ANIMAÇÃO

AQUI ESTÁ UM EXEMPLO DE UMA ANIMAÇÃO SIMPLES QUE FAZ UMA CAIXA MUDAR DE COR E SE MOVER DE UM LADO PARA O OUTRO:

```
1. @KEYFRAMES MOVERCOR {
2. 0% {
3. BACKGROUND-COLOR: RED;
4. TRANSFORM: TRANSLATEX(0);
5. }
6. 50% {
7. BACKGROUND-COLOR: YELLOW;
8. TRANSFORM: TRANSLATEX(100px);
9. }
10. 100% {
11. BACKGROUND-COLOR: BLUE;
12. TRANSFORM: TRANSLATEX(0);
13. }
14. }
15.
16. BOX {
17. WIDTH: 100px;
18. HEIGHT: 100px;
19. BACKGROUND-COLOR: RED;
20. ANIMATION: MOVERCOR 3s EASE-IN-OUT INFINITE; /* ANIMAÇÃO
 DE 3 SEGUNDOS, COM REPETIÇÃO INFINITA */
21. }
```

## PROPRIEDADES DE ANIMAÇÃO

### ANIMATION-NAME

DEFINE O NOME DA ANIMAÇÃO, QUE DEVE CORRESPONDER AO NOME DADO NA REGRAS @KEYFRAMES.

### EXEMPLO:

```
1. BOX {
2. ANIMATION-NAME: MOVERCOR; }
```

### ANIMATION-DURATION

ESPECIFICA A DURAÇÃO DA ANIMAÇÃO, OU SEJA, O TEMPO QUE ELA LEVA PARA SER CONCLUÍDA.

### EXEMPLO:

```
1. BOX {
2. ANIMATION-DURATION: 3s; /* A ANIMAÇÃO DURARÁ 3 SEGUNDOS
 */ }
```



## ANIMATION-TIMING-FUNCTION

CONTROLA A ACELERAÇÃO DA ANIMAÇÃO, OU SEJA, A VELOCIDADE AO LONGO DO TEMPO. ALGUMAS OPÇÕES SÃO:

- **LINEAR**: A ANIMAÇÃO TEM UMA VELOCIDADE CONSTANTE.
- **EASE**: A ANIMAÇÃO COMEÇA DEVAGAR, ACELERA NO MEIO E DESACELERA NO FINAL.
- **EASE-IN**: A ANIMAÇÃO COMEÇA DEVAGAR E ACELERA.
- **EASE-OUT**: A ANIMAÇÃO ACELERA E DESACELERA NO FINAL.
- **EASE-IN-OUT**: COMEÇA E TERMINA DEVAGAR, COM ACELERAÇÃO NO MEIO.
- **CUBIC-BEZIER(n, n, n, n)**: PERMITE CRIAR UMA CURVA PERSONALIZADA.

### EXEMPLO:

```
1. .BOX {
 2. ANIMATION-TIMING-FUNCTION: EASE-IN-OUT;
 3. }
```

## ANIMATION-DELAY

DEFINE UM ATRASO ANTES DE INICIAR A ANIMAÇÃO.

### EXEMPLO:

```
1. .BOX {
 2. ANIMATION-DELAY: 1S; /* A ANIMAÇÃO COMEÇARÁ 1 SEGUNDO
 APÓS O CARREGAMENTO */
 3. }
```

## ANIMATION-ITERATION-COUNT

DEFINE QUANTAS VEZES A ANIMAÇÃO DEVE SER REPETIDA. PODE SER UM NÚMERO INTEIRO OU O VALOR INFINITE PARA REPETIR INDEFINIDAMENTE.

### EXEMPLO:

```
1. .BOX {
 2. ANIMATION-ITERATION-COUNT: INFINITE; /* A ANIMAÇÃO SE
 REPETE INFINITAMENTE */
 3. }
```

## ANIMATION-DIRECTION

DEFINE A DIREÇÃO DA ANIMAÇÃO. ALGUMAS OPÇÕES INCLuem:

- **NORMAL**: A ANIMAÇÃO SEGUe O FLUXO NORMAL.
- **REVERSE**: A ANIMAÇÃO ACONTECE AO CONTRÁRIO.
- **ALTERNATE**: A ANIMAÇÃO ALTERNA ENTRE O FLUXO NORMAL E REVERSO.
- **ALTERNATE-REVERSE**: A ANIMAÇÃO ALTERNA ENTRE O REVERSO E O FLUXO NORMAL.

### EXEMPLO:

```
1. .BOX {
 2. ANIMATION-DIRECTION: ALTERNATE; /* A ANIMAÇÃO ALTERNA DE
 NORMAL PARA REVERSO */
 3. }
```

## ANIMATION-FILL-MODE

CONTROLA O COMPORTAMENTO DO ELEMENTO QUANDO A ANIMAÇÃO TERMINA. ALGUMAS OPÇÕES SÃO:

- **FORWARDS**: MANTÉM OS ESTILOS FINAIS DA ANIMAÇÃO.
- **BACKWARDS**: APLICA OS ESTILOS INICIAIS DA ANIMAÇÃO.
- **BOTH**: APLICA OS ESTILOS DE AMBOS OS EXTREMOS.
- **NONE**: NÃO MANTÉM NENHUM ESTILO APÓS A ANIMAÇÃO.

### EXEMPLO:

```
1. .BOX {
 2. ANIMATION-FILL-MODE: FORWARDS; /* A CAIXA MANTERÁ O
 ESTILO FINAL DA ANIMAÇÃO */
 3. }
```

## ANIMAÇÕES EM LOOP

PARA REPETIR A ANIMAÇÃO INDEFINIDAMENTE, PODEMOS USAR INFINITE EM ANIMATION-ITERATION-COUNT.

### EXEMPLO:

```
1. @KEYFRAMES GIRAR {
 2. 0% {
 3. TRANSFORM: ROTATE(0DEG);
 4. }
 5. 100% {
 6. TRANSFORM: ROTATE(360DEG); /* ROTACIONA O ELEMENTO 360
 GRAUS */
 7. }
 8. }
 9.
 10. .BOX {
 11. WIDTH: 100px;
 12. HEIGHT: 100px;
 13. BACKGROUND-COLOR: green;
 14. ANIMATION: GIRAR 2S LINEAR INFINITE; /* ROTACIONA
 INDEFINIDAMENTE */
 15. }
```

# EFEITOS DE FILTRO DE IMAGEM

## PROPRIEDADE FILTER

A PROPRIEDADE FILTER APPLICA EFEITOS GRÁFICOS DIRETAMENTE NO ELEMENTO. ELA PODE SER COMBINADA COM OUTROS FILTROS, APLICANDO MÚLTIPLOS EFEITOS AO MESMO TEMPO.

### SINTAXE:

```
1. ELEMENTO {
2. FILTER: EFEITO(VALOR);
3. }
```

## PRINCIPAIS EFEITOS DE FILTRO

### BLUR() – DESFOQUE (BORRÃO)

APLICA UM DESFOQUE NA IMAGEM OU NO ELEMENTO. O VALOR É A QUANTIDADE DE DESFOQUE, E É MEDIDO EM PIXELS.

### EXEMPLO:

```
1. IMG {
2. FILTER: BLUR(5px); /* APLICA UM DESFOQUE DE 5 PIXELS */
3. }
```

### BRIGHTNESS() – BRILHO

AJUSTA O BRILHO DA IMAGEM. O VALOR É UM NÚMERO QUE PODE SER MAIOR OU MENOR QUE 1:

- 1: MANTÉM O BRILHO ORIGINAL.
- 0: Torna a imagem completamente preta.
- Valores maiores que 1 aumentam o brilho.

### EXEMPLO:

```
1. IMG {
2. FILTER: BRIGHTNESS(1.5); /* AUMENTA O BRILHO EM 50% */
3. }
```

### CONTRAST() – CONTRASTE

AJUSTA O CONTRASTE DA IMAGEM. O VALOR FUNCIONA DE MANEIRA SEMELHANTE AO BRILHO:

- 1: MANTÉM O CONTRASTE ORIGINAL.
- 0: FAZ A IMAGEM FICAR EM TONS DE CINZA.
- Valores maiores que 1 aumentam o contraste.

### EXEMPLO:

```
1. IMG {
2. FILTER: CONTRAST(1.2); /* AUMENTA O CONTRASTE EM 20% */
3. }
```

### GRAYSCALE() – ESCALA DE CINZA

TRANSFORMA A IMAGEM PARA TONS DE CINZA. O VALOR PODE SER:

- 0%: A IMAGEM NÃO SERÁ ALTERADA.
- 100%: A IMAGEM SERÁ CONVERTIDA COMPLETAMENTE PARA PRETO E BRANCO.

### EXEMPLO:

```
1. IMG {
2. FILTER: GRayscale(100%); /* CONVERTE A IMAGEM PARA PRETO E
3. BRANCO */
```

### HUE-ROTATE() – ROTAÇÃO DE MATIZ

AJUSTA A MATIZ (COR) DA IMAGEM. O VALOR É UM ÂNGULO EM GRAUS (DE 0 A 360), QUE GIRA A TONALIDADE DAS CORES.

### EXEMPLO:

```
1. IMG {
2. FILTER: HUE-ROTATE(90deg); /* GIRA A MATIZ DA IMAGEM EM 90
3. GRAUS */
```

### INVERT() – INVERSÃO DE CORES

INverte as cores da imagem. O valor pode ser:

- 0%: MANTÉM AS CORES ORIGINAIS.
- 100%: INverte todas as cores da imagem.

### EXEMPLO:

```
1. IMG {
2. FILTER: INVERT(100%); /* INverte as cores da imagem */
3. }
```

### OPACITY() – OPACIDADE

AJUSTA A OPACIDADE (TRANSPARÊNCIA) DA IMAGEM. O VALOR PODE SER DE 0 (TOTALMENTE TRANSPARENTE) A 1 (TOTALMENTE OPACO).

### EXEMPLO:

```
1. IMG {
2. FILTER: OPACITY(0.5); /* Torna a imagem 50% transparente */
3. }
```



## SATURATE() – SATURAÇÃO

AJUSTA A SATURAÇÃO DAS CORES DA IMAGEM. O VALOR FUNCIONA DE MANEIRA SEMELHANTE AO BRILHO E AO CONTRASTE:

- 1: MANTÉM A SATURAÇÃO ORIGINAL.
- 0: Torna a imagem em preto e branco.
- Valores maiores que 1 aumentam a saturação.

### EXEMPLO:

```
1. img {
2. filter: saturate(2); /* aumenta a saturação em 100% */
3. }
```

## SEPIA() – SÉPIA

APLICA UM FILTRO SÉPIA, DANDO À IMAGEM UM TOM AMARELADO. O VALOR PODE SER DE 0% (SEM EFEITO) A 100% (EFEITO SÉPIA COMPLETO).

### EXEMPLO:

```
1. img {
2. filter: sepia(100%); /* aplica o efeito sépia completo */
3. }
```

## COMBINANDO FILTROS

VOCÊ PODE COMBINAR VÁRIOS FILTROS PARA CRIAR EFEITOS MAIS COMPLEXOS, SEPARANDO-OS POR ESPAÇOS.

### EXEMPLO DE COMBINAÇÃO:

```
1. img {
2. filter: grayscale(100%) brightness(1.2) contrast(1.5);
3. /* Aplica escala de cinza, aumenta o brilho em 20% e
 aumenta o contraste em 50% */
4. }
```

## FILTROS EM BACKGROUNDS

FILTROS TAMBÉM PODEM SER APLICADOS EM IMAGENS DE FUNDO COM A PROPRIEDADE BACKGROUND-IMAGE.

### EXEMPLO:

```
1. elemento {
2. width: 300px;
3. height: 200px;
4. background-image: url('image.jpg');
5. filter: sepia(50%) contrast(120%);
6. }
```

## EXEMPLO COMPLETO

AQUI ESTÁ UM EXEMPLO DE COMO APLICAR MÚLTIPLOS FILTROS A UMA IMAGEM, CRIANDO UM EFEITO DINÂMICO:

### EXEMPLO:

```
1. <!DOCTYPE html>
2. <html lang="pt-br">
3. <head>
4. <meta charset="UTF-8">
5. <meta name="viewport" content="width=device-width, initial-scale=1.0">
6. <title>Filtros CSS</title>
7. <style>
8. img {
9. width: 300px;
10. height: auto;
11. filter: blur(4px) grayscale(50%) brightness(1.2);
12. transition: filter 0.5s ease-in-out;
13. }
14.
15. img:hover {
16. filter: blur(0) grayscale(0) brightness(1);
17. }
18. </style>
19. </head>
20. <body>
21.
22. </body>
23. </html>
```

### NESTE EXEMPLO:

- Quando o mouse passa sobre a imagem, ela volta ao seu estado normal.
- Inicialmente, a imagem possui desfoque, escala de cinza e brilho aumentado.

# FORMAS DE IMAGEM

No CSS, é possível manipular imagens e outros elementos para que assumam formas personalizadas, como círculos, elipses, polígonos e outras formas complexas. Isso é feito principalmente com a combinação de propriedades como border-radius, clip-path, shape-outside e mask.

## CORTES PERSONALIZADOS COM CLIP-PATH

A propriedade clip-path é usada para criar formas mais complexas, como triângulos, hexágonos e formas personalizadas, ao "cortar" parte do elemento.

### A. FORMAS COM CLIP-PATH

Você pode usar valores como circle, ellipse, polygon ou uma URL de SVG para definir a forma.

#### EXEMPLO DE TRIÂNGULO:

```
1. img {
2. clip-path: polygon(50% 0%, 0% 100%, 100% 100%);
3. width: 300px;
4. }
```

#### EXEMPLO DE CÍRCULO:

```
1. img {
2. clip-path: circle(50% at 50% 50%);
3. width: 300px;
4. }
```

#### EXEMPLO DE ESTRELA:

```
1. img {
2. clip-path: polygon(50% 0%, 61% 35%, 98% 35%, 68% 57%, 79%
 91%, 50% 70%, 21% 91%, 32% 57%, 2% 35%, 39% 35%);
3. width: 300px;
4. }
```

## FORMAS BÁSICAS COM BORDER-RADIUS

A propriedade border-radius permite criar bordas arredondadas em elementos, o que pode ser usado para transformar imagens em círculos, elipses ou outras formas arredondadas.

## criando um círculo

Para criar um círculo, o elemento precisa ser quadrado e ter o border-radius configurado como 50%.

#### EXEMPLO:

```
1. img {
2. width: 200px;
3. height: 200px;
4. border-radius: 50%;
5. }
```

## criando uma elipse

Para criar uma elipse, o elemento deve ter larguras e alturas diferentes e o border-radius deve ser configurado como 50%.

#### EXEMPLO:

```
1. img {
2. width: 300px;
3. height: 150px;
4. border-radius: 50%;
5. }
```

## MASCARAMENTO COM MASK

A propriedade mask permite usar uma imagem ou gradiente como uma máscara para exibir apenas partes específicas de um elemento.

#### EXEMPLO:

```
1. img {
2. width: 300px;
3. height: auto;
4. mask-image: url('mascara.png');
5. -webkit-mask-image: url('mascara.png'); /*
 compatibilidade com webkit */
6. }
```



## TEXTOS ENVOLVENDO FORMAS COM SHAPE-OUTSIDE

A PROPRIEDADE SHAPE-OUTSIDE DEFINE A FORMA QUE O TEXTO AO REDOR DE UM ELEMENTO SEGUIRÁ. GERALMENTE É USADA COM ELEMENTOS FLUTUANTES (FLOAT).

### EXEMPLO:

```
1.IMG {
2. FLOAT: LEFT;
3. SHAPE-OUTSIDE: CIRCLE(50%);
4. WIDTH: 200px;
5. HEIGHT: 200px;
6. BORDER-RADIUS: 50%;
7. MARGIN: 20px;
8.}
```

NESTE EXEMPLO, O TEXTO AO LADO DA IMAGEM SERÁ AJUSTADO EM VOLTA DO CÍRCULO.

## COMBINAÇÃO DE PROPRIEDADES

AS PROPRIEDADES CLIP-PATH, BORDER-RADIUS, SHAPE-OUTSIDE E MASK PODEM SER COMBINADAS PARA CRIAR EFEITOS VISUAIS IMPRESSIONANTES.

### EXEMPLO:

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
 SCALE=1.0">
<TITLE>FORMAS DE IMAGEM COM CSS</TITLE>
<STYLE>
.CIRCLE {
 WIDTH: 200px;
 HEIGHT: 200px;
 BORDER-RADIUS: 50%;
 OVERFLOW: HIDDEN;
}

.CLIP {
 WIDTH: 300px;
 HEIGHT: AUTO;
 CLIP-PATH: POLYGON(50% 0%, 0% 100%, 100% 100%);
}

.MASK {
 WIDTH: 300px;
 HEIGHT: AUTO;
```

```
MASK-IMAGE: URL('MASCARA.PNG');
-WEBKIT-MASK-IMAGE: URL('MASCARA.PNG'); /* COMPATIBILIDADE
COM WEBKIT */
}
```

```
.TEXT-WRAP {
 FLOAT: LEFT;
 SHAPE-OUTSIDE: CIRCLE(50%);
 WIDTH: 200px;
 HEIGHT: 200px;
 BORDER-RADIUS: 50%;
 MARGIN-RIGHT: 20px;
}
</STYLE>
</HEAD>
<BODY>
<DIV>
 <IMG CLASS="CIRCLE" SRC="HTTPS://VIA.PLACEHOLDER.COM/200"
 ALT="CÍRCULO">
 <IMG CLASS="CLIP" SRC="HTTPS://VIA.PLACEHOLDER.COM/300"
 ALT="CLIP-PATH TRIÂNGULO">
 <IMG CLASS="MASK" SRC="HTTPS://VIA.PLACEHOLDER.COM/300"
 ALT="MÁSCARA">
</DIV>
<DIV>
 <IMG CLASS="TEXT-WRAP"
 SRC="HTTPS://VIA.PLACEHOLDER.COM/200" ALT="TEXTO AO REDOR">
</P>
 ESTE É UM EXEMPLO DE TEXTO QUE ENVOLVE A IMAGEM CIRCULAR.
 VOCÊ PODE USAR `SHAPE-OUTSIDE` PARA AJUSTAR COMO O TEXTO SE
 POSICIONA AO REDOR DA IMAGEM.
</P>
</DIV>
</BODY>
</HTML>
```

# PROPRIEDADE OBJECT-FIT

A PROPRIEDADE OBJECT-FIT NO CSS É USADA PARA CONTROLAR COMO O CONTEÚDO DE UM ELEMENTO SUBSTITUÍDO, COMO IMAGENS, VÍDEOS OU <CANVAS>, SE AJUSTA AO ESPAÇO DO SEU CONTÊINER. ESSA PROPRIEDADE É PARTICULARMENTE ÚTIL PARA AJUSTAR IMAGENS OU VÍDEOS SEM DISTORCER OU PERDER PROPORÇÕES.

## SINTAXE

```
1. ELEMENTO {
2. OBJECT-FIT: VALOR;
3. }
```

## VALORES POSSÍVEIS

### FILL (PADRÃO)

- PREENCHE COMPLETAMENTE O CONTÊINER, PODENDO DISTORCER O CONTEÚDO.
- ESTE É O COMPORTAMENTO PADRÃO SE OBJECT-FIT NÃO FOR ESPECIFICADO.

### EXEMPLO:

```
1. IMG {
2. OBJECT-FIT: FILL; }
```

### CONTAIN

- FAZ COM QUE O CONTEÚDO SEJA REDIMENSIONADO PARA CABER TOTALMENTE DENTRO DO CONTÊINER, MANTENDO A PROPORÇÃO ORIGINAL.
- PODE DEIXAR ESPAÇOS VAZIOS (BARRAS LATERAIS OU SUPERIORES/INFERIORES) NO CONTÊINER.

### EXEMPLO:

```
1. IMG {
2. OBJECT-FIT: CONTAIN; }
```

### COVER

- FAZ COM QUE O CONTEÚDO PREENCHA O CONTÊINER COMPLETAMENTE, MANTENDO A PROPORÇÃO ORIGINAL.
- PARTE DO CONTEÚDO PODE SER CORTADA PARA EVITAR DISTORÇÕES.

### EXEMPLO:

```
1. IMG {
2. OBJECT-FIT: COVER; }
```

### NONE

- O CONTEÚDO MANTÉM SUAS DIMENSÕES ORIGINAIS E NÃO É REDIMENSIONADO PARA CABER NO CONTÊINER.
- O CONTEÚDO PODE "VAZAR" DO CONTÊINER.

### EXEMPLO:

```
1. IMG {
2. OBJECT-FIT: NONE;
3. }
```

### SCALE-DOWN

- COMPARA O COMPORTAMENTO DE NONE E CONTAIN, APLICANDO O MENOR DELES:
  - SE O CONTEÚDO JÁ CABE NO CONTÊINER, ELE NÃO É REDIMENSIONADO (NONE).
  - CASO CONTRÁRIO, ELE É AJUSTADO COMO CONTAIN.

### EXEMPLO:

```
1. IMG {
2. OBJECT-FIT: SCALE-DOWN;
3. }
```

## USO PRÁTICO

A PROPRIEDADE OBJECT-FIT É FREQUENTEMENTE USADA EM ELEMENTOS COMO IMAGENS RESPONSIVAS OU THUMBNAILS EM LAYOUTS MODERNOS.

### EXEMPLO: AJUSTANDO IMAGENS DE DIFERENTES PROPORÇÕES

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
 SCALE=1.0">
<TITLE>EXEMPLO DE OBJECT-FIT</TITLE>
<STYLE>
 .CONTAINER {
 WIDTH: 300PX;
 HEIGHT: 200PX;
 BORDER: 2PX SOLID #CCC;
 OVERFLOW: HIDDEN;
 }
</STYLE>
```



```

.FILL {
 width: 100%;
 height: 100%;
 object-fit: fill;
}

.CONTAIN {
 width: 100%;
 height: 100%;
 object-fit: contain;
}

.COVER {
 width: 100%;
 height: 100%;
 object-fit: cover;
}

.NONE {
 width: 100%;
 height: 100%;
 object-fit: none;
}

.SCALE-DOWN {
 width: 100%;
 height: 100%;
 object-fit: scale-down;
}

```

```

</STYLE>
</HEAD>
<BODY>
<H2>EXEMPLO DE OBJECT-FIT</H2>
<DIV CLASS="CONTAINER">
 <IMG CLASS="FILL" SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300"
ALT="FILL">
</DIV>
<DIV CLASS="CONTAINER">
 <IMG CLASS="CONTAIN"
SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="CONTAIN">
</DIV>
<DIV CLASS="CONTAINER">
 <IMG CLASS="COVER"
SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="COVER">
</DIV>
<DIV CLASS="CONTAINER">
 <IMG CLASS="NONE"
SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="NONE">
</DIV>
<DIV CLASS="CONTAINER">
 <IMG CLASS="SCALE-DOWN"
SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="SCALE-DOWN">
</DIV>
</BODY>
</HTML>

```

## PROPRIEDADE OBJECT-POSITION

A PROPRIEDADE OBJECT-POSITION NO CSS É USADA PARA DEFINIR A POSIÇÃO DE UM ELEMENTO SUBSTITUÍDO (COMO UMA IMAGEM OU VÍDEO) DENTRO DO SEU CONTÊINER. ESSA PROPRIEDADE É PARTICULARMENTE ÚTIL QUANDO COMBINADA COM OBJECT-FIT, POIS PERMITE CONTROLAR COMO O CONTEÚDO DESLOCADO É EXIBIDO.

### SINTAXE

1. ELEMENTO {
2. OBJECT-POSITION: VALOR-X VALOR-Y;
3. }

- VALOR-X: DEFINE O POSICIONAMENTO HORIZONTAL (EIXO X).
- VALOR-Y: DEFINE O POSICIONAMENTO VERTICAL (EIXO Y).



## VALORES ACEITOS

### POSIÇÕES PRÉ-DEFINIDAS

- CENTER (PADRÃO): CENTRALIZA O CONTEÚDO NO CONTÊINER.
- TOP, BOTTOM: POSICIONA O CONTEÚDO NO TOPO OU NA PARTE INFERIOR.
- LEFT, RIGHT: POSICIONA O CONTEÚDO À ESQUERDA OU À DIREITA.

### EXEMPLO:

```
1.IMG {
2. OBJECT-POSITION: TOP LEFT; /* POSICIONA NO CANTO SUPERIOR
3. ESQUERDO */
3.}
```

## VALORES DE COMPRIMENTO (PIXELS OU PORCENTAGENS)

VOCÊ PODE USAR UNIDADES COMO PX, EM, OU % PARA POSICIONAR O CONTEÚDO.

- PORCENTAGEM:
  - 0% CORRESPONDE À BORDA INICIAL (ESQUERDA OU SUPERIOR).
  - 100% CORRESPONDE À BORDA FINAL (DIREITA OU INFERIOR).
  - 50% CENTRALIZA.
- UNIDADES DE COMPRIMENTO:
  - UM VALOR COMO 20PX DESLOCA O CONTEÚDO DA BORDA.

### EXEMPLO:

```
1.IMG {
2. OBJECT-POSITION: 50% 20%; /* CENTRALIZA HORIZONTALMENTE E
3. MOVE 20% PARA BAIXO */
3.}
```

## QUANDO USAR OBJECT-POSITION

A PROPRIEDADE É ÚTIL PARA AJUSTAR O FOCO OU A PARTE VISÍVEL DO CONTEÚDO, ESPECIALMENTE QUANDO COMBINADO COM OBJECT-FIT: COVER, ONDE PARTE DO CONTEÚDO PODE SER CORTADA.

### EXEMPLO PRÁTICO:

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
<TITLE>EXEMPLO DE OBJECT-POSITION</TITLE>
<STYLE>
.CONTAINER {
 WIDTH: 300PX;
 HEIGHT: 200PX;
```

```
BORDER: 2PX SOLID #CCC;
OVERFLOW: HIDDEN;
}
```

```
.DEFAULT {
 OBJECT-FIT: COVER;
 OBJECT-POSITION: CENTER;
}
```

```
.TOP-LEFT {
 OBJECT-FIT: COVER;
 OBJECT-POSITION: TOP LEFT;
}
```

```
.BOTTOM-RIGHT {
 OBJECT-FIT: COVER;
 OBJECT-POSITION: BOTTOM RIGHT;
}
```

```
.CUSTOM {
 OBJECT-FIT: COVER;
 OBJECT-POSITION: 30% 70%; /* AJUSTE ESPECÍFICO */
}
```

```
</STYLE>
```

```
</HEAD>
```

```
<BODY>
```

```
<H2>EXEMPLO DE OBJECT-POSITION</H2>
```

```
<DIV CLASS="CONTAINER">
```

```
 <IMG CLASS="DEFAULT"
```

```
 SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="CENTER
(DEFAULT)">
```

```
</DIV>
```

```
<DIV CLASS="CONTAINER">
```

```
 <IMG CLASS="TOP-LEFT"
```

```
 SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="TOP LEFT">
```

```
</DIV>
```

```
<DIV CLASS="CONTAINER">
```

```
 <IMG CLASS="BOTTOM-RIGHT"
```

```
 SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="BOTTOM RIGHT">
```

```
</DIV>
```

```
<DIV CLASS="CONTAINER">
```

```
 <IMG CLASS="CUSTOM"
```

```
 SRC="HTTPS://VIA.PLACEHOLDER.COM/500X300" ALT="CUSTOM
POSITION">
```

```
</DIV>
```

```
</BODY>
```

```
</HTML>
```



# MASCARAMENTO

O MASCARAMENTO EM CSS É UMA TÉCNICA QUE PERMITE ESCONDER OU REVELAR PARTES DE UM ELEMENTO USANDO UMA MÁSCARA. ISSO É FEITO COM AS PROPRIEDADES MASK E CLIP-PATH, QUE PODEM USAR FORMAS, GRADIENTES OU IMAGENS COMO MÁSCARAS.

## MASCARAMENTO COM MASK

A PROPRIEDADE MASK DEFINE UMA MÁSCARA PARA UM ELEMENTO, ONDE APENAS AS PARTES VISÍVEIS DA MÁSCARA SERÃO EXIBIDAS.

### SINTAXE:

1. ELEMENTO {
2. MASK: [IMAGEM OU GRADIENTE];
3. MASK-REPEAT: [VALOR];
4. MASK-POSITION: [VALOR];
5. MASK-SIZE: [VALOR];
6. }

### PROPRIEDADES RELACIONADAS:

- **MASK-IMAGE:** DEFINE A IMAGEM OU GRADIENTE A SER USADA COMO MÁSCARA.
- **MASK-REPEAT:** CONTROLA COMO A MÁSCARA SE REPETE (VALORES COMO REPEAT, NO-REPEAT, ETC.).
- **MASK-POSITION:** DEFINE A POSIÇÃO DA MÁSCARA.
- **MASK-SIZE:** AJUSTA O TAMANHO DA MÁSCARA.

### EXEMPLO:

1. DIV {
2. WIDTH: 300px;
3. HEIGHT: 300px;
4. BACKGROUND: URL('https://via.placeholder.com/300x300');
5. MASK-IMAGE: RADIAL-GRADIENT(CIRCLE, RGBA(0, 0, 0, 1) 50%,  
RGBA(0, 0, 0, 0) 100%);
6. MASK-REPEAT: NO-REPEAT;
7. MASK-POSITION: CENTER;
8. MASK-SIZE: CONTAIN;
9. }

**RESULTADO:** A IMAGEM DE FUNDO SERÁ VISÍVEL APENAS DENTRO DE UM GRADIENTE CIRCULAR NO CENTRO.

## MASCARAMENTO COM CLIP-PATH

A PROPRIEDADE CLIP-PATH É USADA PARA DEFINIR UMA ÁREA DE CORTE NO ELEMENTO, PERMITINDO FORMAS COMO TRIÂNGULOS, CÍRCULOS, POLÍGONOS OU FORMAS PERSONALIZADAS.

### SINTAXE:

1. ELEMENTO {
2. CLIP-PATH: [FORMA OU URL];
3. }

### FORMAS COMUNS:

- **CIRCLE()**: DEFINE UM CÍRCULO.
- **ELLIPSE()**: DEFINE UMA ELIPSE.
- **POLYGON()**: DEFINE UM POLÍGONO COM PONTOS ESPECÍFICOS.
- **INSET()**: DEFINE UMA ÁREA RETANGULAR COM RECUOS.

### EXEMPLO:

1. DIV {
2. WIDTH: 300px;
3. HEIGHT: 300px;
4. BACKGROUND: URL('https://via.placeholder.com/300x300');
5. CLIP-PATH: CIRCLE(50% AT 50% 50%); /\* CÍRCULO CENTRALIZADO \*/
6. }

## SUporte a PREFIXOS

ALGUNS NAVEGADORES EXIGEM O PREFIXO -WEBKIT- PARA A PROPRIEDADE MASK.

### EXEMPLO COM PREFIXO:

1. DIV {
2. MASK-IMAGE: URL('MASCARA.PNG');
3. -WEBKIT-MASK-IMAGE: URL('MASCARA.PNG'); /\* PARA SAFARI E CHROME \*/
4. }

### EXEMPLO COMPLETO

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>MASCARAMENTO CSS</TITLE>
<STYLE>
.MASK-EXAMPLE {
 WIDTH: 300PX;
 HEIGHT: 300PX;
 BACKGROUND: URL('HTTPS://VIA.PLACEHOLDER.COM/300X300') NO-REPEAT CENTER/COVER;
 MASK-IMAGE: RADIAL GRADIENT(CIRCLE, RGBA(0, 0, 0, 1) 50%, RGBA(0, 0, 0, 0) 100%);
 -WEBKIT-MASK-IMAGE: RADIAL GRADIENT(CIRCLE, RGBA(0, 0, 0, 1) 50%, RGBA(0, 0, 0, 0) 100%);
}
}
```

```
.CLIP-EXAMPLE {
 WIDTH: 300PX;
 HEIGHT: 300PX;
 BACKGROUND: URL('HTTPS://VIA.PLACEHOLDER.COM/300X300') NO-REPEAT CENTER/COVER;
 CLIP-PATH: POLYGON(50% 0%, 100% 100%, 0% 100%);
}
</STYLE>
</HEAD>
<BODY>
<H1>EXEMPLO DE MASCARAMENTO CSS</H1>
<DIV CLASS="MASK-EXAMPLE"></DIV>
<H2>COM CLIP-PATH</H2>
<DIV CLASS="CLIP-EXAMPLE"></DIV>
</BODY>
</HTML>
```

## PAGINAÇÃO

A PAGINAÇÃO É UM ELEMENTO DE NAVEGAÇÃO MUITO UTILIZADO EM SITES E APLICATIVOS PARA DIVIDIR O CONTEÚDO EM PÁGINAS. ELA PODE SER ESTILIZADA COM CSS PARA CRIAR UMA APARÊNCIA PROFISSIONAL E RESPONSIVA. VOU MOSTRAR ALGUNS EXEMPLOS PRÁTICOS E MODERNOS DE PAGINAÇÃO.

### PAGINAÇÃO BÁSICA

ESTE EXEMPLO CRIA UMA BARRA DE PAGINAÇÃO SIMPLES COM NÚMEROS CLICÁVEIS.

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>PAGINAÇÃO BÁSICA</TITLE>
<STYLE>
.PAGINATION {
 DISPLAY: FLEX;
 LIST-STYLE: NONE;
 PADDING: 0;
}
```

```
MARGIN: 20PX;
```

```
}
```

```
.PAGINATION LI {
```

```
 MARGIN: 0 5PX;
```

```
}
```

```
.PAGINATION A {
```

```
 TEXT-DECORATION: NONE;
```

```
 PADDING: 8PX 12PX;
```

```
 BORDER: 1PX SOLID #CCC;
```

```
 BORDER-RADIUS: 4PX;
```

```
 COLOR: #333;
```

```
}
```

```
.PAGINATION A:HOVER {
```

```
 BACKGROUND-COLOR: #F0F0F0;
```

```
 BORDER-COLOR: #999;
```

```
}
```

```
.PAGINATION .ACTIVE A {
```

```
 BACKGROUND-COLOR: #007BFF;
```

```
 COLOR: #FFF;
```

```
 BORDER-COLOR: #007BFF;
```



```

 }
</STYLE>
</HEAD>
<BODY>
<UL CLASS="PAGINATION">
&LAQUO;
1
<LI CLASS="ACTIVE">2
3
&RAQUO;

</BODY>
</HTML>

```

#### PAGINAÇÃO COM INDICADORES DE ATIVAÇÃO

NESTE EXEMPLO, DESTACAMOS A PÁGINA ATIVA E ADICIONAMOS TRANSIÇÕES SUAVES.

```

<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
<TITLE>PAGINAÇÃO COM INDICADORES</TITLE>
<STYLE>
.PAGINATION {
 DISPLAY: FLEX;
 JUSTIFY-CONTENT: CENTER;
 LIST-STYLE: NONE;
 PADDING: 10PX;
}

.PAGINATION LI {
 MARGIN: 0 5PX;
}

.PAGINATION A {
 TEXT-DECORATION: NONE;
 PADDING: 10PX 15PX;
 BORDER-RADIUS: 50%;
 COLOR: #555;
 FONT-WEIGHT: BOLD;
 TRANSITION: BACKGROUND-COLOR 0.3S, COLOR 0.3S;
}

```

```

.PAGINATION A:HOVER {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
}

.PAGINATION .ACTIVE A {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
 POINTER-EVENTS: NONE;
}

</STYLE>
</HEAD>
<BODY>
<UL CLASS="PAGINATION">
&LAQUO;
1
<LI CLASS="ACTIVE">2
3
&RAQUO;

</BODY>
</HTML>

```

#### PAGINAÇÃO RESPONSIVA

ESTE EXEMPLO ADAPTA A PAGINAÇÃO AO TAMANHO DA TELA, ESCONDENDO NÚMEROS QUANDO NECESSÁRIO.

```

<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
<TITLE>PAGINAÇÃO RESPONSIVA</TITLE>
<STYLE>
.PAGINATION {
 DISPLAY: FLEX;
 FLEX-WRAP: WRAP;
 JUSTIFY-CONTENT: CENTER;
 LIST-STYLE: NONE;
 PADDING: 10PX;
}

.PAGINATION LI {
 MARGIN: 5PX;
}

```

```

.PAGINATION A {
 TEXT-DECORATION: NONE;
 PADDING: 8PX 12PX;
 BORDER: 1PX SOLID #CCC;
 BORDER-RADIUS: 5PX;
 COLOR: #333;
}

.PAGINATION A:HOVER {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
}

.PAGINATION .ACTIVE A {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
 BORDER: 1PX SOLID #007BFF;
}

@MEDIA (MAX-WIDTH: 600PX) {
 .PAGINATION LI:NTH-CHILD(N+4):NOT(.ACTIVE) {
 DISPLAY: NONE;
 }
}
</STYLE>
</HEAD>
<BODY>
<UL CLASS="PAGINATION">
 &LAQUO;
 1
 2
 <LI CLASS="ACTIVE">3
 4
 5
 &RAQUO;

</BODY>
</HTML>

```

### PAGINAÇÃO COM BOTÕES PERSONALIZADOS

ADICIONA BOTÕES COM ÍCONES PARA UM LAYOUT MAIS SOFISTICADO.

```

<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
 <TITLE>PAGINAÇÃO COM ÍCONES</TITLE>
 <STYLE>
 .PAGINATION {
 DISPLAY: FLEX;
 ALIGN-ITEMS: CENTER;
 JUSTIFY-CONTENT: CENTER;
 LIST-STYLE: NONE;
 PADDING: 10PX;
 }

 .PAGINATION LI {
 MARGIN: 0 5PX;
 }

 .PAGINATION A {
 TEXT-DECORATION: NONE;
 PADDING: 10PX;
 BACKGROUND-COLOR: #F0F0F0;
 BORDER-RADIUS: 5PX;
 COLOR: #333;
 DISPLAY: FLEX;
 ALIGN-ITEMS: CENTER;
 JUSTIFY-CONTENT: CENTER;
 TRANSITION: BACKGROUND-COLOR 0.3S;
 }

 .PAGINATION A:HOVER {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
 }

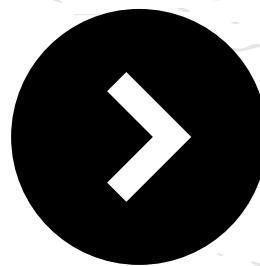
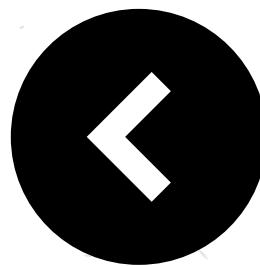
 .PAGINATION .ACTIVE A {
 BACKGROUND-COLOR: #007BFF;
 COLOR: #FFF;
 POINTER-EVENTS: NONE;
 }
 </STYLE>
</HEAD>

```



```
<BODY>
<UL CLASS="PAGINATION">
&LAQUO;
1
<LI CLASS="ACTIVE">2
3
&RAQUO;

</BODY>
</HTML>
```



## COLUNAS MÚLTIPLAS

O CSS COLUNAS MÚLTIPLAS (OU MULTI-COLUMN LAYOUT) É UMA TÉCNICA QUE PERMITE DIVIDIR O CONTEÚDO DE UM ELEMENTO EM VÁRIAS COLUNAS, SIMILAR AO QUE SE VÊ EM JORNais OU REVISTAS. ISSO É FEITO USANDO PROPRIEDADES ESPECÍFICAS DO CSS PARA CRIAR E CONTROLAR O LAYOUT DE COLUNAS.

### PROPRIEDADES PRINCIPAIS

#### COLUMN-COUNT

DEFINE O NÚMERO DE COLUNAS EM QUE O CONTEÚDO SERÁ DIVIDIDO.

#### EXEMPLO:

```
1. ELEMENTO {
 2. COLUMN-COUNT: 3; /* DIVIDE O CONTEÚDO EM 3 COLUNAS */
 3. }
```

#### COLUMN-WIDTH

ESPECIFICA A LARGURA IDEAL DE CADA COLUNA. O NAVEGADOR AJUSTARÁ O NÚMERO DE COLUNAS COM BASE NO ESPAÇO DISPONÍVEL.

#### EXEMPLO:

```
1. ELEMENTO {
 2. COLUMN-WIDTH: 200px; /* CADA COLUNA TERÁ CERCA DE 200px
 DE LARGURA */
 3. }
```

#### COLUMNS

PROPRIEDADE ABREVIADA QUE COMBINA COLUMN-COUNT E COLUMN-WIDTH.

#### EXEMPLO:

```
1. ELEMENTO {
 2. COLUMNS: 3 200px; /* 3 COLUNAS DE APROXIMADAMENTE 200px
 */
 3. }
```

#### COLUMN-GAP

CONTROLA O ESPAÇO ENTRE AS COLUNAS.

#### EXEMPLO:

```
1. ELEMENTO {
 2. COLUMN-GAP: 20px; /* 20px DE ESPAÇO ENTRE AS COLUNAS */
 3. }
```

#### COLUMN-RULE

DEFINE UMA LINHA (REGRA) ENTRE AS COLUNAS.

PROPRIEDADES RELACIONADAS:

- COLUMN-RULE-WIDTH: LARGURA DA LINHA.
- COLUMN-RULE-STYLE: ESTILO DA LINHA (SÓLIDA, PONTILHADA, ETC).
- COLUMN-RULE-COLOR: COR DA LINHA.

#### EXEMPLO:

```
1. ELEMENTO {
 2. COLUMN-RULE: 2px solid #007bff; /* LINHA DE 2px, SÓLIDA E
 AZUL */
 3. }
```

## BREAK-INSIDE

CONTROLA QUEBRAS DE CONTEÚDO DENTRO DE UMA COLUNA.

VALORES COMUNS:

- AUTO (PADRÃO): PERMITE QUEBRAS.
- AVOID: EVITA QUEBRAS DENTRO DA COLUNA.

## EXEMPLO:

```
1.ELEMENTO {
2. BREAK-INSIDE: AVOID; /* EVITA QUEBRA DE PARÁGRAFOS EM
3. COLUNAS */
}
```

## EXEMPLO DE COLUNAS MÚLTIPHAS

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
 <TITLE>COLUNAS MÚLTIPHAS</TITLE>
 <STYLE>
 .MULTICOLUMN {
 COLUMNS: 3 200px; /* 3 COLUNAS COM LARGURA APROXIMADA DE
 200px */
 COLUMN-GAP: 20px; /* ESPAÇO DE 20px ENTRE AS COLUNAS */
 COLUMN-RULE: 2px solid #007bff; /* LINHA AZUL ENTRE AS
 COLUNAS */
 }
 </STYLE>
</HEAD>
<BODY>
 <DIV CLASS="MULTICOLUMN">
 <P>LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT.
 VIVAMUS LACINIA ODIO VITAE VESTIBULUM.</P>
 <P>INTEGER VEL AUGUE NEC JUSTO PHARETRA PHARETRA. NULLAM
 EGET MAGNA NON ELIT MALESUADA VARIUS.</P>
 <P>PELLENTESQUE HABITANT MORBI TRISTIQUE SENECTUS ET NETUS
 ET MALESUADA FAMES AC TURPIS EGESTAS.</P>
 <P>CURABITUR VEL NUNC AC MAURIS EFFICITUR FEUGIAT NEC AC
 LECTUS. PROIN CONVALLIS IPSUM IN DIAM.</P>
 </DIV>
</BODY>
</HTML>
```

## PERSONALIZAÇÕES

CONTROLAR LARGURA FLEXÍVEL COM COLUMN-WIDTH:

```
1.ELEMENTO {
2. COLUMN-WIDTH: 250px; /* ADAPTA O NÚMERO DE COLUNAS AO
3. ESPAÇO DISPONÍVEL */
}
```

ADICIONAR ESTILOS ENTRE COLUNAS:

```
1.ELEMENTO {
2. COLUMN-RULE: 1px dashed gray; /* LINHA TRACEJADA ENTRE
3. COLUNAS */
}
```

QUEBRAS CONTROLADAS COM BREAK-INSIDE:

```
1.ELEMENTO > P {
2. BREAK-INSIDE: AVOID; /* EVITA QUE O PARÁGRAFO SEJA
3. QUEBRADO */
}
```

## EXEMPLO RESPONSIVO

COMBINE MEDIA QUERIES PARA AJUSTAR O LAYOUT DAS COLUNAS EM DIFERENTES TAMANHOS DE TELA.

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
 <TITLE>COLUNAS RESPONSIVAS</TITLE>
 <STYLE>
 .MULTICOLUMN {
 COLUMNS: 3 200px;
 COLUMN-GAP: 20px;
 }

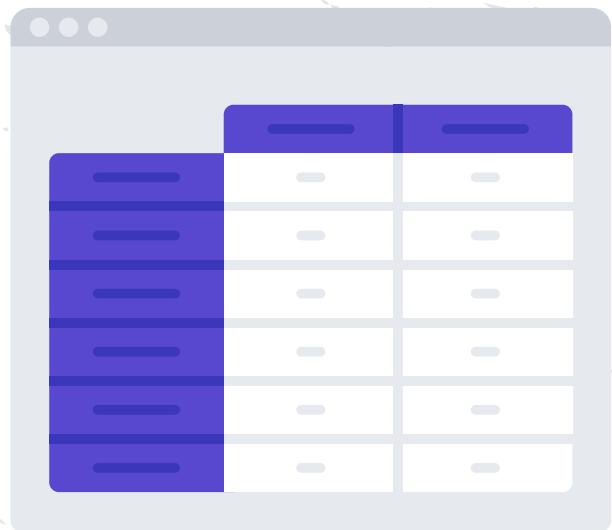
 @MEDIA (MAX-WIDTH: 768px) {
 .MULTICOLUMN {
 COLUMNS: 2 150px; /* REDUZ PARA 2 COLUNAS EM TELAS
 MENORES */
 }
 }
 </STYLE>
</HEAD>
<BODY>
 <DIV CLASS="MULTICOLUMN">
 <P>LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT.
 VIVAMUS LACINIA ODIO VITAE VESTIBULUM.</P>
 <P>INTEGER VEL AUGUE NEC JUSTO PHARETRA PHARETRA. NULLAM
 EGET MAGNA NON ELIT MALESUADA VARIUS.</P>
 <P>PELLENTESQUE HABITANT MORBI TRISTIQUE SENECTUS ET NETUS
 ET MALESUADA FAMES AC TURPIS EGESTAS.</P>
 <P>CURABITUR VEL NUNC AC MAURIS EFFICITUR FEUGIAT NEC AC
 LECTUS. PROIN CONVALLIS IPSUM IN DIAM.</P>
 </DIV>
</BODY>
</HTML>
```



```

@MEDIA (MAX-WIDTH: 480PH) {
 .MULTICOLUMN {
 COLUMNS: 1; /* APENAS 1 COLUNA EM DISPOSITIVOS PEQUENOS */
 }
}
</STYLE>
</HEAD>
<BODY>
<DIV CLASS="MULTICOLUMN">
 <P>LOREM IPSUM DOLOR SIT AMET...</P>
 <P>INTEGER VEL AUGUE NEC JUSTO PHARETRA PHARETRA...</P>
 <P>PELLENTESQUE HABITANT MORBI TRISTIQUE...</P>
 <P>CURABITUR VEL NUNC AC MAURIS EFFICITUR FEUGIAT...</P>
</DIV>
</BODY>
</HTML>

```



## VARIÁVEIS

AS VARIÁVEIS CSS (TAMBÉM CONHECIDAS COMO CUSTOM PROPERTIES) SÃO UMA MANEIRA PODEROSA DE ARMAZENAR E REUTILIZAR VALORES NO CSS, PERMITINDO MAIOR FLEXIBILIDADE E ORGANIZAÇÃO NO CÓDIGO. ELAS SÃO DEFINIDAS USANDO A NOTAÇÃO --NOME-DA-VARIÁVEL E PODEM SER USADAS EM QUALQUER LUGAR DO DOCUMENTO CSS.

### COMO FUNCIONA?

#### DEFINIÇÃO DA VARIÁVEL

AS VARIÁVEIS SÃO DEFINIDAS DENTRO DE UM SELETOR (GERALMENTE :ROOT PARA APLICAÇÃO GLOBAL):

#### EXEMPLO:

1. :ROOT {
2. --COR-PRIMARIA: #007bff;
3. --TAMANHO-FONTE: 16PH;
4. --ESPACO: 10PH;
5. }

#### USO DA VARIÁVEL

VOCÊ UTILIZA A FUNÇÃO VAR() PARA ACESSAR O VALOR DE UMA VARIÁVEL:

#### EXEMPLO:

```

BODY {
 BACKGROUND-COLOR: VAR(--COR-PRIMARIA);
 FONT-SIZE: VAR(--TAMANHO-FONTE);
 MARGIN: VAR(--ESPACO);
}

```

#### VALORES DE BACKUP

VOCÊ PODE ADICIONAR UM VALOR PADRÃO CASO A VARIÁVEL NÃO ESTEJA DEFINIDA:

#### EXEMPLO:

1. P {
2. COLOR: VAR(--COR-SECUNDARIA, #333); /\* USA #333 SE --COR-SECUNDARIA NÃO ESTIVER DEFINIDA \*/
3. }

### EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>VARIÁVEIS CSS</TITLE>
<STYLE>
:ROOT {
--COR-PRIMARIA: #007bff;
--COR-SECUNDARIA: #6c757d;
--TAMANHO-FONTE: 16px;
--PADDING: 20px;
}

BODY {
 font-family: Arial, sans-serif;
 font-size: var(--tamanho-fonte);
 color: var(--cor-secundaria);
 padding: var(--padding);
}

H1 {
 color: var(--cor-primaria);
}

BUTTON {
 background-color: var(--cor-primaria);
 color: white;
 border: none;
 padding: var(--padding);
 font-size: var(--tamanho-fonte);
 cursor: pointer;
 border-radius: 5px;
}

BUTTON:hover {
 background-color: var(--cor-secundaria);
}
</STYLE>
</HEAD>
<BODY>
<H1>EXEMPLO DE VARIÁVEIS CSS</H1>
<BUTTON>CLIQUE AQUI</BUTTON>
</BODY>
</HTML>
```

### TEMAS DINÂMICOS

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>TEMAS COM VARIÁVEIS CSS</TITLE>
<STYLE>
:ROOT {
--COR-PRIMARIA: #007bff;
--COR-SECUNDARIA: #6c757d;
--BACKGROUND: #f8f9fa;
}

BODY {
 background-color: var(--background);
 color: var(--cor-secundaria);
 font-family: Arial, sans-serif;
 padding: 20px;
}

BUTTON {
 background-color: var(--cor-primaria);
 color: white;
 padding: 10px 15px;
 border: none;
 border-radius: 5px;
 cursor: pointer;
}

BUTTON:hover {
 background-color: var(--cor-secundaria);
}

/* TEMA ESCURO */
.DARK-THEME {
--COR-PRIMARIA: #343a40;
--COR-SECUNDARIA: #ffffff;
--BACKGROUND: #212529;
}

</STYLE>
</HEAD>
<BODY>
<H1>TROCAR TEMA COM VARIÁVEIS CSS</H1>
<BUTTON onclick="document.body.classList.toggle('dark-theme')>ALTERAR TEMA</BUTTON>
</BODY>
</HTML>
```



## USANDO VARIÁVEIS EM CONSULTAS DE MÍDIA

### EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>VARIÁVEIS CSS EM MEDIA QUERIES</TITLE>
<STYLE>
:ROOT {
--TAMANHO-FONTE: 16PX;
--LARGURA-MAXIMA: 600PX;
}

BODY {
 FONT-SIZE: VAR(--TAMANHO-FONTE);
 MARGIN: 20PX;
 BACKGROUND-COLOR: LIGHTBLUE;
}

@MEDIA (MAX-WIDTH: VAR(--LARGURA-MAXIMA)) {
:ROOT {
 --TAMANHO-FONTE: 14PX;
 --LARGURA-MAXIMA: 480PX;
}

BODY {
 BACKGROUND-COLOR: LIGHTCORAL;
}
}
</STYLE>
</HEAD>
<BODY>
<H1>EXEMPLO DE VARIÁVEIS CSS EM MEDIA QUERIES</H1>
<P>REDUZA A LARGURA DA JANELA PARA VER AS MUDANÇAS.</P>
</BODY>
</HTML>
```

### ESPAÇAMENTOS COM VARIÁVEIS

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>ESPAÇAMENTOS DINÂMICOS</TITLE>
<STYLE>
:ROOT {
 --PADDING: 20PX;
}

DIV {
 BACKGROUND-COLOR: LIGHTGREEN;
 PADDING: VAR(--PADDING);
 BORDER: 1PX SOLID GREEN;
 TRANSITION: PADDING 0.3S EASE;
}

@MEDIA (MAX-WIDTH: 768PX) {
:ROOT {
 --PADDING: 10PX;
}
}
</STYLE>
</HEAD>
<BODY>
<DIV>
 ESTE BLOCO AJUSTA O ESPAÇAMENTO DINÂMICAMENTE COM BASE NA LARGURA DA TELA.
</DIV>
</BODY>
</HTML>
```

# A REGR A @PROPERTY

A REGR A @PROPERTY NO CSS É USADA PARA DEFINIR PROPRIEDADES PERSONALIZADAS (CUSTOM PROPERTIES) QUE TÊM COMPORTAMENTO ANIMÁVEL. ESSA FUNCIONALIDADE É ÚTIL QUANDO VOCÊ DESEJA QUE VARIÁVEIS CSS (—MINHA-VARIÁVEL) POSSAM SER ANIMADAS OU TRANSICIONADAS COM VALORES ESPECÍFICOS E SEGUROS.

## SINTAXE BÁSICA

1. **@PROPERTY —NOME-DA-PROPRIEDADE {**
2. **SYNTAX: "<TIPO-DE-DADO>";**
3. **INHERITS: <BOOLEANO>;**
4. **INITIAL-VALUE: <VALOR-INICIAL>;**
5. **}**

## PARÂMETROS

### • SYNTAX

DEFINE O TIPO DE VALOR QUE A PROPRIEDADE ACEITA, USANDO A MESMA SINTAXE DO CSS VALUES AND UNITS.

### EXEMPLOS:

- "<COLOR>": ACEITA VALORES DE COR (RED, #000, ETC.).
- "<LENGTH>": ACEITA COMPRIMENTOS (PX, %, ETC.).
- "<NUMBER>": ACEITA NÚMEROS (1, 2.5, ETC.).

### • INHERITS

DEFINE SE A PROPRIEDADE SERÁ HERDADA POR PADRÃO.

VALORES POSSÍVEIS:

- TRUE: SERÁ HERDADA.
- FALSE (PADRÃO): NÃO SERÁ HERDADA.

### • INITIAL-VALUE

- ESPECIFICA O VALOR INICIAL PADRÃO DA PROPRIEDADE.

## EXEMPLO PRÁTICO

DEFINIR E ANIMAR UMA PROPRIEDADE PERSONALIZADA  
AQUI, CRIAMOS UMA PROPRIEDADE PERSONALIZADA ANIMÁVEL PARA CONTROLAR O TAMANHO DE UM ELEMENTO:

```
<!DOCTYPE HTML>
<HTML LANG="PT-BR">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
<TITLE>EXEMPLO DE @PROPERTY</TITLE>
<STYLE>
@PROPERTY —TAMANHO-BLOCO {
 SYNTAX: "<LENGTH>";
 INHERITS: FALSE;
 INITIAL-VALUE: 100PX;
}

.BLOCO {
 WIDTH: VAR(--TAMANHO-BLOCO);
 HEIGHT: VAR(--TAMANHO-BLOCO);
 BACKGROUND-COLOR: LIGHTBLUE;
 TRANSITION: --TAMANHO-BLOCO 1S EASE;
}

.BLOCO:hover {
 --TAMANHO-BLOCO: 200PX;
}
</STYLE>
</HEAD>
<BODY>
<DIV CLASS="BLOCO"></DIV>
</BODY>
</HTML>
```



# DIMENSIONAMENTO DE CAIXA

O DIMENSIONAMENTO DE CAIXA (OU BOX SIZING) NO CSS É UMA PROPRIEDADE IMPORTANTE QUE DEFINE COMO AS DIMENSÕES DE UM ELEMENTO (COMO WIDTH E HEIGHT) SÃO CALCULADAS. COM ESSA PROPRIEDADE, VOCÊ PODE CONTROLAR SE A LARGURA E A ALTURA DE UM ELEMENTO DEVEM INCLUIR OU NÃO O PADDING E BORDER.

## PROPRIEDADE BOX-SIZING

A PROPRIEDADE BOX-SIZING TEM TRÊS VALORES PRINCIPAIS:

1. CONTENT-BOX (VALOR PADRÃO)
2. BORDER-BOX
3. PADDING-BOX (POUCO USADO)

## CONTENT-BOX (PADRÃO)

NO VALOR PADRÃO (CONTENT-BOX), A LARGURA E A ALTURA DE UM ELEMENTO NÃO INCLUEM O PADDING E O BORDER. OU SEJA, O TAMANHO ESPECIFICADO COM WIDTH E HEIGHT REFERE-SE APENAS AO CONTEÚDO, E O PADDING E BORDER SÃO ADICIONADOS AO TAMANHO TOTAL DA CAIXA.

## EXEMPLO DE CONTENT-BOX:

```
1. DIV {
2. WIDTH: 200px;
3. HEIGHT: 100px;
4. PADDING: 20px;
5. BORDER: 5px solid black;
6. BOX-SIZING: content-box; /* ESSE É O VALOR PADRÃO */
7. BACKGROUND-COLOR: lightblue;
8. }
```

- **LARGURA TOTAL:** 200px (CONTEÚDO) + 20px (PADDING ESQUERDO E DIREITO) + 5px (BORDER ESQUERDO E DIREITO) = 250px
- **ALTURA TOTAL:** 100px (CONTEÚDO) + 20px (PADDING SUPERIOR E INFERIOR) + 5px (BORDER SUPERIOR E INFERIOR) = 150px

## BORDER-BOX

QUANDO VOCÊ USA BOX-SIZING: BORDER-BOX, O PADDING E O BORDER SÃO INCLUÍDOS NAS DIMENSÕES DE LARGURA E ALTURA ESPECIFICADAS. OU SEJA, AS DIMENSÕES ESPECIFICADAS EM WIDTH E HEIGHT INCLuem TAMBÉM O ESPAÇO DO PADDING E DO BORDER.

## EXEMPLO DE BORDER-BOX:

```
1. DIV {
2. WIDTH: 200px;
3. HEIGHT: 100px;
4. PADDING: 20px;
5. BORDER: 5px solid black;
6. BOX-SIZING: border-box;
7. BACKGROUND-COLOR: lightgreen;
8. }
```

- **LARGURA TOTAL:** 200px (INCLUSO PADDING E BORDER)
- **ALTURA TOTAL:** 100px (INCLUSO PADDING E BORDER)

COM BORDER-BOX, A CAIXA DO ELEMENTO NÃO ULTRAPASSARÁ AS DIMENSÕES DEFINIDAS, MESMO COM PADDING OU BORDER.

## PADDING-BOX (MENOS USADO)

O VALOR PADDING-BOX DEFINE QUE O PADDING SERÁ INCLUÍDO NAS DIMENSÕES DE WIDTH E HEIGHT, MAS O BORDER NÃO SERÁ. OU SEJA, AS DIMENSÕES ESPECIFICADAS COM WIDTH E HEIGHT INCLuem O PADDING, MAS NÃO INCLuem A BORDA.

ESTE VALOR NÃO É TÃO COMUMENTE USADO, MAS PODE SER ÚTIL EM CASOS ESPECÍFICOS ONDE VOCÊ PRECISA AJUSTAR O COMPORTAMENTO DO DIMENSIONAMENTO.

## EXEMPLO DE PADDING-BOX:

```
1. DIV {
2. WIDTH: 200px;
3. HEIGHT: 100px;
4. PADDING: 20px;
5. BORDER: 5px solid black;
6. BOX-SIZING: padding-box;
7. BACKGROUND-COLOR: lightyellow;
8. }
```

- **LARGURA TOTAL:** 200px (INCLUSO PADDING)
- **ALTURA TOTAL:** 100px (INCLUSO PADDING)

# MEDIA QUERIES

CSS MEDIA QUERIES SÃO UMA TÉCNICA USADA NO DESIGN RESPONSIVO PARA APPLICAR ESTILOS CSS DIFERENTES COM BASE NAS CARACTERÍSTICAS DO DISPOSITIVO OU DA JANELA DE VISUALIZAÇÃO, COMO LARGURA, ALTURA, RESOLUÇÃO E ORIENTAÇÃO. ISSO PERMITE QUE VOCÊ CRIE LAYOUTS QUE SE ADAPTAM AUTOMATICAMENTE A DIFERENTES TAMANHOS DE TELA, COMO DESKTOPS, TABLETS E SMARTPHONES.

## SINTAXE BÁSICA:

```
1. @MEDIA <CONDICÃO> {
2. /* ESTILOS CSS */
3. }
```

AS CONDIÇÕES PODEM ENVOLVER CARACTERÍSTICAS DO DISPOSITIVO, COMO A LARGURA DA TELA, ALTURA, DENSIDADE DE PIXELS, ENTRE OUTRAS.

## EXEMPLO BÁSICO

NO VALOR PADRÃO (CONTENT-BOX), A LARGURA E A ALTURA DE UM ELEMENTO NÃO INCLUEM O PADDING E O BORDER. OU SEJA, O TAMANHO ESPECIFICADO COM WIDTH E HEIGHT REFERE-SE APENAS AO CONTEÚDO, E O PADDING E BORDER SÃO ADICIONADOS AO TAMANHO TOTAL DA CAIXA.

## EXEMPLO BÁSICO:

```
1. /* ESTILOS PADRÃO PARA TELAS GRANDES */
2. BODY {
3. FONT-SIZE: 18PX;
4. }
5.
6. /* ESTILOS PARA TELAS PEQUENAS (POR EXEMPLO, SMARTPHONES)
*/
7. @MEDIA (MAX-WIDTH: 600PX) {
8. BODY {
9. FONT-SIZE: 14PX;
10. }
11. }
```

## NESTE EXEMPLO:

- **PADRÃO:** O TAMANHO DA FONTE É 18PX.
- **MEDIA QUERY:** QUANDO A LARGURA DA TELA FOR MENOR OU IGUAL A 600PX, O TAMANHO DA FONTE SERÁ REDUZIDO PARA 14PX.

## PROPRIEDADES COMUNS EM MEDIA QUERIES

- **WIDTH E HEIGHT:** REFERE-SE À LARGURA E ALTURA DA JANELA DE VISUALIZAÇÃO (VIEWPORT).
- **MIN-WIDTH E MAX-WIDTH:** DEFINE UM INTERVALO DE LARGURA PARA A APLICAÇÃO DO ESTILO.
- **ORIENTATION:** IDENTIFICA A ORIENTAÇÃO DA TELA (PAISAGEM OU RETRATO).
- **RESOLUTION:** REFERE-SE À DENSIDADE DE PIXELS DA TELA (ÚTIL PARA TELAS DE ALTA RESOLUÇÃO, COMO RETINA).
- **ASPECT-RATIO:** RELACIONA A PROPORÇÃO ENTRE LARGURA E ALTURA DA TELA.

## EXEMPLOS DE MEDIA QUERIES COMUNS

### BASEADO NA LARGURA DA TELA

```
1. /* PARA TELAS COM LARGURA DE 600PX OU MENOS */
2. @MEDIA (MAX-WIDTH: 600PX) {
3. .CONTAINER {
4. FLEX-DIRECTION: COLUMN;
5. }
6. }
```

### MUDANÇA DE ORIENTAÇÃO

```
1. /* QUANDO O DISPOSITIVO ESTIVER EM MODO PAISAGEM */
2. @MEDIA (ORIENTATION: LANDSCAPE) {
3. BODY {
4. BACKGROUND-COLOR: LIGHTBLUE;
5. }
6. }
7.
8. /* QUANDO O DISPOSITIVO ESTIVER EM MODO RETRATO */
9. @MEDIA (ORIENTATION: PORTRAIT) {
10. BODY {
11. BACKGROUND-COLOR: LIGHTCORAL;
12. }
13. }
```

### RESOLUÇÃO DE TELA:

```
1. /* PARA TELAS COM RESOLUÇÃO DE 2X OU MAIS */
2. @MEDIA (MIN-RESOLUTION: 192DPI) {
3. .IMAGEM {
4. BACKGROUND-IMAGE: URL('IMAGEM-HIGHRES.PNG');
5. }
6. }
```



## COMBINANDO VÁRIOS CRITÉRIOS

```
1./* QUANDO A LARGURA FOR MAIOR QUE 600PX E A RESOLUÇÃO FOR
ALTA */
2.@MEDIA (MIN-WIDTH: 600PX) AND (MIN-RESOLUTION: 192DPI) {
3. BODY {
4. FONT-SIZE: 20PX;
5. }
6.}
```

## MEDIA QUERIES COM INTERVALOS

VOCÊ PODE COMBINAR VÁRIAS MEDIA QUERIES PARA TRATAR DIFERENTES INTERVALOS DE LARGURA, POR EXEMPLO:

```
1./* PARA TELAS PEQUENAS */
2.@MEDIA (MAX-WIDTH: 480PX) {
3. .CONTAINER {
4. PADDING: 10PX;
5. }
6.}
7.
8./* PARA TELAS MÉDIAS */
9.@MEDIA (MIN-WIDTH: 481PX) AND (MAX-WIDTH: 1024PX) {
10. .CONTAINER {
11. PADDING: 20PX;
12. }
13.}
14.
15./* PARA TELAS GRANDES */
16.@MEDIA (MIN-WIDTH: 1025PX) {
17. .CONTAINER {
18. PADDING: 30PX;
19. }
20.}
```

/\* PARA TABLETS E DISPOSITIVOS COM TELA MAIOR QUE 480PX \*/

```
@MEDIA (MIN-WIDTH: 481PX) AND (MAX-WIDTH: 1024PX) {
BODY {
 BACKGROUND-COLOR: LIGHTGREEN;
}
}
```

/\* PARA DESKTOPS COM TELAS MAIORES QUE 1024PX \*/

```
@MEDIA (MIN-WIDTH: 1025PX) {
BODY {
 BACKGROUND-COLOR: LIGHTBLUE;
}
}
```

## CONSULTAS DE MÍDIA PARA ALTA RESOLUÇÃO:

VOCÊ PODE USAR MEDIA QUERIES PARA AJUSTAR O DESIGN COM BASE NA DENSIDADE DE PIXELS DA TELA, ÚTIL PARA DISPOSITIVOS COM TELAS DE ALTA RESOLUÇÃO.

```
1.@MEDIA ONLY SCREEN AND (MIN-RESOLUTION: 2DPPX) {
2. IMG {
3. CONTENT: URL('IMAGEM-HIGH-RES.PNG');
4. }
5.}
```

## CONSULTAS DE MÍDIA PARA TIPO DE DISPOSITIVO (EX. DISPOSITIVOS TÁTEIS):

```
1.@MEDIA (POINTER: COARSE) {
2. BUTTON {
3. FONT-SIZE: 18PX;
4. }
5.}
```

## MEDIA QUERIES AVANÇADAS

### USANDO MIN-WIDTH E MAX-WIDTH:

AS MEDIA QUERIES PODEM SER COMBINADAS PARA APLICAR ESTILOS EM INTERVALOS ESPECÍFICOS, O QUE É ÚTIL PARA DESIGNS FLUIDOS E ADAPTÁVEIS.

```
/* PARA DISPOSITIVOS MÓVEIS */
@MEDIA (MAX-WIDTH: 480PX) {
BODY {
 BACKGROUND-COLOR: LIGHTGRAY;
}
}
```

# FLEXBOX

CSS FLEXBOX (OU FLEXIBLE BOX LAYOUT) É UM MODELO DE LAYOUT QUE FACILITA A CRIAÇÃO DE LAYOUTS COMPLEXOS E RESPONSIVOS, PERMITINDO DISTRIBUIR O ESPAÇO DISPONÍVEL ENTRE OS ITENS DE UM CONTÊINER DE FORMA EFICIENTE E CONTROLADA. FLEXBOX FOI PROJETADO PARA RESOLVER OS PROBLEMAS DE ALINHAMENTO E DISTRIBUIÇÃO DE ESPAÇO EM LAYOUTS, ESPECIALMENTE QUANDO AS DIMENSÕES DOS ITENS SÃO DESCONHECIDAS OU VARIÁVEIS.

## PROPRIEDADE DISPLAY: FLEX

A FLEXBOX COMEÇA COM O CONTÊINER. QUANDO VOCÊ APlica DISPLAY: FLEX AO ELEMENTO PAI, ELE SE Torna UM CONTÊINER FLEXÍVEL E SEUS FILHOS SE TORNAM ITENS FLEXÍVEIS.

### EXEMPLO BÁSICO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. }
4. }
```

## PROPRIEDADES DO CONTÊINER FLEX

### FLEX-DIRECTION

DEFINE A DIREÇÃO PRINCIPAL DOS ITENS DENTRO DO CONTÊINER. OS ITENS PODEM SER ORGANIZADOS EM UMA LINHA OU COLUNA.

- **ROW (PADRÃO):** OS ITENS SÃO ORGANIZADOS DA ESQUERDA PARA A DIREITA.
- **ROW-REVERSE:** OS ITENS SÃO ORGANIZADOS DA DIREITA PARA A ESQUERDA.
- **COLUMN:** OS ITENS SÃO ORGANIZADOS DE CIMA PARA BAIXO.
- **COLUMN-REVERSE:** OS ITENS SÃO ORGANIZADOS DE BAIXO PARA CIMA.

### EXEMPLO BÁSICO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. FLEX-DIRECTION: COLUMN;
4. }
5. }
```

### FLEX-WRAP

DETERMINA SE OS ITENS DEVEM QUEBRAR PARA A LINHA SEGUINTE QUANDO NÃO HOUVER ESPAÇO SUFICIENTE.

- **NOWRAP (PADRÃO):** OS ITENS FICAM NA MESMA LINHA.
- **WRAP:** OS ITENS QUEBRAM PARA A LINHA SEGUINTE QUANDO NECESSÁRIO.

- **WRAP-REVERSE:** OS ITENS QUEBRAM PARA A LINHA SEGUINTE, MAS DE FORMA INVERTIDA.

### EXEMPLO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. FLEX-WRAP: WRAP;
4. }
5. }
```

## JUSTIFY-CONTENT

CONTROLA O ALINHAMENTO DOS ITENS AO LONGO DO EIXO PRINCIPAL (HORIZONTAL POR PADRÃO, SE FLEX-DIRECTION FOR ROW).

- **FLEX-START:** ALINHA OS ITENS NO INÍCIO (PADRÃO).
- **FLEX-END:** ALINHA OS ITENS NO FINAL.
- **CENTER:** ALINHA OS ITENS NO CENTRO.
- **SPACE-BETWEEN:** DISTRIBUI OS ITENS COM O MÁXIMO DE ESPAÇO POSSÍVEL ENTRE ELES.
- **SPACE-AROUND:** DISTRIBUI OS ITENS COM ESPAÇO IGUAL AO REDOR DE CADA ITEM.
- **SPACE-EVENLY:** DISTRIBUI OS ITENS COM ESPAÇO IGUAL ENTRE TODOS OS ITENS E AS BORDAS DO CONTÊINER.

### EXEMPLO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. JUSTIFY-CONTENT: SPACE-BETWEEN;
4. }
5. }
```

## ALIGN-ITEMS

DEFINE O ALINHAMENTO DOS ITENS AO LONGO DO EIXO TRANSVERSAL (VERTICAL POR PADRÃO, SE FLEX-DIRECTION FOR ROW).

- **STRETCH (PADRÃO):** OS ITENS SÃO ESTICADOS PARA PREENCHER O CONTÊINER.
- **FLEX-START:** ALINHA OS ITENS NO INÍCIO DO EIXO TRANSVERSAL.
- **FLEX-END:** ALINHA OS ITENS NO FINAL DO EIXO TRANSVERSAL.
- **CENTER:** ALINHA OS ITENS NO CENTRO DO EIXO TRANSVERSAL.
- **BASELINE:** ALINHA OS ITENS PELA LINHA DE BASE DO TEXTO.

### EXEMPLO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. ALIGN-ITEMS: CENTER;
4. }
5. }
```



## ALIGN-CONTENT

ALINHA AS LINHAS DO CONTÊINER FLEXÍVEL QUANDO HÁ VÁRIAS LINHAS. FUNCIONA QUANDO FLEX-WRAP É ATIVADO.

- **FLEX-START:** ALINHA AS LINHAS NO INÍCIO.
- **FLEX-END:** ALINHA AS LINHAS NO FINAL.
- **CENTER:** ALINHA AS LINHAS NO CENTRO.
- **SPACE-BETWEEN:** DISTRIBUI AS LINHAS COM O MÁXIMO DE ESPAÇO ENTRE ELAS.
- **SPACE-AROUND:** DISTRIBUI AS LINHAS COM ESPAÇO IGUAL AO REDOR DE CADA LINHA.
- **STRETCH (PADRÃO):** AS LINHAS OCUPAM TODO O ESPAÇO DISPONÍVEL.

## EXEMPLO:

```
1. .CONTAINER {
2. DISPLAY: FLEX;
3. FLEX-WRAP: WRAP;
4. ALIGN-CONTENT: SPACE-AROUND;
5. }
```

## PROPRIEDADES DOS ITENS FLEX

### FLEX-GROW

DEFINE A CAPACIDADE DE UM ITEM CRESCER EM RELAÇÃO AOS OUTROS ITENS DENTRO DO CONTÊINER FLEXÍVEL. O VALOR PADRÃO É 0, O QUE SIGNIFICA QUE O ITEM NÃO CRESCERÁ.

- **FLEX-GROW: 1:** O ITEM PODE CRESCER E OCUPAR O ESPAÇO DISPONÍVEL.
- **FLEX-GROW: 2:** O ITEM CRESCERÁ EM UMA PROPORÇÃO MAIOR, EM RELAÇÃO AOS OUTROS ITENS.

## EXEMPLO:

```
1. .ITEM {
2. FLEX-GROW: 1;
3. }
```

### FLEX-SHRINK

DEFINE A CAPACIDADE DE UM ITEM ENCOLHER QUANDO O ESPAÇO DISPONÍVEL FOR INSUFICIENTE. O VALOR PADRÃO É 1, O QUE SIGNIFICA QUE O ITEM PODE ENCOLHER.

- **FLEX-SHRINK: 1:** O ITEM PODE ENCOLHER.
- **FLEX-SHRINK: 0:** O ITEM NÃO ENCOLHERÁ, MESMO QUE O ESPAÇO SEJA INSUFICIENTE.

## EXEMPLO:

```
1. .ITEM {
2. FLEX-SHRINK: 0;
3. }
```

## FLEX-BASIS

DEFINE O TAMANHO INICIAL DO ITEM ANTES DE APLICAR O CRESCIMENTO OU O ENCOLHIMENTO. PODE SER UM VALOR EM PIXELS, PORCENTAGEM OU AUTO (QUE É O TAMANHO DO CONTEÚDO).

## EXEMPLO:

```
1. .ITEM {
2. FLEX-BASIS: 100px;
3. }
```

### FLEX

A PROPRIEDADE FLEX É UMA ABREVIAÇÃO PARA FLEX-GROW, FLEX-SHRINK E FLEX-BASIS. É A FORMA MAIS FÁCIL DE CONTROLAR O COMPORTAMENTO DE FLEXIBILIDADE DE UM ITEM.

- **FLEX: 1:** SIGNIFICA QUE O ITEM PODE CRESCER, ENCOLHER E TER UM TAMANHO INICIAL DE 0.
- **FLEX: 0 1 AUTO:** NÃO CRESCER, PODE ENCOLHER, E O TAMANHO INICIAL É BASEADO NO CONTEÚDO.

## EXEMPLO:

```
1. .ITEM {
2. FLEX: 1;
3. }
```

### ALIGN-SELF

PERMITE QUE UM ITEM SE ALINHE INDIVIDUALMENTE AO LONGO DO EIXO TRANSVERSAL, INDEPENDENTEMENTE DE ALIGN-ITEMS NO CONTÊINER.

- **AUTO (PADRÃO):** SEGUE O ALINHAMENTO DEFINIDO PELO CONTÊINER.
- **FLEX-START:** ALINHA O ITEM NO INÍCIO DO EIXO TRANSVERSAL.
- **FLEX-END:** ALINHA O ITEM NO FINAL DO EIXO TRANSVERSAL.
- **CENTER:** ALINHA O ITEM NO CENTRO DO EIXO TRANSVERSAL.
- **BASELINE:** ALINHA O ITEM PELA LINHA DE BASE DO TEXTO.
- **STRETCH:** FAZ O ITEM OCUPAR TODO O EIXO TRANSVERSAL.

## EXEMPLO:

```
1. .ITEM {
2. ALIGN-SELF: CENTER;
3. }
```

## **EXEMPLO COMPLETO**

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
<META CHARSET="UTF-8">
<META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
<STYLE>
 .CONTAINER {
 DISPLAY: FLEX;
 JUSTIFY-CONTENT: SPACE-BETWEEN;
 ALIGN-ITEMS: CENTER;
 HEIGHT: 100vh;
 }
 .ITEM {
 FLEX: 1;
 BACKGROUND-COLOR: LIGHTBLUE;
 MARGIN: 10px;
 PADDING: 20px;
 TEXT-ALIGN: CENTER;
 }
 .ITEM::FIRST-CHILD {
 BACKGROUND-COLOR: LIGHTCORAL;
 }

```

```
.ITEM:last-child {
 background-color: lightgreen;
}
</STYLE>
</HEAD>
<BODY>

<DIV CLASS="CONTAINER">
 <DIV CLASS="ITEM">ITEM 1</DIV>
 <DIV CLASS="ITEM">ITEM 2</DIV>
 <DIV CLASS="ITEM">ITEM 3</DIV>
</DIV>

</BODY>
</HTML>
```

## **NESTE EXEMPLO:**

- O CONTÊINER .CONTAINER UTILIZA DISPLAY: FLEX PARA ORGANIZAR OS ITENS EM UMA LINHA HORIZONTAL.
- A PROPRIEDADE JUSTIFY-CONTENT: SPACE-BETWEEN DISTRIBUI O ESPAÇO IGUALMENTE ENTRE OS ITENS.
- A PROPRIEDADE ALIGN-ITEMS: CENTER ALINHA OS ITENS VERTICALMENTE AO CENTRO.

## **ANOTAÇÕES**















CAPÍTULO

# 04

## RESPONSIVE WEB DESIGN: CRIANDO EXPERIÊNCIAS ADAPTÁVEIS



### A CIÊNCIA DO RESPONSIVE WEB DESIGN

O RESPONSIVE WEB DESIGN (RWD) É UMA ABORDAGEM ESSENCIAL NO DESENVOLVIMENTO DE SITES MODERNOS, GARANTINDO QUE AS PÁGINAS WEB SE ADAPTEM AUTOMATICAMENTE A DIFERENTES TAMANHOS DE TELA E DISPOSITIVOS. DESDE SMARTPHONES ATÉ COMPUTADORES DE MESA, O RWD PERMITE QUE OS USUÁRIOS TENHAM UMA EXPERIÊNCIA CONSISTENTE E OTIMIZADA, INDEPENDENTEMENTE DO APARELHO QUE UTILIZAM.

ESSA TÉCNICA UTILIZA PRÁTICAS COMO GRIDS FLEXÍVEIS, IMAGENS RESPONSIVAS E MEDIA QUERIES PARA AJUSTAR O LAYOUT E OS ELEMENTOS VISUAIS, ASSEGURANDO ACESSIBILIDADE E USABILIDADE. COM O AUMENTO DO USO DE DISPOSITIVOS MÓVEIS, O DESIGN RESPONSIVO TORNOU-SE INDISPENSÁVEL PARA ALCANÇAR UM PÚBLICO MAIS AMPLO E MELHORAR O ENGAJAMENTO.

ALÉM DE MELHORAR A EXPERIÊNCIA DO USUÁRIO, O RESPONSIVE WEB DESIGN TAMBÉM IMPACTA POSITIVAMENTE O SEO, JÁ QUE MOTORES DE BUSCA COMO O GOOGLE PRIORIZAM SITES RESPONSIVOS EM SEUS RESULTADOS. ISSO FAZ COM QUE ESSA ABORDAGEM SEJA ESSENCIAL PARA EMPRESAS E PROFISSIONAIS QUE BUSCAM RELEVÂNCIA NO MUNDO DIGITAL.

POR MEIO DO RWD, DESENVOLVEDORES E DESIGNERS CRIAM INTERFACES DINÂMICAS QUE NÃO APENAS ATENDEM ÀS EXPECTATIVAS DOS USUÁRIOS MODERNOS, MAS TAMBÉM ESTABELECEM NOVOS PADRÕES DE EXCELÊNCIA EM DESIGN E FUNCIONALIDADE NA WEB.



# INTRODUÇÃO

## POR QUE O RESPONSIVE WEB DESIGN É IMPORTANTE?

**ACESSO DE MÚLTIPLOS DISPOSITIVOS:** PESSOAS ACESSTAM SITES EM DIFERENTES DISPOSITIVOS COM TAMANHOS DE TELA VARIADOS. UM DESIGN RESPONSIIVO GARANTE UMA EXPERIÊNCIA DE USUÁRIO CONSISTENTE.

**SEO (SEARCH ENGINE OPTIMIZATION):** O GOOGLE DÁ PRIORIDADE A SITES RESPONSIVOS NOS RESULTADOS DE BUSCA, ESPECIALMENTE EM DISPOSITIVOS MÓVEIS.

**MELHOR EXPERIÊNCIA DO USUÁRIO (UX):** OFERECE LAYOUTS OTIMIZADOS, MELHORANDO A NAVEGAÇÃO E LEITURA SEM NECESSIDADE DE ZOOM OU ROLAGEM EXCESSIVA.

**EFICIÊNCIA NO DESENVOLVIMENTO:** COM UM ÚNICO CÓDIGO BASE, VOCÊ COBRE DIFERENTES TAMANHOS DE TELA.

## COMO FUNCIONA O DESIGN RESPONSIIVO?

### GRATES FLEXÍVEIS

USE PORCENTAGENS OU UNIDADES RELATIVAS (EM VEZ DE PIXELS FIXOS) PARA DEFINIR LARGURAS.

### EXEMPLO:

```
1. CONTAINER {
2. WIDTH: 100%;
3. MAX-WIDTH: 1200PX;
4. MARGIN: AUTO;
5. }
```

### IMAGENS E MÍDIAS FLEXÍVEIS

AS IMAGENS E VÍDEOS SE AJUSTAM AUTOMATICAMENTE AO TAMANHO DO CONTAINER.

### EXEMPLO:

```
1. IMG {
2. MAX-WIDTH: 100%;
3. HEIGHT: AUTO;
4. }
```

### MEDIA QUERIES (CONSULTAS DE MÍDIA)

SÃO REGRAS CSS QUE PERMITEM CRIAR ESTILOS ESPECÍFICOS PARA DIFERENTES TAMANHOS DE TELA.

### EXEMPLO BÁSICO:

```
1. @MEDIA (MAX-WIDTH: 768PX) {
2. BODY {
3. FONT-SIZE: 16PX;
4. }
5. }
```

### EXEMPLO PRÁTICO

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
SCALE=1.0">
 <TITLE>EXEMPLO RESPONSIVO</TITLE>
<STYLE>
 BODY {
 FONT-FAMILY: ARIAL, SANS-SERIF;
 MARGIN: 0;
 PADDING: 0;
 }
 .HEADER {
 BACKGROUND: #333;
 COLOR: #FFF;
 PADDING: 10PX 20PX;
 TEXT-ALIGN: CENTER;
 }
 .CONTENT {
 DISPLAY: FLEX;
 FLEX-WRAP: WRAP;
 }
 .BOX {
 FLEX: 1 1 300PX; /* FLEXÍVEL, MÍNIMO DE 300PX */
 MARGIN: 10PX;
 PADDING: 20PX;
 BACKGROUND: #F4F4F4;
 BORDER: 1PX SOLID #CCC;
 }
```

```

@MEDIA (MAX-WIDTH: 600PX) {
 .BOX {
 FLEX: 1 1 100%; /* EM TELAS MENORES, AS CAIXAS OCUPAM
 100% DA LARGURA */
 }
}
</STYLE>
</HEAD>
<BODY>

```

```

<DIV CLASS="HEADER">
<H1>SITE RESPONSIVO</H1>
</DIV>
<DIV CLASS="CONTENT">
<DIV CLASS="BOX">CAIXA 1</DIV>
<DIV CLASS="BOX">CAIXA 2</DIV>
<DIV CLASS="BOX">CAIXA 3</DIV>
</DIV>
</BODY>
</HTML>

```

## VIEWPORT

### O VIEWPORT NO DESIGN RESPONSIVO

O VIEWPORT É A ÁREA VISÍVEL DE UMA PÁGINA DA WEB EM UM DISPOSITIVO. ELE REPRESENTA A REGIÃO DA TELA ONDE O CONTEÚDO É EXIBIDO E É ESSENCIAL PARA O DESIGN RESPONSIVO, POIS PERMITE QUE A PÁGINA SEJA AJUSTADA PARA DIFERENTES DISPOSITIVOS DE MANEIRA EFICIENTE.

### O QUE É A META TAG VIEWPORT?

No design responsivo, a meta tag viewport é usada para controlar como o conteúdo da página é exibido em diferentes tamanhos de tela. Essa tag é inserida no <head> do HTML e informa ao navegador como ele deve dimensionar e ajustar o conteúdo para diferentes dispositivos.

### EXEMPLO BÁSICO DE CÓDIGO DA META TAG VIEWPORT:

- <META NAME="viewport" CONTENT="width=device-width, initial-scale=1.0">

### ATRIBUTOS DA META TAG VIEWPORT

#### • WIDTH=DEVICE-WIDTH

- DEFINE A LARGURA DO VIEWPORT IGUAL À LARGURA DA TELA DO DISPOSITIVO.
- ISSO GARANTE QUE O LAYOUT DO SITE SEJA AJUSTADO AUTOMATICAMENTE AO TAMANHO DA TELA, SEJA EM UM DESKTOP, TABLET OU CELULAR.]

### EXEMPLO:

1. <META NAME="viewport" CONTENT="width=device-width">

#### • INITIAL-SCALE=1.0

- DEFINE O NÍVEL DE ZOOM INICIAL QUANDO A PÁGINA É CARREGADA.
- O VALOR 1.0 SIGNIFICA QUE A PÁGINA SERÁ EXIBIDA EM TAMANHO NORMAL, SEM ZOOM. PODE SER AJUSTADO PARA CONTROLAR O NÍVEL DE ZOOM INICIAL.

### EXEMPLO:

1. <META NAME="viewport" CONTENT="initial-scale=1.0">

#### • MAXIMUM-SCALE=N

- CONTROLA O ZOOM MÁXIMO PERMITIDO NA PÁGINA.
- POR EXEMPLO, MAXIMUM-SCALE=1 IMPIDE QUE OS USUÁRIOS FAÇAM ZOOM NA PÁGINA.

### EXEMPLO:

1. <META NAME="viewport" CONTENT="maximum-scale=1.0">

#### • USER-SCALABLE=N

- IMPIDE QUE OS USUÁRIOS POSSAM FAZER ZOOM NA PÁGINA, OU SEJA, DESATIVA A FUNCIONALIDADE DE ZOOM.

### EXEMPLO:

1. <META NAME="viewport" CONTENT="user-scalable=no">



### EXEMPLO COMPLETO DA META TAG VIEWPORT

AQUI ESTÁ UM EXEMPLO QUE COMBINA ESSES ATRIBUTOS PARA UM CONTROLE TOTAL DO VIEWPORT:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
```

#### NESTE EXEMPLO:

- **WIDTH=DEVICE-WIDTH:** O LAYOUT SERÁ AJUSTADO PARA O TAMANHO DA TELA DO DISPOSITIVO.
- **INITIAL-SCALE=1.0:** O ZOOM INICIAL SERÁ NORMAL, SEM AMPLIAÇÃO.
- **MAXIMUM-SCALE=1.0:** O USUÁRIO NÃO PODERÁ DAR ZOOM.
- **USER-SCALABLE=NO:** O ZOOM ESTÁ DESATIVADO.

### POR QUE ESSA TAG É IMPORTANTE?

**ADAPTAÇÃO AUTOMÁTICA:** ELA ASSEGURA QUE O CONTEÚDO DA PÁGINA SE AJUSTE CORRETAMENTE AO DISPOSITIVO QUE ESTÁ SENDO USADO PARA VISUALIZAÇÃO.

**CONTROLE DE ZOOM:** PERMITE DEFINIR SE O ZOOM SERÁ PERMITIDO OU NÃO, O QUE É ÚTIL PARA GARANTIR QUE O LAYOUT FIQUE SEMPRE CONSISTENTE E LEGÍVEL.

**MELHORIA DA EXPERIÊNCIA DO USUÁRIO:** SEM A META TAG, OS SITES PODEM APARECER EM UM FORMATO REDUZIDO OU MUITO AMPLIADO EM DISPOSITIVOS MÓVEIS, O QUE PREJUDICA A NAVEGAÇÃO.

### EXEMPLO PRÁTICO

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>EXEMPLO DE VIEWPORT</title>
 <style>
 body {
 font-family: Arial, sans-serif;
 margin: 0;
 padding: 0;
 }
 .content {
 width: 100%;
 max-width: 1200px;
 margin: auto;
 padding: 20px;
 }
 </style>
</head>
<body>
 <div class="content">
 <h1>BEM-VINDO AO DESIGN RESPONSIVO</h1>
 <p>ESTE SITE ESTÁ OTIMIZADO PARA TODOS OS DISPOSITIVOS!</p>
 </div>
</body>
</html>
```

## GRID

### O VIEWPORT NO DESIGN RESPONSIVO

O CSS GRID É UMA TÉCNICA DE LAYOUT QUE PERMITE DIVIDIR O CONTEÚDO DE UMA PÁGINA EM LINHAS E COLUNAS. ELE FORNECE CONTROLE TOTAL SOBRE A DISTRIBUIÇÃO E ALINHAMENTO DOS ELEMENTOS EM UMA PÁGINA, O QUE É ESSENCIAL PARA CRIAR DESIGNS RESPONSIVOS.

O GRID LAYOUT FUNCIONA MELHOR QUANDO VOCÊ TEM UM LAYOUT COM MÚLTIPLOS ELEMENTOS QUE PRECISAM SER ORGANIZADOS EM UMA ESTRUTURA DE LINHAS E COLUNAS. AO USAR O GRID, VOCÊ PODE AJUSTAR O NÚMERO DE COLUNAS, A LARGURA DAS COLUNAS, O ESPAÇAMENTO E O ALINHAMENTO DOS ITENS DE MANEIRA EFICIENTE.

### COMO FUNCIONA O GRID LAYOUT

#### DEFINIR UM CONTÊINER DE GRID

PARA USAR O GRID LAYOUT, VOCÊ PRECISA DEFINIR UM CONTÊINER DE GRID NO CSS, APlicando a propriedade `display: grid` ao elemento pai. Esse contêiner vai gerar o sistema de linhas e colunas.

#### DEFINIR COLUNAS E LINHAS

DENTRO DO CONTÊINER DE GRID, VOCÊ PODE DEFINIR O NÚMERO DE COLUNAS E O TAMANHO DELAS COM AS PROPRIEDADES `grid-template-columns` (PARA AS COLUNAS) E `grid-template-rows` (PARA AS LINHAS).

## POSICIONAR OS ITENS NO GRID

OS ITENS DENTRO DO CONTÊINER DE GRID PODEM SER POSICIONADOS AUTOMATICAMENTE OU MANUALMENTE EM CÉLULAS ESPECÍFICAS DA GRADE.

## PROPRIEDADES DO GRID LAYOUT

### DISPLAY: GRID

TRANSFORMA O ELEMENTO PAI EM UM CONTÊINER DE GRID.

### EXEMPLO:

```
1. .GRID-CONTAINER {
2. DISPLAY: GRID;
3. }
```

### GRID-TEMPLATE-COLUMNS E GRID-TEMPLATE-ROWS

DEFINE O NÚMERO DE COLUNAS E LINHAS NO GRID, ASSIM COMO O TAMANHO DELAS.

VOCÊ PODE USAR VALORES COMO PX, FR (FRAÇÕES DO ESPAÇO DISPONÍVEL), OU AUTO.

### EXEMPLO:

```
1. .GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR); /* 3 COLUNAS COM O
 MESMO TAMANHO */
4. GRID-TEMPLATE-ROWS: AUTO; /* O NÚMERO DE LINHAS SERÁ
 DETERMINADO PELO CONTEÚDO */
5. }
```

### GRID-GAP (OU GAP)

DEFINE O ESPAÇO ENTRE AS LINHAS E COLUNAS DO GRID.

### EXEMPLO:

```
1. .GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR);
4. GAP: 20PX;
5. }
```

## EXEMPLO BÁSICO DE LAYOUT COM GRID

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
 SCALE=1.0">
 <TITLE>EXEMPLO DE GRID</TITLE>
<STYLE>
 .GRID-CONTAINER {
 DISPLAY: GRID;
 GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR); /* 3 COLUNAS DE
 TAMANHO IGUAL */
 GAP: 20PX; /* ESPAÇAMENTO ENTRE AS COLUNAS E LINHAS */
 MARGIN: 20PX;
 }
 .GRID-ITEM {
 BACKGROUND-COLOR: #F4F4F4;
 PADDING: 20PX;
 BORDER: 1PX SOLID #CCC;
 }
</STYLE>
</HEAD>
<BODY>
 <DIV CLASS="GRID-CONTAINER">
 <DIV CLASS="GRID-ITEM">ITEM 1</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 2</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 3</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 4</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 5</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 6</DIV>
 </DIV>
</BODY>
</HTML>
```

### NESTE EXEMPLO:

- O CONTÊINER .GRID-CONTAINER É DIVIDIDO EM 3 COLUNAS DE TAMANHO IGUAL USANDO GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR).
- O GAP: 20PX CRIA UM ESPAÇAMENTO DE 20PX ENTRE OS ITENS DA GRADE.
- O LAYOUT É ADAPTÁVEL, PODENDO SE AJUSTAR CONFORME O TAMANHO DA TELA DO DISPOSITIVO.



## EXEMPLO BÁSICO DE LAYOUT COM GRID

PARA TORNAR O LAYOUT AINDA MAIS RESPONSIVO, PODEMOS USAR MEDIA QUERIES PARA AJUSTAR O NÚMERO DE COLUNAS DEPENDENDO DO TAMANHO DA TELA.

### EXEMPLO RESPONSIVO DE GRID

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-SCALE=1.0">
 <TITLE>EXEMPLO DE GRID RESPONSIVO</TITLE>
 <STYLE>

 .GRID-CONTAINER {
 DISPLAY: GRID;
 GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR); /* 3 COLUNAS PARA
TELAS GRANDES */
 GAP: 20PX;
 }

 .GRID-ITEM {
 BACKGROUND-COLOR: #F4F4F4;
 PADDING: 20PX;
 BORDER: 1PX SOLID #CCC;
 }

 /* AJUSTE PARA TELAS PEQUENAS */
 @MEDIA (MAX-WIDTH: 768PX) {
 .GRID-CONTAINER {
 GRID-TEMPLATE-COLUMNS: REPEAT(2, 1FR); /* 2 COLUNAS
PARA TELAS MENORES */
 }
 }

 /* AJUSTE PARA TELAS MUITO PEQUENAS */
 @MEDIA (MAX-WIDTH: 480PX) {
 .GRID-CONTAINER {
 GRID-TEMPLATE-COLUMNS: 1FR; /* 1 COLUNA PARA TELAS
MUITO PEQUENAS */
 }
 }
 </STYLE>
</HEAD>
<BODY>
 <DIV CLASS="GRID-CONTAINER">
 <DIV CLASS="GRID-ITEM">ITEM 1</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 2</DIV>
 <DIV CLASS="GRID-ITEM">ITEM 3</DIV>
 </DIV>
</BODY>
```

```
<DIV CLASS="GRID-ITEM">ITEM 4</DIV>
<DIV CLASS="GRID-ITEM">ITEM 5</DIV>
<DIV CLASS="GRID-ITEM">ITEM 6</DIV>
</DIV>
</BODY>
</HTML>
```

### NESTE EXEMPLO:

- PARA TELAS GRANDES, TEMOS 3 COLUNAS.
- PARA TELAS MÉDIAS (COM LARGURA MÁXIMA DE 768PX), O LAYOUT SE AJUSTA PARA 2 COLUNAS.
- PARA TELAS PEQUENAS (COM LARGURA MÁXIMA DE 480PX), O LAYOUT MUDA PARA 1 COLUNA.

## GRID CONTAINER NO CSS GRID LAYOUT

O GRID CONTAINER É O ELEMENTO QUE DEFINE UM CONTÊINER DE GRID. TODOS OS ITENS DENTRO DESSE CONTÊINER SÃO AUTOMATICAMENTE ORGANIZADOS EM UMA ESTRUTURA DE GRID. QUANDO VOCÊ USA O CSS GRID, O PRIMEIRO PASSO É DEFINIR UM ELEMENTO COMO GRID CONTAINER COM A PROPRIEDADE DISPLAY: GRID. DENTRO DESSE CONTÊINER, OS ITENS SÃO DISPOSTOS EM UMA GRADE QUE PODE SER CONFIGURADA EM TERMOS DE LINHAS E COLUNAS.

## PROPRIEDADES DO GRID CONTAINER

### DISPLAY: GRID

A PROPRIEDADE DISPLAY: GRID É O PONTO DE PARTIDA PARA CRIAR UM CONTÊINER DE GRID. ELA ATIVA O SISTEMA DE GRID NO ELEMENTO, PERMITINDO QUE SEUS FILHOS SEJAM ORGANIZADOS EM UMA ESTRUTURA DE GRID.

### EXEMPLO:

1. .GRID-CONTAINER {
2. DISPLAY: GRID;
3. }

### GRID-TEMPLATE-COLUMNS

DEFINE O NÚMERO DE COLUNAS DO GRID E SEU TAMANHO. VOCÊ PODE ESPECIFICAR TAMANHOS FIXOS, RELATIVOS OU AUTOMÁTICOS PARA AS COLUNAS.

### EXEMPLO: CRIAR 3 COLUNAS DE TAMANHO FIXO.

1. .GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: 100PX 100PX 100PX; /\* 3 COLUNAS COM 100PX DE LARGURA \*/
4. }

**EXEMPLO: CRIAR 3 COLUNAS COM TAMANHOS PROPORCIONAIS (FRAÇÕES DE ESPAÇO DISPONÍVEL).**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: 1FR 2FR 1FR; /* A SEGUNDA COLUNA
 OCUPA O DOBRO DE ESPAÇO */
4. }
```

#### GRID-TEMPLATE-ROWS

DEFINE O NÚMERO DE LINHAS E SEU TAMANHO. SIMILAR A GRID-TEMPLATE-COLUMNS, VOCÊ PODE CONFIGURAR AS LINHAS COM TAMANHOS FIXOS, RELATIVOS OU AUTOMÁTICOS.

**EXEMPLO: CRIAR 2 LINHAS COM ALTURA FIXA.**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-ROWS: 200px 100px; /* 2 LINHAS, A PRIMEIRA
 COM 200px E A SEGUNDA COM 100px */
4. }
```

#### GRID-TEMPLATE-AREAS

PERMITE NOMEAR AS ÁREAS DO GRID. ISSO É ÚTIL PARA LAYOUTS MAIS COMPLEXOS, ONDE VOCÊ PODE ESPECIFICAR VISUALMENTE A POSIÇÃO DOS ITENS NO GRID.

**EXEMPLO: USANDO GRID-TEMPLATE-AREAS.**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: 1fr 2fr;
4. GRID-TEMPLATE-ROWS: 100px 200px;
5. GRID-TEMPLATE-AREAS:
6. "HEADER HEADER"
7. "CONTENT SIDEBAR";
8. }
```

#### GAP OU GRID-GAP

A PROPRIEDADE GAP DEFINE O ESPAÇAMENTO ENTRE AS CÉLULAS DO GRID (TANTO ENTRE LINHAS QUANTO COLUNAS). ELA É UM ATALHO PARA AS PROPRIEDADES GRID-ROW-GAP E GRID-COLUMN-GAP.

**EXEMPLO:**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. GRID-TEMPLATE-COLUMNS: 1fr 1fr 1fr;
4. GAP: 10px; /* ESPAÇAMENTO DE 10px ENTRE TODAS AS CÉLULAS
 DO GRID */
5. }
```

#### JUSTIFY-ITEMS

A PROPRIEDADE JUSTIFY-ITEMS CONTROLA O ALINHAMENTO HORIZONTAL DOS ITENS DENTRO DO GRID. PODE SER USADA PARA DEFINIR COMO OS ITENS SÃO ALINHADOS NAS CÉLULAS DO GRID.

**VALORES POSSÍVEIS: START, END, CENTER, STRETCH (PADRÃO).**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. JUSTIFY-ITEMS: center; /* ALINHA TODOS OS ITENS AO CENTRO
 HORIZONTALMENTE */
4. }
```

#### ALIGN-ITEMS

A PROPRIEDADE ALIGN-ITEMS CONTROLA O ALINHAMENTO VERTICAL DOS ITENS DENTRO DO GRID.

**VALORES POSSÍVEIS: START, END, CENTER, STRETCH (PADRÃO).**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. ALIGN-ITEMS: center; /* ALINHA TODOS OS ITENS AO CENTRO
 VERTICALMENTE */
4. }
```

#### JUSTIFY-CONTENT

A PROPRIEDADE JUSTIFY-CONTENT DEFINE O ALINHAMENTO DO CONTEÚDO DO GRID NO EIXO HORIZONTAL (LINHA). ELA AFETA O ESPAÇO ENTRE AS CÉLULAS DO GRID QUANDO O CONTEÚDO NÃO PREENCHE TODO O ESPAÇO DISPONÍVEL.

**VALORES POSSÍVEIS: START, END, CENTER, STRETCH, SPACE-BETWEEN, SPACE-AROUND.**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. JUSTIFY-CONTENT: center; /* ALINHA O CONTEÚDO DO GRID AO
 CENTRO HORIZONTALMENTE */
4. }
```

#### ALIGN-CONTENT

A PROPRIEDADE ALIGN-CONTENT DEFINE O ALINHAMENTO DO CONTEÚDO DO GRID NO EIXO VERTICAL (COLUNA). SEMELHANTE AO JUSTIFY-CONTENT, MAS NO EIXO VERTICAL.

**VALORES POSSÍVEIS: START, END, CENTER, STRETCH, SPACE-BETWEEN, SPACE-AROUND.**

```
1. GRID-CONTAINER {
2. DISPLAY: GRID;
3. ALIGN-CONTENT: center; /* ALINHA O CONTEÚDO DO GRID AO
 CENTRO VERTICALMENTE */
4. }
```



## GRID ITEM NO CSS GRID LAYOUT

OS GRID ITEMS SÃO OS ELEMENTOS FILHOS DIRETOS DE UM GRID CONTAINER. QUANDO VOCÊ DEFINE UM CONTÊINER COMO DISPLAY: GRID, TODOS OS ELEMENTOS FILHOS DENTRO DELE SE TORNAM AUTOMATICAMENTE GRID ITEMS E SÃO POSICIONADOS CONFORME AS REGRAS DEFINIDAS NO CONTÊINER.

## PROPRIEDADES DOS GRID ITEMS

### GRID-COLUMN

A PROPRIEDADE GRID-COLUMN PERMITE QUE VOCÊ CONTROLE ONDE O GRID ITEM COMEÇARÁ E TERMINARÁ AO LONGO DAS COLUNAS. VOCÊ PODE USAR VALORES NUMÉRICOS OU PALAVRAS-CHAVE PARA DEFINIR O POSICIONAMENTO.

- **SINTAXE:**

- **GRID-COLUMN: START / END;**

**EXEMPLO: UM ITEM COMEÇA NA COLUNA 1 E VAI ATÉ A COLUNA 3:**

```
1. .GRID-ITEM {
2. GRID-COLUMN: 1 / 3;
3. }
```

### GRID-ROW

A PROPRIEDADE GRID-ROW CONTROLA O POSICIONAMENTO DO GRID ITEM AO LONGO DAS LINHAS DO GRID. ASSIM COMO GRID-COLUMN, VOCÊ PODE ESPECIFICAR ONDE O ITEM COMEÇA E TERMINA NAS LINHAS.

- **SINTAXE:**

- **GRID-ROW: START / END;**

**EXEMPLO: UM ITEM COMEÇA NA LINHA 1 E VAI ATÉ A LINHA 3:**

```
1. .GRID-ITEM {
2. GRID-ROW: 1 / 3;
3. }
```

### GRID-COLUMN-START E GRID-COLUMN-END

ESSAS DUAS PROPRIEDADES OFERECEM CONTROLE MAIS DETALHADO SOBRE O POSICIONAMENTO DE UM GRID ITEM AO LONGO DAS COLUNAS, ESPECIFICANDO EXPLICITAMENTE ONDE ELE DEVE COMEÇAR E TERMINAR.

**EXEMPLO: UM ITEM COMEÇA NA COLUNA 1 E TERMINA NA COLUNA 3:**

```
1. .GRID-ITEM {
2. GRID-COLUMN-START: 1;
3. GRID-COLUMN-END: 3;
4. }
```

## GRID-ROW-START E GRID-ROW-END

SEMELHANTE AO USO DAS PROPRIEDADES GRID-COLUMN-START E GRID-COLUMN-END, ESSAS PROPRIEDADES CONTROLAM O POSICIONAMENTO AO LONGO DAS LINHAS DO GRID.

**EXEMPLO: UM ITEM COMEÇA NA LINHA 1 E TERMINA NA LINHA 3:**

```
1. .GRID-ITEM {
2. GRID-ROW-START: 1;
3. GRID-ROW-END: 3;
4. }
```

### GRID-AREA

A PROPRIEDADE GRID-AREA É UM ATALHO PARA GRID-ROW-START, GRID-ROW-END, GRID-COLUMN-START E GRID-COLUMN-END. ELA PERMITE QUE VOCÊ DEFINA O POSICIONAMENTO DO GRID ITEM DE UMA FORMA MAIS COMPACTA.

- **SINTAXE:**

- **GRID-AREA: ROW-START / COLUMN-START / ROW-END / COLUMN-END;**

**EXEMPLO: DEFINIR UMA ÁREA DO GRID QUE COMEÇA NA LINHA 1, COLUNA 1 E TERMINA NA LINHA 3, COLUNA 3:**

```
1. .GRID-ITEM {
2. GRID-AREA: 1 / 1 / 3 / 3;
3. }
```

### JUSTIFY-SELF

A PROPRIEDADE JUSTIFY-SELF ALINHA UM GRID ITEM HORIZONTALMENTE DENTRO DA CÉLULA DO GRID. É USADA PARA DEFINIR COMO O ITEM É POSICIONADO AO LONGO DO EIXO HORIZONTAL (DA ESQUERDA PARA A DIREITA).

**VALORES POSSÍVEIS:** START, END, CENTER, STRETCH (PADRÃO).

```
1. .GRID-ITEM {
2. JUSTIFY-SELF: CENTER; /* ALINHA HORIZONTALMENTE NO
3. CENTRO */
4. }
```

### ALIGN-SELF

A PROPRIEDADE ALIGN-SELF ALINHA UM GRID ITEM VERTICALMENTE DENTRO DE SUA CÉLULA NO GRID. É USADA PARA CONTROLAR O ALINHAMENTO AO LONGO DO EIXO VERTICAL (DE CIMA PARA BAIXO).

**VALORES POSSÍVEIS:** START, END, CENTER, STRETCH (PADRÃO).

```
1. .GRID-ITEM {
2. ALIGN-SELF: START; /* ALINHA VERTICALMENTE AO TOPO */
3. }
```

## ORDER

A PROPRIEDADE ORDER PERMITE ALTERAR A ORDEM DE EXIBIÇÃO DOS GRID ITEMS NO GRID. POR PADRÃO, TODOS OS ITENS TÊM VALOR DE ORDER: 0, MAS VOCÊ PODE MODIFICAR ESSE VALOR PARA REORDENAR OS ITENS.

**EXEMPLO: MUDAR A ORDEM DE EXIBIÇÃO DE UM ITEM.**

```
1. .GRID-ITEM {
2. ORDER: 1; /* ALTERA A ORDEM DE EXIBIÇÃO DO ITEM */
3. }
```



# MEDIA QUERIES

## O VIEWPORT NO DESIGN RESPONSIVO

AS MEDIA QUERIES SÃO UMA FUNCIONALIDADE DO CSS QUE PERMITEM APPLICAR ESTILOS DIFERENTES DEPENDENDO DAS CARACTERÍSTICAS DO DISPOSITIVO OU DO AMBIENTE DE VISUALIZAÇÃO. COM ELAS, É POSSÍVEL AJUSTAR O LAYOUT DE UM SITE DE MANEIRA DINÂMICA, CRIANDO UMA EXPERIÊNCIA DE USUÁRIO ADAPTADA A DIFERENTES TAMANHOS DE TELA, RESOLUÇÕES E ORIENTAÇÕES DE DISPOSITIVOS.

EM OUTRAS PALAVRAS, MEDIA QUERIES AJUDAM A APPLICAR REGRAS CSS ESPECÍFICAS DE ACORDO COM AS CONDIÇÕES DO DISPOSITIVO (COMO LARGURA DA TELA, RESOLUÇÃO, ORIENTAÇÃO, ETC).

## SINTaxe BÁSICA DE MEDIA QUERIES

1. **@MEDIA:** A REGRA INICIAL QUE INDICA QUE ESTAMOS USANDO UMA MEDIA QUERY.
2. **CONDICÃO:** A EXPRESSÃO QUE DEFINE A CARACTERÍSTICA QUE ESTAMOS TESTANDO, COMO A LARGURA DA TELA OU A RESOLUÇÃO.
3. **BLOCO DE ESTILO:** O CONJUNTO DE REGRAS CSS QUE SERÁ APLICADO QUANDO A CONDIÇÃO FOR ATENDIDA.

**EXEMPLO BÁSICO DE UMA MEDIA QUERY:**

```
1. @MEDIA (MAX-WIDTH: 768PX) {
2. /* ESTILOS PARA TELAS COM LARGURA MÁXIMA DE 768PX */
3. BODY {
4. BACKGROUND-COLOR: LIGHTBLUE;
5. }
6. }
```

## PRINCIPAIS CARACTERÍSTICAS DAS MEDIA QUERIES

- **WIDTH / HEIGHT:** A LARGURA E ALTURA DA JANELA DE VISUALIZAÇÃO (VIEWPORT).
- **MIN-WIDTH / MAX-WIDTH:** USADAS PARA ESPECIFICAR LIMITES DE LARGURA, PODENDO SER APLICADAS EM CONDIÇÕES MÍNIMAS OU MÁXIMAS.
- **MIN-HEIGHT / MAX-HEIGHT:** DEFINEM O LIMITE DE ALTURA MÍNIMA OU MÁXIMA DA TELA.
- **ORIENTATION:** DETERMINA A ORIENTAÇÃO DA TELA, SEJA ELA PORTRAIT (RETRATO) OU LANDSCAPE (PAISAGEM).
- **RESOLUTION:** REFERE-SE À RESOLUÇÃO DA TELA, ÚTIL PARA TELAS DE ALTA DEFINIÇÃO (COMO RETINA DISPLAY).
- **ASPECT-RATIO:** DEFINE A PROPORÇÃO DA TELA (LARGURA/ALTURA).



## EXEMPLO DE MEDIA QUERIES

### MUDANDO O LAYOUT COM BASE NA LARGURA DA TELA

ESTE EXEMPLO MUDA O NÚMERO DE COLUNAS DO LAYOUT CONFORME A LARGURA DA TELA DO DISPOSITIVO.

```
1./* LAYOUT PADRÃO - 3 COLUNAS */
2.GRID-CONTAINER {
3. DISPLAY: GRID;
4. GRID-TEMPLATE-COLUMNS: REPEAT(3, 1FR);
5. GAP: 20PX;
6.
7.
8./* PARA TELAS MENORES QUE 768PX, MUDA PARA 2 COLUNAS */
9.@MEDIA (MAX-WIDTH: 768PX) {
10. .GRID-CONTAINER {
11. GRID-TEMPLATE-COLUMNS: REPEAT(2, 1FR);
12. }
13. }
14.
15./* PARA TELAS MENORES QUE 480PX, MUDA PARA 1 COLUNA */
16.@MEDIA (MAX-WIDTH: 480PX) {
17. .GRID-CONTAINER {
18. GRID-TEMPLATE-COLUMNS: 1FR;
19. }
20. }
```

### NESTE EXEMPLO:

- TELAS GRANDES: O LAYOUT TEM 3 COLUNAS.
- TELAS MÉDIAS (ATÉ 768PX): O LAYOUT MUDA PARA 2 COLUNAS.
- TELAS PEQUENAS (ATÉ 480PX): O LAYOUT FICA COM 1 COLUNA.

### MUDANDO A ORIENTAÇÃO DO LAYOUT COM ORIENTATION

VOCÊ TAMBÉM PODE ADAPTAR O DESIGN COM BASE NA ORIENTAÇÃO DA TELA.

```
1./* PARA TELAS EM MODO RETRATO (VERTICAL), USE UMA COR DE FUNDO DIFERENTE */
2.@MEDIA (ORIENTATION: PORTRAIT) {
3. BODY {
4. BACKGROUND-COLOR: LIGHTYELLOW;
5. }
6.
7.
8./* PARA TELAS EM MODO PAISAGEM (HORIZONTAL), USE OUTRA COR DE FUNDO */
9.@MEDIA (ORIENTATION: LANDSCAPE) {
10. BODY {
11. BACKGROUND-COLOR: LIGHTGREEN;
12. }
13. }
```

## USANDO MIN-WIDTH E MAX-WIDTH

VOCÊ PODE USAR MIN-WIDTH E MAX-WIDTH PARA APLICAR REGRAS BASEADAS NA LARGURA DA TELA, AJUDANDO A CONTROLAR O LAYOUT EM DIFERENTES TAMANHOS DE DISPOSITIVOS.

```
1./* PARA TELAS GRANDES (MÍNIMO DE 1024PX), O TEXTO FICA MAIOR */
2.@MEDIA (MIN-WIDTH: 1024PX) {
3. BODY {
4. FONT-SIZE: 18PX;
5. }
6.
7.
8./* PARA TELAS MENORES QUE 1024PX, O TEXTO FICA MENOR */
9.@MEDIA (MAX-WIDTH: 1023PX) {
10. BODY {
11. FONT-SIZE: 14PX;
12. }
13. }
```



# IMAGENS

## COMO TORNAR IMAGENS RESPONSIVAS?

A PRINCIPAL ESTRATÉGIA PARA TORNAR AS IMAGENS RESPONSIVAS É GARANTIR QUE ELAS SE AJUSTEM AUTOMATICAMENTE AO TAMANHO DA TELA, SEM ULTRAPASSAR OS LIMITES DO CONTÊINER ONDE ESTÃO INSERIDAS.

### PROPRIEDADE MAX-WIDTH: 100%

UMA DAS MANEIRAS MAIS COMUNS DE TORNAR UMA IMAGEM RESPONSIVA É USAR A PROPRIEDADE MAX-WIDTH NO CSS. ISSO PERMITE QUE A IMAGEM OCupe NO MÁXIMO 100% DA LARGURA DO SEU CONTÊINER, MANTENDO A PROPORÇÃO ORIGINAL DA IMAGEM.

```
IMG {
 max-width: 100%;
 height: auto; /* MANTÉM A PROPORÇÃO DA IMAGEM */
}
```

### NO EXEMPLO ACIMA:

- MAX-WIDTH: 100% FAZ COM QUE A IMAGEM NUNCA ULTRAPASSE A LARGURA DO SEU CONTÊINER, AJUSTANDO-SE AO SEU TAMANHO.
- HEIGHT: AUTO GARANTE QUE A ALTURA DA IMAGEM SEJA AJUSTADA PROPORCIONALMENTE À LARGURA, EVITANDO QUE ELA SEJA DISTorcIDA.

### EXEMPLO DE USO COM HTML

```
1. <!DOCTYPE HTML>
2. <HTML lang="en">
3. <HEAD>
4. <meta charset="UTF-8">
5. <meta name="viewport" content="width=device-width,
6. initial-scale=1.0">
7. <title>EXEMPLO DE IMAGEM RESPONSIVA</title>
8. <style>
9. img {
10. max-width: 100%;
11. height: auto; }
12. </style>
13. </HEAD>
14. <BODY>
15. <div class="container">
16.
17. </div>
18. </BODY>
19. </HTML>
```

## ATRIBUTO SRCSET PARA IMAGENS RESPONSIVAS

OUTRA ABORDAGEM PARA IMAGENS RESPONSIVAS É O USO DO ATRIBUTO SRCSET. ESSE ATRIBUTO PERMITE FORNECER DIFERENTES VERSÕES DA MESMA IMAGEM PARA DIFERENTES TAMANHOS DE TELA E RESOLUÇÕES, OTIMIZANDO O DESEMPENHO AO CARREGAR A IMAGEM MAIS ADEQUADA AO DISPOSITIVO.

### COMO FUNCIONA O SRCSET?

O ATRIBUTO SRCSET PERMITE DEFINIR VÁRIAS FONTES DE IMAGEM COM DIFERENTES LARGURAS OU RESOLUÇÕES, E O NAVEGADOR ESCOLHERÁ AUTOMATICAMENTE A MELHOR IMAGEM COM BASE NA RESOLUÇÃO DA TELA E NA LARGURA DA JANELA DE VISUALIZAÇÃO.

1. 

### NESTE EXEMPLO:

- SRCSET FORNECE TRÊS VERSÕES DA IMAGEM:
  - IMAGEM-GRANDE.JPG 1024W: IMAGEM DE MAIOR RESOLUÇÃO (PARA TELAS GRANDES).
  - IMAGEM-MEDIA.JPG 768W: IMAGEM DE RESOLUÇÃO MÉDIA (PARA TABLETS).
  - IMAGEM-PEQUENA.JPG 500W: IMAGEM DE RESOLUÇÃO BAIXA (PARA DISPOSITIVOS MÓVEIS).

## PICTURE PARA IMAGENS RESPONSIVAS

ALÉM DE USAR O SRCSET, O ELEMENTO <PICTURE> PERMITE ESPECIFICAR DIFERENTES IMAGENS E FORMATS DE IMAGEM PARA DIFERENTES CONDIÇÕES, COMO RESOLUÇÃO E FORMATO DE IMAGEM.

### EXEMPLO:

```
1. <picture>
2. <source srcset="imagem-2x.jpg" media="(min-width:
3. 768px)">
3. <source srcset="imagem-1x.jpg" media="(max-width:
4. 767px)">
4.
5. </picture>
```



#### NO EXEMPLO ACIMA:

- O ELEMENTO <PICTURE> OFERECE UMA FORMA DE FORNECER IMAGENS DIFERENTES DEPENDENDO DA LARGURA DA TELA.
- MEDIA="MIN-WIDTH: 768PX": A IMAGEM IMAGEM-2X.JPG SERÁ CARREGADA PARA TELAS COM LARGURA MÍNIMA DE 768PX.
- MEDIA="MAX-WIDTH: 767PX": A IMAGEM IMAGEM-1X.JPG SERÁ CARREGADA PARA TELAS COM LARGURA DE ATÉ 767PX.
- <IMG>: A TAG <IMG> SERVE COMO UMA IMAGEM DE FALLBACK, CASO O NAVEGADOR NÃO SUPORTE O ELEMENTO

#### IMAGENS DE ALTA RESOLUÇÃO E RETINA DISPLAYS

QUANDO O SEU SITE FOR VISUALIZADO EM DISPOSITIVOS COM TELAS DE ALTA RESOLUÇÃO (COMO O RETINA DISPLAY DOS DISPOSITIVOS APPLE), VOCÊ PODE OFERECER IMAGENS COM MAIOR DENSIDADE DE PIXELS PARA GARANTIR QUE A QUALIDADE DAS IMAGENS SEJA MANTIDA.

#### EXEMPLO PARA RETINA DISPLAY

USANDO O SRCSET COM UMA VERSÃO DE MAIOR RESOLUÇÃO DA IMAGEM:

1. **<IMG SRC="IMAGEM.JPG"**
  2. **SRCSET="IMAGEM-2X.JPG 2X, IMAGEM-3X.JPG 3X"**
  3. **ALT="IMAGEM COM ALTA RESOLUÇÃO"**
- IMAGEM-2X.JPG 2X: PARA TELAS COM DENSIDADE DE PIXELS DE 2X (COMO A RETINA DISPLAY).
  - IMAGEM-3X.JPG 3X: PARA TELAS COM DENSIDADE DE PIXELS DE 3X (COMO DISPOSITIVOS MAIS RECENTES COM TELAS DE ALTA DEFINIÇÃO).

## VIDEOS

#### CÓMO TORNAR VÍDEOS RESPONSIVOS

ASSIM COMO AS IMAGENS, OS VÍDEOS DEVEM SER AJUSTADOS PARA SE ADAPTAREM AO TAMANHO DA TELA DE DIFERENTES DISPOSITIVOS. PARA GARANTIR QUE OS VÍDEOS SEJAM RESPONSIVOS, A ABORDAGEM MAIS COMUM É USAR A TÉCNICA DE CAIXA DE CONTÊINER COM PROPORÇÃO DE ASPECTO FIXA.

UMA DAS MANEIRAS MAIS SIMPLES DE TORNAR UM VÍDEO RESPONSIVO É ENVOLVENDO O ELEMENTO <IFRAME> (OU O <VIDEO> EM HTML5) COM UMA DIV DE CONTÊINER E DEFININDO A ALTURA E LARGURA DO VÍDEO COM BASE NA PROPORÇÃO DO CONTÊINER.

#### TÉCNICA DE CAIXAS DE CONTÊINER RESPONSIVAS

A TÉCNICA MAIS COMUM PARA TORNAR VÍDEOS RESPONSIVOS ENVOLVE USAR O CSS PARA MANTER A PROPORÇÃO DE ASPECTO DO VÍDEO, ENQUANTO A LARGURA SE ADAPTA AO TAMANHO DA TELA. ESSA ABORDAGEM FUNCIONA BEM PARA VÍDEOS INCORPORADOS VIA <IFRAME>, COMO OS DO YOUTUBE.

#### CSS PARA VÍDEOS RESPONSIVOS

AQUI ESTÁ A ABORDAGEM USANDO UMA CAIXA DE CONTÊINER PARA MANTER A PROPORÇÃO DO VÍDEO:

#### 1./\* CONTÊINER DE VÍDEO RESPONSIVO \*/

```
2..VIDEO-CONTAINER {
3. POSITION: RELATIVE;
4. PADDING-BOTTOM: 56.25%; /* 16:9 ASPECT RATIO
 (ALTURA/LARGURA) */
5. HEIGHT: 0;
6. OVERFLOW: HIDDEN;
7. MAX-WIDTH: 100%;
8. }
9.
10./* O IFRAME OCUPA TODA A LARGURA E ALTURA DO CONTÊINER */
11..VIDEO-CONTAINER IFRAME {
12. POSITION: ABSOLUTE;
13. TOP: 0;
14. LEFT: 0;
15. WIDTH: 100%;
16. HEIGHT: 100%;
17. }
```

## HTML COM O CONTÊINER RESPONSIVO

```
1. <!DOCTYPE HTML>
2. <HTML LANG="en">
3. <HEAD>
4. <META CHARSET="UTF-8">
5. <META NAME="viewport" CONTENT="width=device-width,
 initial-scale=1.0">
6. <TITLE>VÍDEO RESPONSIVO</TITLE>
7. <STYLE>
8. /* INCLUA O CÓDIGO CSS ACIMA AQUI */
9. </STYLE>
10. </HEAD>
11. <BODY>
12. <DIV CLASS="VIDEO-CONTAINER">
13. <IFRAME
14. SRC="HTTPS://WWW.YOUTUBE.COM/EMBED/DQW4W9WGHCQ"
15. FRAMEBORDER="0"
16. ALLOW="ACCELEROMETER; AUToplay; ENCRYPTED-
 MEDIA; GYROSCOPE; PICTURE-IN-PICTURE"
17. ALLOWFULLSCREEN></IFRAME>
18. </DIV>
19. </BODY>
19. </HTML>
```

### EXPLICAÇÃO:

- PADDING-BOTTOM: 56.25%: ISSO DEFINE A PROPORÇÃO DE ASPECTO DO VÍDEO. PARA UM VÍDEO DE 16:9, A ALTURA É 56,25% DA LARGURA. ISSO GARANTE QUE O VÍDEO SE AJUSTE CORRETAMENTE EM QUALQUER TAMANHO DE TELA, MANTENDO A PROPORÇÃO.
- HEIGHT: 0; OVERFLOW: HIDDEN; ESSAS PROPRIEDADES FAZEM COM QUE A ALTURA SEJA CALCULADA COM BASE NA LARGURA E QUE O CONTEÚDO QUE ULTRAPASSAR O CONTÊINER SEJA ESCONDIDO.
- WIDTH: 100%; HEIGHT: 100%; NO IFRAME GARANTE QUE O VÍDEO OCUPE TODA A ÁREA DISPONÍVEL NO CONTÊINER.

### VÍDEOS COM O ELEMENTO <VIDEO> NO HTML5

PARA VÍDEOS QUE NÃO ESTÃO EMBUTIDOS DE OUTRAS FONTES (COMO YOUTUBE OU VIMEO), VOCÊ PODE USAR A TAG <VIDEO> DO HTML5. PARA TORNÁ-LA RESPONSIVA, VOCÊ PODE USAR A MESMA ABORDAGEM DE CONTÊINER COM PROPORÇÃO DE ASPECTO FIXA.

### CSS PARA VÍDEO RESPONSIVO COM A TAG <VIDEO>

```
1. VIDEO-CONTAINER {
2. POSITION: RELATIVE;
3. PADDING-BOTTOM: 56.25%; /* MANTÉM A PROPORÇÃO 16:9 */
4. HEIGHT: 0;
5. OVERFLOW: HIDDEN;
6. MAX-WIDTH: 100%;
7. }
8.
9. VIDEO-CONTAINER VIDEO {
10. POSITION: ABSOLUTE;
11. TOP: 0;
12. LEFT: 0;
13. WIDTH: 100%;
14. HEIGHT: 100%;
15. }
```

### HTML COM A TAG <VIDEO>

```
<!DOCTYPE HTML>
<HTML LANG="en">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="viewport" CONTENT="width=device-width, initial-
scale=1.0">
 <TITLE>VÍDEO RESPONSIVO</TITLE>
 <STYLE>
 /* INCLUA O CÓDIGO CSS ACIMA AQUI */
 </STYLE>
</HEAD>
<BODY>
 <DIV CLASS="VIDEO-CONTAINER">
 <VIDEO CONTROLS>
 <SOURCE SRC="VIDEO-EXEMPLO.MP4" TYPE="VIDEO/MP4">
 SEU NAVEGADOR NÃO SUPORTA A TAG DE VÍDEO.
 </VIDEO>
 </DIV>
</BODY>
</HTML>
```

### NESTE EXEMPLO:

- O VÍDEO SERÁ REDIMENSIONADO AUTOMATICAMENTE PARA OCUPAR A LARGURA DO CONTÊINER E MANTER A PROPORÇÃO DE 16:9.



## OBJECT-FIT PARA CONTROLAR O AJUSTE DO VÍDEO

O OBJECT-FIT É UMA PROPRIEDADE DO CSS QUE DEFINE COMO O CONTEÚDO DE UM ELEMENTO (COMO UMA IMAGEM OU VÍDEO) DEVE SE AJUSTAR AO SEU CONTÊINER. ELE PODE SER ÚTIL PARA GARANTIR QUE O VÍDEO PREENCHA O CONTÊINER SEM DISTORCER A IMAGEM OU CORTAR PARTES IMPORTANTES.

### EXEMPLO COM OBJECT-FIT

```
1. video {
2. width: 100%;
3. height: 100%;
4. object-fit: cover; /* FAZ COM QUE O VÍDEO PREENCHA O
5. CONTÊINER SEM DISTORÇÃO */
}
```

A PROPRIEDADE OBJECT-FIT: COVER GARANTE QUE O VÍDEO OCUPE TODO O ESPAÇO DO CONTÊINER SEM PERDER A PROPORÇÃO. O VÍDEO SERÁ CORTADO, SE NECESSÁRIO, PARA COBRIR A ÁREA DO CONTÊINER.

### OPÇÕES DE OBJECT-FIT:

- COVER: O VÍDEO PREENCHE O CONTÊINER, MAS PODE SER CORTADO PARA MANTER A PROPORÇÃO.
- CONTAIN: O VÍDEO É REDIMENSIONADO PARA CABER DENTRO DO CONTÊINER SEM SER CORTADO.
- FILL: O VÍDEO PREENCHE TODO O CONTÊINER, MAS PODE SER DISTORCIDO.

# FRAMEWORKS

## DESIGN RESPONSIVO COM FRAMEWORKS

FRAMEWORKS DE FRONT-END SÃO FERRAMENTAS PODEROSAS QUE AJUDAM OS DESENVOLVEDORES A CRIAR LAYOUTS E DESIGNS RESPONSIVOS DE MANEIRA MAIS EFICIENTE. ELES OFERECEM ESTRUTURAS PRÉ-DEFINIDAS, COMPONENTES E FUNCIONALIDADES QUE ACELERAM O DESENVOLVIMENTO, PERMITINDO QUE VOCÊ SE CONCENTRE EM CRIAR EXPERIÊNCIAS DE USUÁRIO INCRÍVEIS SEM PRECISAR REINVENTAR A RODA.

EXISTEM VÁRIOS FRAMEWORKS POPULARES PARA DESIGN RESPONSIVO, COMO O BOOTSTRAP, O FOUNDATION, E O TAILWIND CSS. VAMOS EXPLORAR OS PRINCIPAIS ASPECTOS DE CADA UM DESESSESS FRAMEWORKS.

## BOOTSTRAP

O BOOTSTRAP É UM DOS FRAMEWORKS MAIS POPULARES PARA DESENVOLVIMENTO DE LAYOUTS RESPONSIVOS. ELE FOI CRIADO PELO TIME DO TWITTER E POSSUI UMA VASTA COLEÇÃO DE FERRAMENTAS E COMPONENTES PARA FACILITAR A CRIAÇÃO DE SITES RESPONSIVOS E MODERNOS.

### CARACTERÍSTICAS DO BOOTSTRAP:

- **SISTEMA DE GRID RESPONSIVO:** O BOOTSTRAP UTILIZA UM SISTEMA DE GRID (GRADE) BASEADO EM 12 COLUNAS, O QUE FACILITA O CONTROLE DO LAYOUT EM DIFERENTES TAMANHOS DE TELA.

• **COMPONENTES PRÉ-ESTILIZADOS:** INCLUI UMA GRANDE VARIEDADE DE COMPONENTES, COMO BOTÕES, TABELAS, FORMULÁRIOS, BARRAS DE NAVEGAÇÃO, E MUITO MAIS, QUE PODEM SER FACILMENTE INTEGRADOS AO SEU PROJETO.

- **UTILIZAÇÃO DE MEDIA QUERIES:** O BOOTSTRAP FAZ USO DE MEDIA QUERIES INTEGRADAS PARA ADAPTAR O LAYOUT A DIFERENTES TAMANHOS DE TELA.
- **FACILIDADE DE USO:** FÁCIL DE COMEÇAR, MESMO PARA QUEM NÃO TEM MUITA EXPERIÊNCIA COM CSS OU DESIGN RESPONSIVO.

### EXEMPLO BÁSICO DE BOOTSTRAP

```
<!DOCTYPE HTML>
<HTML LANG="EN">
<HEAD>
 <META CHARSET="UTF-8">
 <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH, INITIAL-
 SCALE=1.0">
 <TITLE>EXEMPLO BOOTSTRAP</TITLE>
 <LINK HREF="HTTPS://CDN.JSDELIVR.NET/NPM/BOOTSTRAP@5.3.0-
 ALPHA1/DIST/CSS/BOOTSTRAP.MIN.CSS" REL="stylesheet">
</HEAD>
```

```

<BODY>
 <DIV CLASS="CONTAINER">
 <DIV CLASS="ROW">
 <DIV CLASS="COL-12 COL-MD-6">
 <H1>EXEMPLO DE LAYOUT RESPONSIVO</H1>
 <P>ESSE CONTEÚDO OCUPA 12 COLUNAS EM TELAS PEQUENAS E
6 COLUNAS EM TELAS MÉDIAS OU MAIORES.</P>
 </DIV>
 </DIV>
 </DIV>

 <SCRIPT SRC="HTTPS://CDN.JSDELIVR.NET/NPM/BOOTSTRAP@5.3.0-
ALPHA1/DIST/JS/BOOTSTRAP.BUNDLE.MIN.JS"></SCRIPT>
</BODY>
</HTML>

```

#### EXPLICAÇÃO:

- CONTAINER: CRIA UM CONTÊINER COM PADDING AUTOMÁTICO PARA MANTER O CONTEÚDO CENTRALIZADO.
- ROW: DEFINE UMA LINHA ONDE OS ELEMENTOS (COLUNAS) SERÃO ALINHADOS.
- COL-12 COL-MD-6: A CLASSE COL-12 FAZ COM QUE A COLUNA OCupe TODA A LARGURA DA TELA EM DISPOSITIVOS PEQUENOS. A CLASSE COL-MD-6 FAZ COM QUE A COLUNA OCupe METADE DA LARGURA EM TELAS MÉDIAS OU MAIORES.

#### FOUNDATION

O FOUNDATION É OUTRO FRAMEWORK DE FRONT-END MUITO POPULAR, CRIADO PELA ZURB. ELE TAMBÉM OFERECE UMA SÉRIE DE FERRAMENTAS PARA DESIGN RESPONSIVO E COMPONENTES PRONTOS PARA USO.

#### CARACTERÍSTICAS DO FOUNDATION:

- **SISTEMA DE GRID FLEXÍVEL:** O FOUNDATION OFERECE UM SISTEMA DE GRID MUITO FLEXÍVEL QUE PERMITE QUE VOCÊ CRIE LAYOUTS COMPLEXOS E RESPONSIVOS COM FACILIDADE.
- **DESIGN MÓVEL PRIMEIRO:** O FOUNDATION SEGUE O PRINCÍPIO DE "MOBILE-FIRST", O QUE SIGNIFICA QUE ELE É OTIMIZADO PARA DISPOSITIVOS MÓVEIS E ESCALA PARA DESKTOPS.
- **COMPONENTES E PLUGINS:** INCLUI DIVERSOS COMPONENTES E PLUGINS COMO BOTÕES, FORMULÁRIOS, BARRAS DE NAVEGAÇÃO E SLIDESHOWS.

#### EXEMPLO BÁSICO DE FOUNDATION

```

1. <!DOCTYPE HTML>
2. <HTML LANG="En">
3. <HEAD>
4. <META CHARSET="UTF-8">
5. <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH,
INITIAL-SCALE=1.0">
6. <TITLE>EXEMPLO FOUNDATION</TITLE>
7. <LINK REL="stylesheet"
HREF="HTTPS://CDN.JSDELIVR.NET/NPM/FOUNDATION-
SITES@6.6.3/DIST/CSS/FOUNDATION.MIN.CSS">
8. </HEAD>
9. <BODY>
10. <DIV CLASS="GRID-CONTAINER">
11. <DIV CLASS="GRID-X GRID-PADDING-X">
12. <DIV CLASS="CELL SMALL-12 MEDIUM-6">
13. <H1>EXEMPLO DE LAYOUT RESPONSIVO COM
FOUNDATION</H1>
14. <P>ESSE CONTEÚDO OCUPA 12 COLUNAS EM
DISPOSITIVOS PEQUENOS E 6 COLUNAS EM DISPOSITIVOS MÉDIOS
OU MAIORES.</P>
15. </DIV>
16. </DIV>
17. </DIV>
18.
19. <SCRIPT SRC="HTTPS://CDN.JSDELIVR.NET/NPM/FOUNDATION-
SITES@6.6.3/DIST/JS/FOUNDATION.MIN.JS"></SCRIPT>
20. </BODY>
21. </HTML>

```

#### EXPLICAÇÃO:

- GRID-CONTAINER: CONTÊINER QUE CONTÉM A GRID.
- GRID-X GRID-PADDING-X: DEFINE UMA LINHA FLEXÍVEL, COM O ESPAÇAMENTO ENTRE AS CÉLULAS.
- CELL SMALL-12 MEDIUM-6: A CLASSE SMALL-12 FAZ COM QUE A CÉLULA OCupe TODA A LARGURA EM TELAS PEQUENAS, ENQUANTO MEDIUM-6 FAZ COM QUE OCupe METADE DA LARGURA EM TELAS MÉDIAS OU MAIORES.



## TAILWIND CSS

O TAILWIND CSS É UM FRAMEWORK UTILITÁRIO DE CSS QUE PERMITE CONSTRUIR DESIGNS DIRETAMENTE NO HTML. AO CONTRÁRIO DO BOOTSTRAP E DO FOUNDATION, QUE VÊM COM COMPONENTES PRONTOS, O TAILWIND OFERECE UMA ABORDAGEM MAIS MODULAR, PERMITINDO CRIAR LAYOUTS PERSONALIZADOS USANDO CLASSES UTILITÁRIAS.

### CARACTERÍSTICAS DO TAILWIND CSS:

- ABORDAGEM DE CLASSES UTILITÁRIAS:** TAILWIND É BASEADO EM CLASSES UTILITÁRIAS, O QUE SIGNIFICA QUE VOCÊ APLICA ESTILOS DIRETAMENTE NO HTML, COMO BG-BLUE-500, TEXT-WHITE, FLEX, ENTRE OUTROS.
- DESIGN PERSONALIZÁVEL:** AO CONTRÁRIO DOS FRAMEWORKS TRADICIONAIS, O TAILWIND PERMITE UMA PERSONALIZAÇÃO MAIS GRANULAR SEM SOBRESCREVER CLASSES PREDEFINIDAS.
- MOBILE-FIRST:** ASSIM COMO O FOUNDATION, O TAILWIND TAMBÉM ADOTA O CONCEITO DE "MOBILE-FIRST", COM BREAKPOINTS PARA ADAPTAR O LAYOUT A DIFERENTES DISPOSITIVOS.

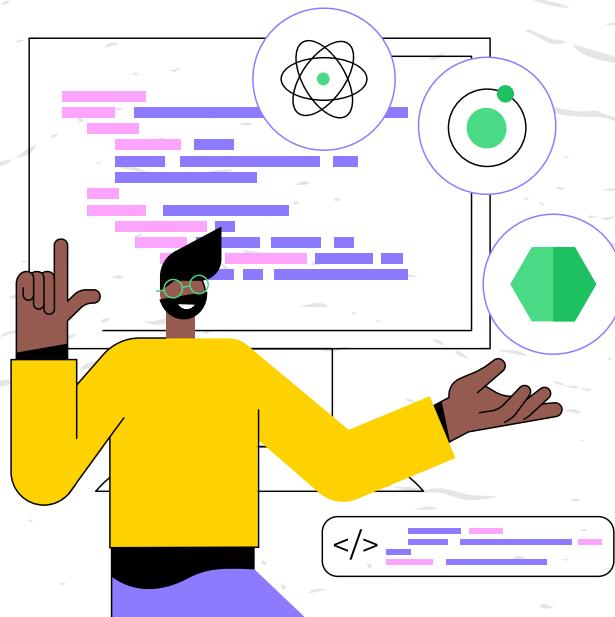
### EXEMPLO BÁSICO DE TAILWIND CSS

```

1. <!DOCTYPE HTML>
2. <HTML LANG="en">
3. <HEAD>
4. <META CHARSET="UTF-8">
5. <META NAME="VIEWPORT" CONTENT="WIDTH=DEVICE-WIDTH,
 INITIAL-SCALE=1.0">
6. <TITLE>EXEMPLO TAILWIND CSS</TITLE>
7. <SCRIPT SRC="HTTPS://CDN.TAILWINDCSS.COM"></SCRIPT>
8. </HEAD>
9. <BODY>
10. <DIV CLASS="CONTAINER MX-AUTO PH-4">
11. <DIV CLASS="GRID GRID-COLS-1 MD:GRID-COLS-2 GAP-4">
12. <DIV>
13. <H1 CLASS="TEXT-XL FONT-BOLD">EXEMPLO DE LAYOUT
 RESPONSIVO COM TAILWIND</H1>
14. <P>AQUI, O CONTEÚDO OCUPA TODA A LARGURA EM
 DISPOSITIVOS PEQUENOS E METADE DA LARGURA EM DISPOSITIVOS
 MÉDIOS OU MAIORES.</P>
15. </DIV>
16. </DIV>
17. </DIV>
18. </BODY>
19. </HTML>
```

## VANTAGENS DE USAR FRAMEWORKS PARA DESIGN RESPONSIVO

- DESENVOLVIMENTO RÁPIDO:** FRAMEWORKS OFERECEM COMPONENTES E FUNCIONALIDADES PRONTOS, O QUE ACELERA O PROCESSO DE DESENVOLVIMENTO.
- CONSISTÊNCIA:** GARANTEM QUE O DESIGN SEJA CONSISTENTE EM DIFERENTES DISPOSITIVOS E NAVEGADORES, JÁ QUE AS CLASSES E COMPONENTES SÃO TESTADOS E PADRONIZADOS.
- MOBILE-FIRST:** A MAIORIA DOS FRAMEWORKS É PROJETADA COM UMA ABORDAGEM MOBILE-FIRST, O QUE É CRUCIAL PARA GARANTIR UMA BOA EXPERIÊNCIA DE USUÁRIO EM DISPOSITIVOS MÓVEIS.
- FÁCIL PERSONALIZAÇÃO:** A PERSONALIZAÇÃO DE LAYOUTS E ESTILOS É FACILITADA, ESPECIALMENTE NO TAILWIND CSS, ONDE VOCÊ PODE APLICAR CLASSES UTILITÁRIAS DIRETAMENTE NO HTML.



## **ANOTAÇÕES**









