CSS - Cascading Style Sheets

As declarações de estilos no CSS devem sempre conter a propriedade que será alterada, seguida por dois pontos para separar do valor que será atribuído. Por fim, coloca-se um ponto-e-vírgula para indicar que a declaração terminou. Exemplo: `font-weight: bold;`

Alguns Grupos de Propriedades

Formatação de Fontes

A formatação de fontes no CSS utiliza um conjunto de propriedades para alterar a forma de apresentação dos desenhos das fontes:

- **font-style**: define o estilo da fonte. Valores: **normal** (sem efeito), **italic** (itálico) e **oblique** (inclinado). Valor padrão: **normal**.
- **font-variant**: indica a forma de escrita de uma fonte. Valores: **normal** (sem efeito) e **small-caps** (caixa-alta ou versalete ou "Todas em Maiúsculas"). Valor padrão: **normal**.
- **font-weight**: especifica a espessura da fonte. Valores: **normal** (sem efeito), **bold** (grosso), **bolder** (mais grosso), **lighter** (mais fino) e as espessuras da mais fina para a mais grossa com base nos valores: **100**, **200**, **300**, **400**, **500**, **600**, **700**, **800** e **900**. Valor padrão: **normal**.
- **font-size**: determina o tamanho da fonte. Valores: **xx-small** (super extra pequeno), **x-small** (extra pequeno), **small** (pequeno), **medium** (médio), **large** (grande), **x-large** (extra grande), **xx-large** (super extra grande), **smaller** (pequeno relativo), **larger** (grande relativo) ou informar uma medida de tamanho em: **%** (porcentagem em relação ao elemento-pai), **in** (polegadas), **cm** (centímetros), **mm** (milímetros), **pt** (pontos), **px** (pixels), **ch** (relativo à largura do caractere "zero" da fonte em uso), **em** (relativo ao tamanho da fonte em uso pelo elemento), **rem** (relativo ao tamanho da fonte em uso pelo elemento-raiz ou navegador), **vw** (relativo a 1% da largura da janela do navegador), **vh** (relativo a 1% da altura da janela do navegador), **vmin** (relativo a 1% da menor dimensão da janela do navegador), **vmax** (relativo a 1% maior dimensão da janela do navegador), **ch** (relativo à largura do caractere "zero" da fonte em uso), **ex** (relativo à metade da altura da fonte em uso – altura do caractere "x" minúsculo), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **medium**.
- **line-height**: indica o espaçamento entre linhas de um texto. Valores: **normal** (sem efeito), um número que será multiplicado pelo tamanho da fonte atual, uma unidade de medida para altura (px, pt, cm, etc.), **%** (porcentagem em relação ao tamanho da fonte em uso), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **normal**.
- **font-family**: Nomes de família de fontes (como **serif**, **sans-serif**, **cursive**, **fantasy** e **monospace**) ou nome da fonte específica como **Courier**, **Courier New**, **Arial**, **Times Nem Roman**, **Georgia**, **Lucila Console**, **Tahoma**, **Impact**, etc.
- **font**: indica numa única declaração todas as propriedades relativas ao tipo de letra, podendo usar as opções das propriedades **font-style**, **font-variant**, **font-weight**, **font-size** / **line-height** e **font-family**.

Formatação de Textos

A formatação de textos no CSS é conseguida com um conjunto de propriedades para permitir ajustes na forma de apresentação dos textos:

- **direction**: permite definir a direção de escrita de um texto. Valores: **ltr** (left-to-right – da esquerda para a direita) e **rtl** (right-to-left – da direita para a esquerda). Valor padrão: **ltr**. O efeito visual dessa propriedade é semelhante ao da propriedade **text-align** com os valores **left** e **right**.
- **text-align**: indica o formato de alinhamento do texto. Como opções de uso podem ser os valores **left** (esquerda), **right** (direita), **center** (centralizado) e **justify** (justificado). Valor padrão: **left**.
- **text-decoration**: determina um efeito decorativo em um texto. Valores: **none** (nenhum), **underline** (sublinhado), **overline** (sobrelinhado), **line-through** (sobreposto/tachado) e **blink** (piscante). Valor padrão: **none**.
- **text-indent**: define o nível de deslocamento positivo ou negativo de um texto para a direita ou para a esquerda. Os valores podem ser atribuídos em: uma unidade de medida (px, pt,

cm, em, etc.), **%** (porcentagem em relação à largura do elemento-pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **0**.

- **text-transform**: especifica o formato de apresentação das letras de um texto. Valores: **none** (nenhum), **capitalize** (todas as primeiras letras de uma frase serão iniciadas com caracteres maiúsculos), **uppercase** (maiúsculo) e **lowercase** (minúsculos). Valor padrão: **none**.
- **letter-spacing**: indica a distância do espaçamento entre os caracteres de um texto. Valores: **normal** (sem efeito), o comprimento positivo ou negativo da distância fornecido em uma unidade de medida (px, pt, cm, em, etc.), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **normal**.
- **word-spacing**: define a distância do espaçamento entre as palavras de um texto. Valores: **normal** (sem efeito), o comprimento positivo ou negativo da distância fornecido em uma unidade de medida (px, pt, cm, em, etc.), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **normal**.
- **text-shadow**: (sombra do texto) adiciona sombra a um texto. Valores: **none** (sem efeito), posicionamento horizontal + posicionamento vertical + raio da desfocagem (opcional) + cor (opcional), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.

FORMATAÇÃO DE LISTAS

Os recursos de formatação baseados em CSS aplicam-se também às listas. Propriedades:

- **list-style-type**: indica o tipo de marcador de uma lista. Valores: **disc** (círculo preenchido), **circle** (círculo vazio), **square** (quadrado), **decimal** (números simples), **decimal-leading-zero** (para valores precedidos de zero – 01, 02...), **lower-roman** (algarismos romanos minúsculos), **upper-roman** (algarismos romanos maiúsculos), **lower-latin** (itens alfabéticos minúsculos), **upper-latin** (itens alfabéticos maiúsculos). Valor padrão: **disc**.
- **list-style-position**: indica o posicionamento dos itens da lista. Valores: **inside** (apresenta a lista alguns pixels à frente do ponto padrão) e **outside** (apresenta a lista em seu ponto padrão). Valor padrão: **outside**.
- **list-style-image**: (Imagem do Marcador da Lista) No local dos marcadores tradicionais, é possível usar imagens, desde que estejam nos formatos **.gif** ou **.jpg**. Para isso, esta propriedade deve ser associada à função **url()**, que estabelece o vínculo da imagem como marcador.

CORES E FUNDOS

Os recursos de formatação de cores e de fundos possibilitam usar efeitos visuais em um documento escrito em HTML.

Para se definir cores e efeitos visuais, pode-se usar as propriedades:

- **color:** aceita como valores três opções: o nome da cor; o código da cor no formato hexadecimal - Exemplo: #00FF00; e o código da cor em formato decimal por meio da função rgb() - Exemplo: rgb(0,0,255).
- **opacity:** define o nível de opacidade de um elemento (nível de transparência), onde 1 não é transparente, 0.5 é 50% transparente e 0 é completamente transparente. Valor padrão: **1**. É possível ainda definir cores e opacidade em uma única propriedade, utilizando a função RGBA. Trata-se de uma extensão de valores de cores RGB com um parâmetro "alfa" - que especifica a opacidade da cor. Desta forma, um valor de cor RGBA é especificado como: rgba (vermelho, verde, azul, alfa). O parâmetro alfa é um número entre 0.0 (totalmente transparente) e 1.0 (totalmente opaco). Exemplo: rgba(255, 0, 0, 0.4).
- **filter:** define efeitos visuais (filtro) em um elemento (normalmente imagens). Dica: para usar vários filtros, separe cada filtro com um espaço. Valores: **none** (sem efeito), **blur(px)** (desfocado), **brightness(%)** (brilho), **contrast(%)** (contraste), **drop-shadow(h-shadow v-shadow blur spread color)** (sombra em imagem), **grayscale(%)** (escala de cinza), **hue-rotate(deg)** (rotação de tonalidade), **invert(%)** (inverter cores – efeito negativo), **opacity(%)** (opacidade em imagem), **saturate(%)** (saturação), **sepia(%)** (sépia), **url(XML)** (SVG - Scalable Vector Graphics em formato XML), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.

A formatação de fundos altera a cor e também apresenta imagens no fundo de algum elemento, como, por exemplo, o corpo de um documento em uso. Para esse tipo de formatação, existe um grupo de propriedades:

- **background-color**: define a cor de fundo de um elemento. Valores: nomes das cores, o código hexadecimal da cor, o código decimal da cor por meio da função rgb() e o valor **transparent** (transparente). Valor padrão: **transparent**.
- **background-image**: determina a imagem de fundo que será apresentada. Valores: o endereço do local onde se encontra a imagem e **none** (nenhum). Como padrão, uma imagem de plano de fundo é colocada no canto superior esquerdo de um elemento e repetida vertical e horizontalmente. Valor padrão: **none**. Exemplo: **url("**Pasta/Figura.xxx**")**
  É possível indicar mais de uma imagem, separando os endereços com vírgulas.
  Exemplo: **url("**Pasta/Figura.xxx**"), url("**Pasta/Figura.xxx**")**
- **background-repeat**: indica a repetição da ilustração de fundo formando um efeito de mosaico ou não. Valores: **repeat** (repete a imagem tanto no sentido vertical como no horizontal), **repeat-x** (repete a imagem no sentido horizontal), **repeat-y** (repete a imagem no sentido vertical), **no-repeat** (apresenta a imagem sem repeti-la), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **repeat**.
- **background-attachment**: (ligação com o fundo) define se a imagem apresentada fica fixa ou se move quando se fizer a rolagem da janela do programa de navegação. Como opções de uso podem ser os valores: **scroll** (o plano de fundo rola junto com o elemento), **fixed** (o plano de fundo é fixo em relação à janela de exibição), **local** (o plano de fundo rola juntamente com o conteúdo do elemento), **initial** (define que deverá ser adotado o valor padrão para este elemento), **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **scroll**.
- **background-position**: (alinhamento do fundo) estabelece o local onde a imagem começa a ser apresentada. A apresentação de uma imagem parte da informação das coordenadas **x** e **y**. Se apenas um valor for especificado, o outro valor será "center". Valores: **top left** (superior esquerdo), **top center** (superior centro), **top right** (superior direito), **center left** (centro esquerdo), **center center** (centro centro), **center right** (centro direito), **bottom left** (inferior esquerdo), **bottom center** (inferior centro), **bottom right** (inferior direito), **x-% y-%** (% horizontal % vertical – o canto superior esquerdo corresponde ao valores 0% 0%, o canto inferior direito corresponde aos valores 100% 100%; se for apenas um valor, o segundo é automaticamente considerado como 50%) e **x-pos y-pos** (posicionamento horizontal posicionamento vertical – baseados nas unidades de medida permitidas em CSS: **in**, **cm**, **mm**, **pt**, **px**, **em**, etc; os valores 0 0 correspondem ao posicionamento superior esquerdo). Valor padrão: **0% 0%**.
- **background-size**: define o tamanho da ilustração de fundo. Devem ser indicados os valores de largura e altura, nesta ordem. Se apenas um valor for informado, o segundo é definido como "auto" e será calculado um valor proporcional ao que foi definido. Os valores de largura e altura também podem ser definidos como uma porcentagem do elemento "pai" a que esta propriedade pertence. Outros valores: **auto** (mantém a largura e altura original da imagem), **cover** (dimensiona a imagem para ser a maior possível para que a área do plano de fundo seja completamente coberta pela imagem), **contain** (dimensiona a imagem para ser a maior possível em que a largura e a altura caibam dentro da área), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **auto**.
- **background-origin**: especifica onde a imagem de plano de fundo está posicionada. Valores: **padding-box** (a ilustração de fundo começa no canto superior esquerdo da borda de preenchimento), **border-box** (a ilustração de fundo começa no canto superior esquerdo da borda), **content-box** (a ilustração de fundo começa no canto superior esquerdo da borda do conteúdo), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **padding-box**.
- **background-clip**: especifica a área de pintura do plano de fundo. Valores: **border-box** (a ilustração de fundo é cortada a partir da borda), **padding-box** (a ilustração de fundo é cortada a partir do preenchimento), **content-box** (a ilustração de fundo é cortada de acordo com o conteúdo), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **padding-box**.

- **background**: indica em uma única declaração todas as propriedades relativas ao formato de fundo, podendo-se usar as opções das propriedades: **background-color**, **background-image**, **background-repeat**, **background-attachment** e **background-position**.
- **backdrop-filter**: adiciona um efeito gráfico (filtro) à área atrás de um elemento (pano de fundo). Nesta propriedade deve ser indicada uma função de filtro ou uma URL para um filtro SVG que será aplicado ao pano de fundo. Outros valores: **none** (nenhum filtro é aplicado ao pano de fundo), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.

***** Background com cores dégradé / gradientes *****

Os gradientes em CSS3 permitem exibir transições suaves entre duas ou mais cores.
O CSS3 apresenta duas funções que permitem criar imagens com efeitos gradientes:

linear-gradient: Gradientes Lineares (para baixo / cima / esquerda / direita / diagonal)

Para criar um gradiente linear deve-se definir pelo menos duas cores para realizar as transições. Também pode-se definir uma direção (ou um ângulo) para o efeito de fundo gradiente.

Sintaxe: **background**: **linear-gradient**(direção, cor1, cor2, ...);

A direção padrão é "to bottom". Outros valores possíveis são: "to top", "to right", "to left", "to bottom right" (diagonal) ou um valor de ângulo positivo ou negativo (45deg, por exemplo). As transições são compostas por um valor de cor, seguido por um posicionamento de parada opcional (uma porcentagem entre 0% e 100% ou um comprimento ao longo do eixo do gradiente).

radial-gradient: Gradientes Radiais (definidos pelo centro)

Para criar um gradiente radial também deve-se definir pelo menos duas cores para realizar as transições.

Sintaxe: **background**: **radial-gradient**(forma tamanho e posicionamento, cor1, cor2, ...);

A forma padrão é "ellipse", o tamanho padrão é "farthest-corner" (canto mais distante) e o posicionamento padrão é "center". Outro valor possível de forma é: "circle". Para o tamanho, outros valores possíveis são: "closest-side", "farthest-side" e "closest-corner". As transições também são compostas por um valor de cor, seguido por um posicionamento de parada opcional (uma porcentagem entre 0% e 100% ou um comprimento ao longo do eixo do gradiente).

QUADROS

Em CSS, os quadros (caixas) apresentam um grande conjunto de propriedades que permite formatar diversos objetos relacionados a um elemento. Para ser formatado, seus limites territoriais devem ser ajustados, com o uso das propriedades:

- **margin**: (margens) elemento invisível de 4 lados: esquerda (left), direita (right), superior (top) e inferior (bottom). O valor da distância das margens pode ser estabelecido em: uma unidade de medida (px, pt, cm, em, etc.), **%** (porcentagem em relação à largura do elemento-pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Também pode ser usada a opção **auto** (centralizada), para que o valor seja calculado automaticamente. Esta propriedade pode ser usada em modo separado: **margin-top** (margem superior), **margin-right** (margem direita), **margin-bottom** (margem inferior) e **margin-left** (margem esquerda) ou de forma mais breve (**margin**) com os valores na ordem: superior, direita, inferior e esquerda. Valor padrão: **0**.
- **padding**: delimita a área do quadro que receberá o efeito de preenchimento. O valor da distância das margens pode ser estabelecido em: uma unidade de medida (px, pt, cm, em, etc.), **%** (porcentagem em relação à largura do elemento-pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Esta propriedade pode ser usada em modo separado: **padding-top** (preenchimento superior), **padding-right** (preenchimento direito), **padding-bottom** (preenchimento inferior) e **padding-left** (preenchimento esquerdo) ou de forma mais breve (**padding**) com os valores na ordem: superior, direita, inferior e esquerda. Valor padrão: **0**.
- **border** (borda geral): delimitam, em um quadro, a apresentação de linhas decorativas.
  A propriedade **border-width** é utilizada para definir a espessura da borda. Valores: **thin** (linha fina), **medium** (linha média), **thick** (linha grossa) ou os valores nas unidades de medida **em**, **rem**, **ex**, **px**, **in**, **cm**, **mm** e **pt**.
  A propriedade **border-style** é utilizada para definir o estilo da borda (formato da linha de uma borda). Valores: **none** ou **hidden** (sem linha de borda), **dotted** (pontilhado), **dashed**

(tracejado), **solid** (linha contínua), **double** (linhas duplas), **groove** (3D Rústica), **ridge** (3D Suave), **inset** (3D com sombra superior) e **outset** (3D com sombra inferior).
A propriedade **border-color** determina a cor da borda. Valores: nomes das cores, o código hexadecimal da cor ou o código decimal da cor por meio da função **rgb()**.
Pode-se usar as propriedades de largura (**width**), estilo (**style**) e cor (**color**) das bordas juntas, utilizando as propriedades **border-bottom** (borda inferior), **border-top** (borda superior), **border-left** (borda esquerda) e **border-right** (borda direita) com os valores na ordem: largura, estilo e cor.
Também é possível utilizar a propriedade **border** de forma mais breve, informando os valores na ordem: largura, estilo e cor.
A propriedade **border-radius** é utilizada para definir o raio dos cantos do elemento. Esta propriedade permite adicionar bordas arredondadas aos elementos. Valores: formas do canto superior esquerdo, canto superior direito, canto inferior direito e canto inferior esquerdo em porcentagem, **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **0**.
Observação: a propriedade **border** também pode ser aplicada em imagens (tag <img>). Existe ainda a opção de especificar uma imagem para ser usada como borda de um elemento. Para isso, podem ser usadas as propriedades: **border-image-source** (especifica a função URL com o caminho do arquivo da imagem de borda), **border-image-slice** (especifica o fatiamento da imagem de borda), **border-image-width** (especifica a largura da imagem da borda), **border-image-outset** (especifica a extensão da imagem de borda além da caixa) e **border-image-repeat** (especifica como será a repetição da imagem da borda, que pode ser repetida, arredondada, espaçada ou esticada). Existe ainda a propriedade **border-image**, que é uma propriedade abreviada para definir todas as propriedades border-image-*.

- **height** e **width**: são utilizadas respectivamente para indicar a altura e a largura de um quadro. O valor pode ser estabelecido em: uma unidade de medida absoluta (**px**, **pt**, **cm**, **in** etc.), uma unidade de medida relativa, como **em**, **rem**, **vw**, **vh** e **%** (porcentagem em relação à largura ou altura do elemento-pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **auto** (o navegador calcula a altura ou largura).
- **outline**: aplica linhas de contorno em volta de um elemento para destacá-lo.
  A propriedade **outline-color** é utilizada para definir a cor da linha de contorno. Aceita como valores: nomes das cores, o código hexadecimal da cor ou o código decimal da cor por meio da função **rgb()**. Também é possível fazer uso do valor **invert** que assegura sempre a apresentação da linha de contorno, não importando a cor utilizada de fundo. Se as duas forem iguais, a linha de contorno será apresentada na cor inversa à cor utilizada de fundo em uso. Valor padrão: **invert**.
  A propriedade **outline-style** indica o estilo da linha de contorno. Valores: **none** (sem linha), **dotted** (pontilhado), **dashed** (tracejado), **solid** (linha contínua), **double** (linhas duplas), **groove** (3D Rústica), **ridge** (3D Suave), **inset** (3D com sombra superior) e **outset** (3D com sombra inferior). Valor padrão: **none**.
  A propriedade **outline-width** é utilizada para definir a espessura da linha de contorno. Valores: **thin** (linha fina), **medium** (linha média), **thick** (linha grossa) ou os valores nas unidades de medida **em**, **ex**, **px**, **in**, **cm**, **mm** e **pt**. Valor padrão: **medium**.

BLOCOS

As tags ** ** e ** ** determinam o agrupamento ou divisão de blocos de tags em seções. Permite a aplicação de efeitos de formatação a um determinado bloco de texto.
As tags **** e **** possuem a mesma função das tags e , com a diferença que a formatação é limitada a um trecho do texto que pode ser uma frase, uma palavra ou uma letra. Para os blocos podem ser aplicadas as mesmas propriedades dos quadros e mais algumas propriedades:

- **top:** coordenada superior.
- **right:** coordenada direita.
- **bottom:** coordenada inferior.
- **left:** coordenada esquerda.
- **position:** (posição) permite posicionar um elemento na tela usando as coordenadas definidas nas propriedades top, right, bottom e left. Valores: **static** (valor padrão para qualquer elemento que não tenha sua posição explicitamente declarada; não permite definir as coordenadas para posicionar o elemento), **fixed** (fixa a posição do elemento na coordenada definida; à medida que a página é rolada, o elemento continua fixo na posição definida e o

conteúdo da página rola normalmente), **absolute** (posição absoluta em relação ao ponto superior esquerdo de um elemento "pai". Se o elemento "pai" não existir, utiliza o "body"), **relative** (posição relativa a si mesmo. Ou seja, o ponto zero será o canto superior esquerdo do próprio elemento, e ele começará a contar a partir dali).

- **z-index:** especifica um número de ordem de exibição de um elemento. Um elemento com maior número de ordem estará sempre na frente de um elemento com número de ordem inferior. Esta propriedade só funciona em elementos onde a propriedade **position** estiver configurada como **absolute**, **relative** ou **fixed**. Valor padrão: **auto** (igual seus pais).
- **float**: quando se trabalha com blocos, pode-se desejar que um elemento flutue à esquerda ou à direita do restante do conteúdo. A propriedade **float** (lado para flutuar) permite definir o local onde o elemento será apresentado, sem fixá-lo exclusivamente neste local. Valores: **none** (o elemento não flutua e será exibido exatamente onde ele ocorre no texto), **left** (esquerda), **right** (direita), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.
- **clear:** quando se usa a propriedade **float**, pode ser necessário utilizar a propriedade **clear** para indicar em qual lado do elemento não são permitidos outros elementos flutuantes. Valores: **none** (permite elementos flutuantes em ambos os lados), **left** (não permite elementos flutuantes no lado esquerdo), **right** (não permite elementos flutuantes no lado direito) ou **both** (não permite elementos flutuantes em ambos os lados), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.
- **clip:** permite especificar um retângulo para cortar um elemento absolutamente posicionado. O retângulo é especificado como quatro coordenadas, todas do canto superior esquerdo do elemento a ser cortado. A propriedade de **clip** não funciona se a propriedade **overflow** estiver configurada para **visible**. Valores: **auto** (sem corte), a forma do elemento representado por: **rect (top, right, bottom, left)**, **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **auto**.
- **display:** especifica a disposição do elemento HTML. Alguns valores: **inline** (exibe um elemento em linha), **block** (exibe um elemento como um bloco), **flex** (exibe um elemento como um contêiner flexível - FLEXBOX), **grid** (exibe um elemento como um contêiner com layout de grade, com linhas e colunas), **inline-block** (exibe um elemento como um bloco em linha), **inline-flex** (exibe um elemento como um contêiner flexível em linha), **table** (deixa o elemento se comportar como um elemento <table>), **none** (o elemento é removido) etc. Valor padrão: **inline**.
- **gap**: especifica o tamanho da lacuna entre linhas e colunas. É uma abreviação para as propriedades: **row-gap** e **column-gap**. Portanto, pode ser usada em modo separado: **row-gap** (lacuna entre linhas) e **column-gap** (lacuna entre colunas) ou de forma mais breve (**gap**) com os valores na ordem para: linha e coluna. Valor padrão: **normal normal**.
- **grid-template-columns**: especifica quantidade e largura das colunas em um contêiner com layout de grade. Os valores são uma lista separada por espaços, onde cada valor especifica o tamanho da respectiva coluna. Nesta propriedade é possível usar um comprimento flexível com a unidade **fr**, que representa uma fração do espaço restante no contêiner com layout de grade. Valores: **auto** (determinado pelo tamanho do contêiner e do conteúdo), **max-content** (tamanhos dependem do maior item na coluna), **min-content** (tamanhos dependem do menor item na coluna), uma unidade de medida absoluta ou relativa e **none** (nenhum tamanho é definido; colunas criadas conforme a necessidade). Valor padrão: **none**.
- **grid-template-rows**: especifica quantidade e altura das linhas em um contêiner com layout de grade. Os valores são uma lista separada por espaços, onde cada valor especifica o tamanho da respectiva linha. Nesta propriedade é possível usar um comprimento flexível com a unidade **fr**, que representa uma fração do espaço restante no contêiner com layout de grade. Valores: **auto** (determinado pelo tamanho do contêiner e do conteúdo), **max-content** (tamanhos dependem do maior item na linha), **min-content** (tamanhos dependem do menor item na linha), uma unidade de medida absoluta ou relativa e **none** (nenhum tamanho é definido; linhas criadas conforme a necessidade). Valor padrão: **none**.
- **grid-template-areas**: estrutura de áreas do grid. Especifica áreas dentro de um contêiner com layout de grade. Faz referência aos itens da grade que foram nomeados usando a propriedade grid-area. Cada área é delimitada por aspas e pode ser usado um ponto para se referir a um item sem nome. Valores: uma sequência que especifica como cada coluna e linha deve ser estruturada e **none** (nenhuma área nomeada). Valor padrão: **none**.

- **grid-area**: nome da área do grid. Esta propriedade possui duas funções: 1) atribuir um nome a um item de grade. Os itens de grade nomeados podem ser referenciados pela propriedade grid-template-areas em um contêiner com layout de grade; 2) especificar o tamanho e a localização de um item em um contêiner com layout de grade. Também é uma propriedade abreviada para as propriedades: **grid-row-start**, **grid-column-start**, **grid-row-end** e **grid-column-end**. Valores: indicar os valores na ordem para exibir os itens: linha em que inicia, coluna em que inicia, linha em que finaliza e coluna em que finaliza. Valor padrão: auto / auto / auto / auto.
- **max-height, max-width, min-height** e **min-width**: são utilizadas para indicar respectivamente: altura máxima, largura máxima, altura mínima e largura mínima de um elemento. O valor pode ser estabelecido em: uma unidade de medida absoluta (px, pt, cm, em, etc.), ou relativa, como **%** (porcentagem em relação ao elemento-pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.
- **overflow:** especifica o que deve ser feito se um elemento é muito grande para caber em uma área especificada (recorte). Esta propriedade especifica quando se deve cortar o conteúdo ou adicionar barras de rolagem. Esta propriedade só funciona para elementos de bloco com uma altura especificada. Esta propriedade pode ser usada em modo separado: **overflow-x** (recorte horizontal), **overflow-y** (recorte vertical). Valores: **visible** (o excesso não é cortado, ou seja, é apresentado fora do elemento), **hidden** (o excesso é cortado, ou seja, o resto do conteúdo estará invisível), **scroll** (o excesso é cortado, mas uma barra de rolagem é adicionada para se visualizar o resto do conteúdo), **auto** (se o excesso for cortado, uma barra de rolagem pode ser adicionada para se visualizar o resto do conteúdo), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **visible**.
- **visibility:** especifica quando um elemento é visível ou não (visibilidade). Mesmo os elementos invisíveis ocupam espaço na página. Utilizar a propriedade display para criar elementos invisíveis que não ocupam espaço. Valores: **visible** (o elemento é apresentado), **hidden** (o elemento é estará invisível, mas ocupando espaço), **collapse** (apenas para tabelas: remove uma linha ou coluna, mas não afeta o layout da tabela. O espaço ocupado pela linha ou coluna estará disponível para outros conteúdos. Se for usado em outros elementos, ele fica oculto), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **visible**.
- **vertical-align:** define o alinhamento vertical de um elemento. Valores: **baseline** (alinha a linha de base do elemento com a linha de base do elemento pai), altura (aumenta ou diminui um elemento pela altura especificada. Valores negativos são permitidos), **%** (aumenta ou diminui um elemento em um percentual da propriedade line-height. Valores negativos são permitidos), **sub** (alinha o elemento como se fosse subscrito), **super** (alinha o elemento como se fosse subrescrito), **top** (a parte superior do elemento é alinhada com a parte superior do elemento mais alto da linha), **text-top** (a parte superior do elemento é alinhada com a parte superior da fonte do elemento pai), **middle** (o elemento é apresentado no meio do elemento pai), **bottom** (a parte inferior do elemento é alinhada com o elemento mais baixo na linha), **text-bottom** (a parte inferior do elemento é alinhada com a parte inferior da fonte do elemento pai), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **baseline**.
- **box-shadow**: adiciona sombra em elemento. Valores: **none** (sem efeito), posicionamento horizontal + posicionamento vertical + raio da desfocagem (opcional) + raio de propagação (opcional) + cor (opcional), **inset** (sombra interna), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **none**.
- **box-sizing**: Indica ao navegador quais devem ser as propriedades de dimensionamento (largura e altura). Valores: **content-box** (as propriedades de largura e altura incluem apenas o conteúdo. Bordas, preenchimentos ou margens não estão incluídas), **border-box** (as propriedades de largura e altura incluem conteúdo, bordas e preenchimentos. As margens não estão incluídas), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **content-box**.

***** FLEXBOX *****

A utilização de caixas flexíveis, também conhecidas como flexbox, é uma nova forma de organizar layouts de blocos (DIVs) no CSS3. O uso de flexbox garante que os elementos se comportem de forma previsível quando o layout da página deve se ajustar a diferentes tamanhos de tela e diferentes dispositivos. Um flexbox consiste em contêineres flexíveis e itens flexíveis. Dentro de um contêiner flexível existe um ou mais itens flexíveis.

Um contêiner flex é declarado a partir da definição da propriedade **display** do elemento para **flex** (renderizado como um bloco) ou **inline-flex** (renderizado como inline).

Propriedades aplicadas a flexbox:

- **flex-direction:** especifica a direção dos itens dentro do container. Valores: **row** (da esquerda para direita e de cima para baixo), **row-reverse** (inverte a direção de escrita previamente adotada), **column** (se o sistema de escrita for horizontal, os itens serão dispostos verticalmente), **column-reverse** (inverte a direção da coluna previamente adotada). Valor padrão: **row**.
- **justify-content:** especifica o espaçamento entre os itens do container. Esta propriedade alinha horizontalmente os itens do contêiner quando os itens não usam todo o espaço disponível no eixo principal. Valores: **flex-start** (os itens são posicionados a partir do início do contêiner), **flex-end** (os itens são posicionados a partir do fim do contêiner), **center** (os itens são posicionados a partir do centro do contêiner), **space-between** (os itens são posicionados com espaços entre as linhas), **space-around** (os itens são posicionados com espaços antes, entre e depois das linhas). Valor padrão: **flex-start**.
- **align-items:** especifica o alinhamento dos itens do container. Esta propriedade alinha verticalmente os itens do contêiner quando os itens não usam todo o espaço disponível no eixo principal. Valores: **stretch** (os itens são esticados para caber no contêiner), **flex-start** (os itens são posicionados a partir do topo do contêiner), **flex-end** (os itens são posicionados a partir do fim do contêiner), **center** (os itens são verticalmente posicionados a partir do centro do contêiner), **baseline** (os itens são posicionados a partir da linha base do contêiner). Valor padrão: **stretch**.
- **flex-wrap:** especifica se deve quebrar para a próxima linha ou não, quando não houver espaço suficiente para os itens em uma linha. Valores: **nowrap** (os itens não serão quebrados), **wrap** (os itens serão quebrados, se necessário), **wrap-reverse** (os itens serão quebrados, se necessário, na ordem inversa). Valor padrão: **nowrap**.
- **align-content:** modifica o comportamento da propriedade flex-wrap. Esta propriedade é similar à propriedade align-items, mas ao invés de alinhar itens, ela alinha linhas. Valores: **stretch** (as linhas são esticadas para cobrir o espaço restante), **flex-start** (as linhas são posicionadas a partir do início do contêiner), **flex-end** (as linhas são posicionadas a partir do fim do contêiner), **center** (as linhas são verticalmente centralizadas no contêiner), **space-between** (as linhas são uniformemente distribuídas no contêiner), **space-around** (as linhas são uniformemente distribuídas no contêiner, com metade do tamanho em cada extremidade). Valor padrão: **stretch**.
- **order:** especifica a ordem de um item em relação aos demais itens do mesmo contêiner.
- **margin**: usada para mover itens para posições diferentes.
- **align-self**: substitui a propriedade align-items. Apresenta os mesmos valores da propriedade align-items.
- **flex:** especifica o comprimento flexível de um item em relação aos demais itens do mesmo contêiner.

ESTILOS EM TABELAS

As tags **<table>**, **<th>**, **<td>** e **<tr>** possuem um conjunto de propriedades que permitem a definição de bordas, larguras, fundo e centralização:

- **border**: apresenta as linhas de borda de uma tabela. Deve-se usar as propriedades de largura (**width**), estilo (**style**) e cor (**color**) das bordas juntas para se obter o efeito desejado. Vide a propriedade **border** dos Quadros.
- **width**: é utilizada para indicar a largura de uma tabela. O valor pode ser estabelecido em **px** (pixels) ou em formato percentual.
- **background-color**: define a cor de fundo de uma tabela. Valores: nomes das cores, o código hexadecimal da cor, o código decimal da cor por meio da função **rgb()** e o valor **transparent** (transparente). Valor padrão: **transparent**.

- **background-image**: determina a imagem de fundo de uma tabela. Valores: o endereço do local onde se encontra a imagem e **none** (nenhum). Valor padrão: **none**.
- **margin**: o valor **auto** na propriedade **margin** centraliza a tabela verticalmente no navegador.
- **caption**-**side:** especifica o posicionamento da legenda da tabela. Valores: **top** (acima da tabela), **bottom** (abaixo da tabela), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **top**.

ESTILOS EM FORMULÁRIOS

As tags **<fieldset>** e **</fieldset>** apresentam um quadro que cerca um conjunto de elementos do formulário. Pode ser usado junto com as tags **<legend>** e **</legend>**.
As demais tags usadas em formulário (<form>, <label>, <button>, <input>, etc.) também podem ter estilos aplicados via CSS.

ESTILOS DE TRANSFORMAÇÃO

Aplica uma transformação 2D ou 3D para um elemento. Propriedade: **transform**. Valores: **none** (sem transformação), **matrix** (define uma transformação usando uma matriz), **translate** (muda a posição de um elemento), **scale** (define uma transformação de escala), **rotate** (define uma rotação), **skew** (define uma inclinação), **perspective** (define uma vista em perspectiva), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai").

TRANSIÇÕES

Os efeitos de transições estipulam um tempo de duração enquanto valores de propriedades são alterados. Para criar um efeito de transição deve-se especificar pelo menos: a propriedade CSS que receberá o efeito e o tempo de duração do efeito. O efeito de transição será iniciado quando a propriedade CSS especificada for alterada. Pode ser indicada mais que uma propriedade para receber o efeito. Para isto, deve-se separar cada combinação de propriedade + duração com vírgula. Versão abreviada:

- **transition**: propriedade duração velocidade(opcional) atraso(opcional);

Propriedades de transição:

- **transition-property:** (propriedade da transição) especifica o nome da propriedade CSS que receberá o efeito. Valores: **all** (todas as propriedades alteradas vão receber o efeito), **none** (nenhuma propriedade alterada vai receber o feito), **propriedade CSS**, **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **all**.
- **transition-duration:** (duração da transição) especifica a duração em segundos ou milissegundos de um efeito de transição. Valor padrão: **0**.
- **transition**-**timing**-**function**: (velocidade da transição) especifica a velocidade do efeito de transição. Valores: **ease** (início lento, depois rápido e termina lentamente), **linear** (mesma velocidade do início ao fim), **ease-in** (início lento), **ease-out** (fim lento), **ease-in-out** (início e fim lento), **cubic-bezier(n,n,n,n)** (permite definir valores em uma função Cúbica de Bézier). Valor padrão: **ease**.
- **transition**-**delay**: (atraso da transição) especifica um atraso em segundos (s) ou milissegundos (ms) para o início da transição. Valor padrão: **0s**
- **scroll-behavior**: rolagem suave da página. A propriedade deve ser adicionada à tag "**html**". Valores: **auto** (salto direto entre elementos), **smooth** (efeito animado e suave), **initial** (define que deverá ser adotado o valor padrão para este elemento) e **inherit** (define que deverá ser adotado o valor do elemento "pai"). Valor padrão: **auto**.

ANIMAÇÕES

Uma animação permite que um elemento mude gradualmente de um estilo para outro. Para usar animação CSS, é necessário especificar alguns @keyframes para a animação.

** @keyframes **

Os @keyframes são códigos identificados por um nome que contêm os estilos que serão aplicados ao elemento, em cada momento especificado. Os estilos especificados dentro da regra @keyframes irão mudar gradualmente, passando do estilo atual para o novo estilo nos momentos determinados. Portanto, especificamos quando o estilo será alterado usando os seletores "from" e "to", que equivalem 0% (início) e 100% (completo). Também é possível usar

porcentagem, variando os valores de 0 a 100%. Neste caso, é possível adicionar quantas alterações de estilo desejar.
Sintaxe: @keyframes nome { seletores { propriedade_css: valor; } }
**
Versão abreviada:

- **animation**: nome duração velocidade atraso repetição direção preenchimento estado;

Propriedades de animação:

- **animation-name:** especifica o nome da animação, referente ao @keyframe que será vinculado ao elemento. Valor padrão: **none**.
- **animation-duration:** especifica a duração da animação, ou seja, quanto tempo uma animação deve levar para completar um ciclo. Valor padrão: **0** (sem animação).
- **animation-timing-function**: especifica a velocidade da animação. Valores: **ease** (início lento, depois rápido e termina lentamente), **linear** (mesma velocidade do início ao fim), **ease-in** (início lento), **ease-out** (fim lento), **ease-in-out** (início e fim lento), **steps**(int, start|end) (escalonamento), **cubic-bezier(n,n,n,n)** (permite definir valores em uma função Cúbica de Bézier). Valor padrão: **ease**.
- **animation-delay**: (atraso da animação) especifica um atraso em segundos (s) ou milissegundos (ms) para o início da animação. Valor padrão: **0s**
- **animation-iteration-count**: (repetição da animação) define quantas vezes uma animação deve ser reproduzida. Valores: **valor** (número de vezes), **infinite** (infinitas vezes. Valor padrão: **1**.
- **animation-direction**: (direção da animação) define se uma animação deve ser reproduzida para frente, para trás ou em ciclos alternados. Valores: **normal** (para frente), **reverse** (para trás), **alternate** (para frente e depois para trás), **alternate-reverse** (para trás e depois para frente). Valor padrão: **normal**.
- **animation-fill-mode**: (preenchimento da animação) define um estilo para o elemento quando a animação não está sendo reproduzida (antes de começar, depois de terminar ou ambos). Valores: **none** (nenhum), **forwards** (último estilo), **backwards** (primeiro estilo), **both** (estende as propriedades em ambas direções). Valor padrão: **none**.
- **animation-play-state**: (estado da animação) determina se a animação está em execução ou pausada. Valores: **running** (em execução), **paused** (em pausa). Valor padrão: **running**.

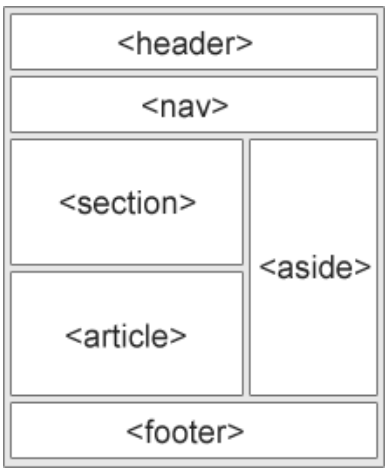
CURSORES

A propriedade **cursor** especifica o cursor do mouse a ser exibido ao apontar sobre um elemento.
Alguns valores: **auto** (o navegador determina o cursor), **grab** (indica que algo pode ser agarrado), **help** (indica que a ajuda está disponível), **none** (nenhum cursor), **pointer** (ponteiro que indica um link), **text** (indica que um texto pode ser selecionado), **wait** (indica que o programa está ocupado), **zoom-in** (pode ser ampliado), **zoom-out** (pode ser reduzido) etc.

ELEMENTOS SEMÂNTICOS:

O HTML5 incorporou novas tags com significados semânticos. Em vez de agrupar os elementos do cabeçalho em uma div genérica e sem significado, é possível usar uma tag **<header>**, por exemplo, que carrega em si o significado de representar um cabeçalho.
Com isso, têm-se um HTML com estrutura baseada no significado de seu conteúdo, o que traz uma série de benefícios, como a facilidade de manutenção e compreensão do documento.
Portanto, o HTML5 adicionou novos elementos semânticos para definir diferentes partes de uma página da Web:

- **<article>** - Define um artigo
- **<aside>** - Define um conteúdo na lateral da página
- **<details>** - Define detalhes adicionais que o usuário pode visualizar ou ocultar
- **<figure>** - Define um conteúdo independente, como ilustrações, diagramas, fotos etc.
- **<figcaption>** - Define uma legenda para um elemento <figure>
- **<footer>** - Define um rodapé para um documento ou seção
- **<header>** - Define um cabeçalho para um documento ou seção



- **<main>** - Define o conteúdo principal de um documento
- **<mark>** - Define um texto marcado ou realçado
- **<nav>** - Define links de navegação
- **<section>** - Define uma seção em um documento
- **<summary>** - Define um título visível para um elemento <details>
- **<time>** - Define uma data/hora

Regras e Elementos do CSS

SELETORES

Em CSS, os seletores permitem o uso de diversos efeitos de formatação em um mesmo elemento (uma tag, por exemplo). Com os seletores é possível escolher qualquer elemento numa página para aplicar estilos. Principais tipos de seletores:

UNIVERSAL: Seleciona todos os elementos HTML da página. Sintaxe:

```
* {
    propriedade: valor;
}
```

TIPO: Este seletor permite selecionar todas as tags de um mesmo tipo para aplicar um estilo. Sintaxe:

```
elemento {
    propriedade: valor;
}
```

DESCENDENTE: Este seletor permite escolher um ou mais elementos que estão dentro de outro, ou seja, que são descendentes do elemento principal. Sintaxe:

```
elemento-pai elemento-filho {
    propriedade: valor;
}
```

ID: seleciona o elemento com um id especificado. O nome do id é escolhido pelo programador e não deve ser iniciado por um número. Cada id é único e não pode ser repetido no mesmo documento. Normalmente é utilizado para identificar elementos estruturais da página. Para usar um id é necessário alterar dois pontos do código: a tag que receberá a ação do id e o arquivo CSS. Sintaxe:

```
<tag id="destaque">...</tag>

#destaque {
   propriedade: valor;
}
```

CLASS: seleciona elementos com uma classe (class) específica aplicada. O nome da classe é escolhido pelo programador e não deve ser iniciado por um número. Uma classe é reutilizável: pode se repetir na página e também combinar-se com outras (é possível colocar mais de uma classe em um elemento, separando por espaço). Para usar uma classe é necessário alterar dois pontos do código: a tag que receberá a ação da classe e o arquivo CSS. Sintaxe:

```
<tag class="destaque">...</tag>

.destaque {
   propriedade: valor;
}
```

Combinação de Seletores: é possível combinar seletores para selecionar partes mais específicas de um elemento. Exemplo de seleção de um elemento associado a uma classe:

```
seletor.classe {
   propriedade: valor;
}
```

Também é possível separar seletores por vírgulas para selecionar diferentes elementos para aplicar os mesmos estilos. É útil para diminuir a repetição de regras no arquivo CSS. Exemplo:

```
seletor1, seletor2 {
   propriedade: valor;
}
```

PSEUDO-CLASSES: são tipos de classes especiais que não são definidas pelo desenvolvedor (já são pré-definidas). Permitem definir um estilo para ser aplicado a um elemento quando ele atingir um estado especial. Sintaxe:

```
seletor:pseudo-classe {
    propriedade: valor;
}
```

Algumas pseudo-classes:
:active – ao ativar o elemento. Por exemplo, quando se clica em um link e não se solta o botão do mouse. O estilo é aplicado entre o momento que se aperta e solta o mouse (nesse caso, é algo bem rápido).
:hover – ao passar o mouse em cima do elemento.
:focus – ao receber o foco. Muito utilizado em campos de texto.
:checked – elementos marcados
:disabled – elementos desabilitados
:enabled – elementos habilitados
:empty – elementos sem filhos
:link – antes de clicar no link.
:visited – ao visitar o link.
:lang – definir regras especiais para diferentes idiomas.
:root – corresponde ao elemento raiz do documento.

PSEUDO-ELEMENTOS: permite definir um estilo para ser aplicado a uma parte de um elemento. Sintaxe:

```
seletor::pseudo-elemento {
    propriedade: valor;
}
```

Alguns pseudo-elementos:
::after e **::before** – permitem adicionar conteúdo antes (before) e/ou depois (after) de um elemento.
::first-letter – permite adicionar um estilo à primeira letra de um elemento.
::first-line – permite adicionar um estilo à primeira linha de um elemento.
::selection – permite adicionar um estilo no momento em que um elemento (ou parte dele) é selecionado.
::file-selector-button – botão de seleção para um campo (input) do tipo arquivo

SELETORES POR ATRIBUTOS: seleciona elementos a partir de um atributo ou de um valor de atributo específico. Sintaxe:

```
seletor[atributo] {
    propriedade: valor;
}

seletor[atributo="valor"] {
    propriedade: valor;
}
```

Também é possível selecionar elementos em que o valor de um atributo contenha uma palavra (inteira e sozinha) ou um valor especificado. Sintaxe:

```
[atributo~="palavra"] {
    propriedade: valor;
}

[atributo*="valor"] {
    propriedade: valor;
}
```

Outra opção é selecionar elementos em que o valor de um atributo começa com um valor específico. Para esta situação, o valor deve ser uma palavra inteira, sozinha, ou seguida por um hífen (-). Sintaxe:

```
[atributo|="valor"] {
    propriedade: valor;
}
```

Semelhante à opção anterior, existe a possibilidade de selecionar elementos em que o valor de um atributo começa com um valor específico, mas em que não é necessário ser uma palavra inteira. Sintaxe:

```
[atributo^="valor"] {
    propriedade: valor;
}
```

Também é possível selecionar elementos em que o valor de um atributo termine com um valor específico. Sintaxe:

```
[atributo$="palavra"] {
    propriedade: valor;
}
```

Em formulários, os seletores de atributo podem ser úteis para aplicar estilos em elementos dos formulários sem utilizar classe ou ID. Exemplos:

```
input[type="text"] {
    propriedade: valor;
}

input[type="button"] {
    propriedade: valor;
}
```

Funções CSS

As funções CSS são usadas como um valor para diferentes propriedades CSS. Exemplos:

attr()	Retorna o valor de um atributo do elemento.
calc()	Permite calcular valores das propriedades CSS.
repeat()	Representa um fragmento repetido. Permite que um grande número de colunas ou linhas recorrentes sejam escritas de forma mais compacta.
var()	Insere o valor de uma propriedade personalizada (variável previamente declarada).

Variáveis CSS

Variáveis CSS são elementos que armazenam valores específicos para serem reutilizados ao longo do código.
As variáveis são declaradas usando "—" (dois traços) seguido pelo nome que identifica a variável. Exemplo: `--cor-principal: black;`
As variáveis são acessadas usando a função var(). Exemplo: `color: var(--cor-principal);`

Importação de Fontes

Existem duas maneiras de importar fontes. É possível usar a regra CSS **@import** ou a tag HTML **<link>**. Quando usamos a regra CSS @import, ela deve ser sempre a primeira linha do arquivo CSS ou a primeira instrução da tag <style>. A regra @import deve ser seguida por uma url ou string que representa a localização do recurso a ser importado. Para importar a fonte usando a tag HTML <link>, ela deve ser adicionada à tag de cabeçalho <head>. Após importar a fonte, é necessário especificar a propriedade font-family em cada elemento que irá utilizar a fonte importada. A regra **@import** também permite importar outros arquivos CSS e pode ser combinada a uma media query para condicionar a aplicação das propriedades contidas no arquivo.

Sites Responsivos

O web design responsivo torna uma página web adequada a todos os dispositivos (desktops, tablets e telefones). Para isso, utiliza CSS e HTML para redimensionar, ocultar, diminuir, ampliar ou mover um conteúdo para que se adeque a qualquer tela.
Sendo assim, as páginas Web devem adaptar seu conteúdo para qualquer dispositivo.

Algumas dicas:

- Os usuários costumam rolar páginas verticalmente, mas não horizontalmente. Então não é uma boa prática forçar os usuários a percorrer a página horizontalmente ou reduzir o zoom para ver toda a página.
- NÃO se deve usar largura fixa em elementos grandes. Por exemplo, se uma imagem for exibida em uma largura maior do que a tela, isso pode fazer com que a janela de exibição se desloque horizontalmente. Deve-se ajustar este conteúdo para caber dentro da largura da tela. É indicado usar valores de largura relativos, como largura: 100%. Se a propriedade de largura de uma imagem estiver definida para 100%, ela será responsiva e poderá aumentar e diminuir. Se a propriedade de largura máxima (max-width) for definida como 100%, a imagem diminuirá se for necessário, mas nunca será maior do que o tamanho original.
- Já que as dimensões variam entre os dispositivos, NÃO se deve deixar que o conteúdo dependa de uma largura de exibição particular, referente a um tipo de dispositivo, para ser exibido adequadamente.
- Deve-se aplicar diferentes estilos para diferentes tamanhos de telas.
- Desenvolva designs "Mobile First". Isso significa projetar para celular antes de projetar para desktop ou qualquer outro dispositivo (Isso tornará a exibição da página mais rápida em dispositivos menores). Então, em vez de mudar os estilos quando a largura é menor do que 768px, devemos mudar o design quando a largura for maior do que 768px.
- Se a propriedade de tamanho do fundo (background-size) estiver configurada para "cover", a ilustração de fundo será dimensionada para cobrir toda a área de conteúdo. Ele mantém a proporção e, por conta disso, uma parte da ilustração de fundo pode ser cortada.
- Os elementos HTML devem ter a propriedade de dimensionamento dos blocos (box-sizing) definida como "border-box". Isso garante que o preenchimento e a borda sempre estarão incluídos na largura e altura total dos elementos.

Inspetor de Elementos

Os navegadores Google Chrome e Mozilla Firefox apresentam uma ferramenta que serve para analisar todos os objetos que compõe uma página, trata-se do Inspetor de Elementos. Ele pode ser acessado clicando com o botão direito do mouse sobre qualquer local da página e, em seguida, na opção "Inspecionar" ou "Inspecionar Elemento" no menu de contexto. Outra opção é usar as teclas de atalho: Ctrl + Shift + I.

O navegador exibirá painéis onde é possível ver toda a estrutura do site, com destaque às opções relacionadas ao item que for clicado. Em algumas áreas destes painéis é possível fazer alterações nos códigos e visualizar o resultado, sem que o código seja de fato alterado. Como nada fica gravado, não há risco de prejudicar a página. Se não ficar satisfeito com o resultado, basta recarregar a página. Desta forma, todos os itens que compõe uma página podem ser editados e visualizados através do Inspetor de Elementos.

O Inspetor de Elementos também permite alternar a visualização da página, selecionando o modo responsivo ou escolhendo o dispositivo desejado.

Viewport

Viewport é a área visível do usuário de uma página da web. O viewport varia de acordo com o dispositivo, portanto será menor em um telefone celular do que na tela de um computador. Antes de existirem tablets e telefones celulares, as páginas eram projetadas apenas para telas de computador e era comum que as páginas possuíssem um projeto estático e um tamanho fixo. Estas páginas de tamanho fixo eram muito grandes para caber no viewport de tablets e telefones

celulares. Para corrigir isso rapidamente, os navegadores desses dispositivos reduziam as páginas para se ajustarem à tela, mas esta solução não foi a ideal.

A maioria dos navegadores de dispositivos pequenos dimensionam as páginas HTML para a largura do viewport, fazendo com que as páginas se encaixem proporcionalmente à tela. O HTML5 incluiu a <meta> tag viewport para permitir que os web designers assumissem o controle sobre a viewport. Portanto, a meta tag viewport permite redefinir isso:

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

Neste exemplo, a meta tag viewport diz ao navegador para usar como largura do layout, a largura do viewport, desativando a escala inicial.
Portanto, este elemento fornece instruções ao navegador sobre como controlar as dimensões e a escala da página. A largura da página pode ser definida para seguir a largura da tela do dispositivo e a escala inicial permite definir o nível de zoom quando a página for carregada pela primeira vez pelo navegador.

Media Query

Media Query é uma técnica introduzida no CSS3. Ela usa a regra **@media** para incluir um bloco de propriedades CSS somente se uma determinada condição for verdadeira. Com as Media Queries o CSS é capaz de consultar o navegador para verificar o tipo de mídia ou dispositivo que o usuário está usando. Com isso é possível especificar estilos diferentes para cada tipo de mídia. Esta consulta ocorre em tempo real. Desta forma, as Media Queries são uma forma de enviar informações ao navegador sobre como renderizar a página, dependendo do tamanho do viewport. O propósito das Media Queries é aplicar estilos diferentes, de acordo com o tamanho do viewport do dispositivo. Estes estilos podem ser criados em um único arquivo CSS ou em arquivos separados.

Exemplo
Se a janela do navegador for a partir de 768px (largura mínima de 768px), a cor do plano de fundo deve mudar para amarelo:

```
@media (min-width: 768px) {
    body {
        background-color: yellow;
    }
}
```

É possível utilizar o min-device-width em vez de min-width para verificar a largura do dispositivo em vez da largura do navegador.

Uma Media Query também pode ser usada para alterar o layout de uma página dependendo da orientação do navegador. Pode ser definido um conjunto de propriedades CSS que só serão aplicadas se a janela do navegador for maior do que sua altura, a chamada orientação “Landscape” ou "Paisagem":

```
@media (orientation: landscape) {
    body {
        color: blue;
    }
}
```