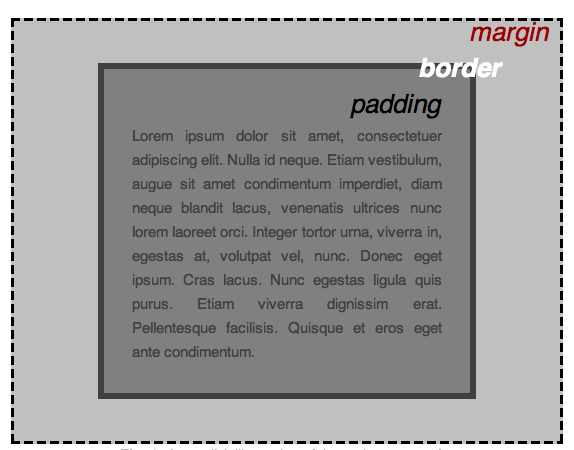
COMANDOS CSS / FORMATAÇÃO

* padding, o espaço ao redor do conteúdo (ex.: ao redor do texto de um parágrafo).
* border, a linha sólida do lado de fora do padding.
* margin, o espaço externo a um elemento.

Exemplo:



* width (largura de um elemento).
* background-color, a cor atrás do conteúdo de um elemento e do padding.
* color, a cor do conteúdo de um elemento (geralmente texto).
* text-shadow: cria uma sombra no texto dentro de um elemento.
* display: define a maneira de dispor um elemento (não se preocupe com isso ainda).

MUDANDO A COR DA PÁGINA INTEIRA:

html {

background-color: #00539F;

}

SEPARANDO O CORPO:

body {

width: 600px;

margin: 0 auto;

background-color: #FF9500;

padding: 0 20px 20px 20px;

border: 5px solid black;

}

* width: 600px; — Isso força o corpo a ter 600 pixels de largura.
* margin: 0 auto; — Quando você define dois valores em uma propriedade como margin ou padding, o primeiro valor afeta a parte superior do elemento **e** a parte inferior (tornando-os 0 nesse caso), e no segundo valor os lados esquerdo **e** direito (aqui, auto é um valor especial que divide o espaço horizontal uniformemente entre esquerda e direita). Você também pode utilizar um, três ou quatro valores, como documentado [aqui](https://developer.mozilla.org/pt-BR/docs/Web/CSS/margin#Sintaxe).
* background-color: #FF9500; — como antes, isso define a cor de fundo do elemento. Usamos um tipo de laranja avermelhado no corpo, para opor ao azul escuro no elemento [<html>](https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element/html), mas fique à vontade para ir em frente e experimentar.
* padding: 0 20px 20px 20px; — temos quatro valores definidos no padding, para criar um pouco de espaço em torno do nosso conteúdo. Dessa vez, estamos definindo sem padding na parte superior do corpo, e 20 pixels no lado esquerdo, na parte inferior e no lado direito. Os valores definem a parte superior, o lado direito, a parte inferior e o lado esquerdo, nessa ordem. Como com a margin, você também pode usar um, dois, ou três valores, conforme documentado na [sintaxe do padding](https://developer.mozilla.org/pt-BR/docs/Web/CSS/padding#Sintaxe).
* border: 5px solid black; — isso simplesmente define uma borda preta sólida de 5 pixels de largura em todos os lados do corpo.

POSICIONANDO E ESTILIZANDO O TÍTULO DA PÁGINA:

h1 {

margin: 0;

padding: 20px 0;

color: #00539F;

text-shadow: 3px 3px 1px black;

}

Você deve ter notado que há um espaço horrível no topo do corpo. Isso acontece porque os browsers aplicam algumas **estilizações padrão** ao <h1> (entre outros), mesmo quando você não aplicou nenhum CSS! Isso pode soar como uma má ideia, mas queremos ter uma legibilidade básica, mesmo em uma página sem estilos. Para nos livrarmos desse espaço, sobrescrevemos o estilo padrão, definindo margin: 0;.

Em seguida, definimos o padding das partes superior e inferior do cabeçalho para 20 pixels e fizemos o texto do cabeçalho da mesma cor que a cor de fundo do HTML.

Uma propriedade bastante interessante que usaremos aqui é o text-shadow, que aplica uma sombra ao conteúdo de texto do elemento. Seus quatro valores são os seguintes:

* O primeiro valor em pixel define o **deslocamento horizontal**da sombra do texto — até onde ele se move: um valor negativo deve movê-la para a esquerda.
* O segundo valor em pixel define o **deslocamento vertical** da sombra do texto — o quanto ela se move para baixo, neste exemplo; um valor negativo deve movê-la para cima.
* O terceiro valor em pixel define o **raio de desfoque** da sombra — um valor maior significará uma sombra mais borrada.
* O quarto valor define a cor base da sombra.

De novo, tente experimentar com diferentes valores para ver o que você pode criar!

Centralizando a imagem

img {

display: block;

margin: 0 auto;

}

Finalmente, centralizaremos a imagem para melhorar a aparência. Nós poderiamos usar novamente o truque margin: 0 auto que aprendemos anteriormente para o corpo, mas também precisamos fazer outra coisa. O elemento [<body>](https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element/body) é **em nível de bloco**, o que significa que ocupa espaço na página e pode ter margens e outros valores de espaçamento aplicados a ele. Imagens, por outro lado, são elementos**em linha**, o que significa que não podem ter margens. Então, para aplicar margens a uma imagem, temos que dar o comportamento de nível de bloco a imagem usando display: block;.

FLEX BOX:

Sempre inicie com Display: Flex; para criar um container flexível a nível de bloco ou inline-flex para fazer o mesmo só que inline.

Agora começaremos a brincar:

.container {

**flex-direction**: *row / row-reverse*; (coloca em linha na ordem e inversamente)

**flex-direction:** *column / columreverse*; ( alinha em coluna de cima para baixo e inversamente

}

.container {

**flex-wrap**: *now-wrap*; ( deixa todos elementos na mesma linha. Mesmo q tenha muitos )

**flex-wrap**: *wrap*; ( organiza os elementos caso sejam maior do que o container, distribuindo para próxima linha.)

**flex-wrap**: *wrap-reverse*; ( mesma coisa, porém inversamente)

}

.container {

**justify-content**: *flex-start / flex-end / center / space-between / space-around / space-evenly;*

(Define o alinhamento em relação ao eixo principal )

}

.container {

**align-items**: *stretch / flex-start / fle-end / center / baseline;*

(Define o alinhamento dos elementos em relação ao eixo transversal.)

}

.container {

**aligne-content**: *flex-star / flex-end / center / stretch / space-between / space-around;*

( alinha os elementos dentro do container de forma a remover os espaços que aparentam sobrar)

}

.item {

**order**: *<integer>;* ( alinha a ordem dos items )

}

.item {

**flex-grow**: *<número>;* ( ira distribuir um tamanho para o item igual, removendo espaços)

}

.item {

**Flex-shrink**: *<número>*; ( encolhe o item dentro do container)

}

.item {

align-self: ;auto / flex-start / flex-end / center / baseline / stretch (Ela alinha um único item no container)

}