Em nosso site, temos uma descrição com o título "Sobre a Barbearia Alura". Trata-de se informações que são fechadas em si mesmas, como uma sessão. Em nosso código esse conteúdo está sendo envolvido por uma <div>, mas para casos assim utilizamos a tag <section>.

A diferença entre <div> e <section> é que no primeiro caso, trata-se apenas de uma divisão visual. Já no caso da <sectiom> teremos uma divisão por conteúdo complexo, semanticamente homogêneo.

É importante dizer que que a classe titulo-centralizado não possui um bom nome, afinal sempre que criamos uma classe do CSS é importante que ela seja específica e ao mesmo tempo o mais genérica possível. Quando temos um nome como "titulo centralizado", estamos imprimindo um comportamento, então se quiséssemos alinhar o título à esquerda, teríamos de modifcar o nome da classe, ou criar uma nova.

Portanto, alteraremos o nome da classe para titulo-principal, e em style.css faremos as especificações de estilo. As medidas proporcionais no CSS são um assunto crucial no mento de definir escalas, tamanho de fonte e assim por diante. Até agora em nosso projeto utilizamos pixel e percentual, no caso do percentual ela se trata de uma média proporcional, como notamos ao definir que a imagem do banner deveria o ocupar 100% de largura em relação a página, se quiséssemos que o banner ocupasse metade da largura da página, bastaria escrever 50%.

Se quisermos que a fonte de titulo-principal seja o dobro da fonte padrão, independe do tamanho dessa fonte padrão, utilizamos a medida em, a média proporcional para pixels. Se quisermos duas vezes o tamanho base (15 pixels), basta escrever 2em.

Incluiremos uma margem de 0 0 1em. É sempre interessante inserir um espaçamento que seja proporcional ao tamanho da fonte, o que facilita a leitura do usuário.

O próximo passo é fazer com que o texto englobe essa imagem. Conhecemos algumas formas de tratar esses elementos, a primeira é utilizar o display do elemento para inline-block, e então todos os parágrafos seguirão o mesmo modo. Neste caso específico esta abordagem não funcionará, pois o display altera o comportamento do elemento padrão, e neste caso queremos que a imagem ocupando uma área específica e que o texto ocupe a largura inteira da página, mas que ceda espaço para a imagem.

Estudamos também o posicionamento(position) relativo, que coleta o ponto inicial e a partir disso faz deslocamentos. Aprendemos ainda sobre o posicionamento absoluto, em que podemos deslocar o ponto inicial para qualquer lugar. Neste caso, a imagem ficará por cima do texto, descolada da apresentação. Em suma, trabalhar com position não resolverá nossos problemas.

Temos, ainda, mais uma forma de posicionar e tratar elementos: o float, em tradução livre "flutuação". Quando utilizamos este recurso, o elemento "descola" da página, mas o que seria a sua sombra, continua sendo ocupada virtualmente, isto é, o texto respeita esse espaço ocupado.

Para fazermos nossa imagem "flutuar" acessaremos style.css e definiremos em utensílios qual é o lado que desejamos que a flutuação ocorra, neste caso, left.

Feita essa alteração, os parágrafos começam a se alinhar de acordo com a imagem. Contudo, o elemento está posicionado na extrema esquerda da tela, e o texto se inicia na mesma linha e o título "Benefícios" também foi desalinhado. O float é um recurso que altera completamente a estrutura da página, todos os elementos abaixo do float passam a ser afetados por ele. Podemos criar uma "barreira" que delimitará seu alcance na página.

De volta a style.css em titulo-principal, adicionaremos a propriedade clear que "limpa" o float que está posicionado à esquerda.

Ao recarregarmos o navegador, o mapa já estará visível na página. A tag <iframe> é específica para abrir "janelas" em nosso site, com proporções definidas e endereço da informação. Podemos fazer o msemo com outros serviços que apresentem essa possibilidade de compartilhamento de informações, como vídeos, fotos de Instagram, páginas de Facebook e outros. Os acessos externos que complementam a página são, geralmente, feitos pela tag <iframe>.

Incluiremos a classe itens em style.css. Feito isso, adicionaremos a propriedade line-height para definir o espaçamento entre as linhas, proporcional ao tamanho das fontes. Neste caso colocaermos o tamanho da fonte, mais 50%.

A próxima etapa é negritar o primeiro item da lista. Antigamente, a única possibilidade de realizar esse tipo de operação era criar classes específicas para uma alteração que deveria ser no CSS. Hoje em dia com as atualizações e melhorias do CSS, temos os já mencionados pseudo-elementos.

Para seleção de elementos específicos temos três opções: a primeira delas é utilizar uma classe pseudo-elemento first-child. Temos seis itens que são irmãos nos itens da lista, e as alteração serão feitas no primeiro item. Escreveremos que o peso da fonte será em negrito.

Dessa maneira, o primeiro item estará em negrito ao carregarmos o navegador. Dessa maneira criamos, via CSS, uma marcação. Além do first-child, temos a opção last-child que marcará o último item da lista. Isso é interessante, porque não precisaremos mais saber quantos itens existem na lista para passar configurações precisas, essa caracaterística é muito útil em manipulação de tabelas, por exemplo.

Por fim, temos a possibilidade de selecionar qualquer elemento da lista por via do nth-child(), que receberá o número que quisermos, como por exemplo 4, que se refere ao quarto elemento da lista. Podemos ainda utilizar valores como 2n com o nth-child(), o que quer dizer marcamos todos os elementos pares da lista, isto é, o segundo, quarto e sexto elemento da lista ficam em negrito.

Dessa maneira, mapa ficará centralizado, com as proporções que desejamos. Com essas modificações, estamos prontos para trabalhar com a propriedade background em mapa. Precisamos aqui ajustar a transição da cor branca para o cinza, então utilizaremos o linear-gradient() que receberá as cores que desejamos trabalhar. O interessante dessa propriedade é que podemos adiciconar quantas cores quisermos, que o CSS irá administrá-las automaticamente. É possível, inclusive, ajustar a inclinação das cores, por exemplo 45deg e então ficarão posicionadas em 45º.

Como estamos utilizando quatro cores no exemplo, há valores subentendidos em cada uma delas, na sequência: 0%, 25%, 50%, e 75% para a última cor da sequência. Mas podemos manipular esses valores de acordo com nosso interesse , como definir que orange, por exemplo, terá o valor de 50%.

Com os seletores avançados do CSS,podemos selecionar os filhos diretos de <main>, para tanto, utilizaremos o sinal >, e todo os outros parágrafos serão ignorados, afinal são filhos diretos da <section> e não de <main>.

main > p {

background: red;

Mas como selecionar o primeiro parágrafo que sucede uma imagem, por exemplo? Conseguimos selecionar o primeiro filho com o seletor que acabamos de conhecer, mas neste caso estamos falando do primeiro irmão que vem depois de um elemento.

Neste caso, usamos img como elemento âncora e para o primeiro irmão usamos o sinal de "+"

img + p {

balckground: blue;

Para selecionar todos os parágrafos localizados depois de uma imagem usamos o seletor ~

img ~ p {

background: yellow

}

É importante lembrar que um seletor pode sobrescrever o outro, pois todos possuem a mesma força. Os seletores avançados nos ajudam a criar estilos mais complexos e utilizar o CSS de uma maneira incrível, com uma autonomia maior do HTML.

No CSS podemos, inclusive, excluir itens específicos. Se quisermos excluir todos os parágrafos que não compõe missao, escreveremos:

.principal p:not(#missao) {

background: orange;

}

Para que a imagem sempre ocupe a medida correta em outros dispositivos, utilizamos a propriedade calc. O cálculo que desejamos realizar é escrito entre parênteses, em que inserimos o primeiro valor, o tipo de operção e o resultado esperado.

.utensilios {

width: calc(40% - 26px);

float: left;

Margin: 0 20px 20px

Dessa forma, foi calculado precisamente 350px de largura para este elemento. É possível encadear quantas operações quisermos, no mesmo modelo de sintaxe.

Utilizaremos a imagem da sessão "Benefícios" para compreendermos o funcionamento da ferramenta de opacidade no HTML. Em index.css , utilizaremos a propriedade opacity que pode ter valores entre 0 a 1, sendo que 0 o elemento estará totalmente oculto. Se quisermos exibir a imagem com opacidade de 30%, ajustaremos o valor da propriedade para 0.3.

Esse recurso é interessante quando montamos composições visuais com elementos sobrepostos, ou para sugerir interações com o mouse ao utilizarmos o hover. Quando o mouse estiver posicionado sobre a imagem teremos 1 de opacidade.

Com CSS 3, podemos utilizar mais uma camada - a de opacidade, chamada alfa - nas cores em RGB. Para tanto, utilizamos o rgba e então definir os valores que quisermos. Exemplo = (rgba(0,0,0,0.3)).

Continuaremos a usar a imagem da sessão de "Benefícios". A sombra é resultado de um efeito de "luz" que lançaremos sobre o elemento. O nome da propriedade que utilizaremos para gerar esse efeito é box-shadow, que possui a propriedade do eixo X, T e uma cor. Queremos deslocar 10px no eixo X e Y, a cor que utilizaremos será preto.

.imagem-beneficios {

width: 60%

opacity: 1;

transition: 400ms;

box-shadow: 10px 10px #000000;

}

Podemos melhorar a qualidade estética dessa sombra ao adicionarmos uma terceira propriedade chamada blur, em que podemos aplicar um nível de desfoque específico, no caso, inseriremos um valor de 5px. Quanto maior a quantidade de pixels que inserirmos, mais claro sera o efeito de desfoque. Temos ainda uma quarta propriedade que configura a intensidade da borda a partir do tamanho do elemento, isto é, o tamanho da sombra projetada. Neste caso, inseriremos 20px: No navegador, a sombra sugirá expandida. Podemos adicionar várias sombras em um mesmo elemento, basta que o conteúdo esteja separado por uma vírgula. Essa nova sombra terá valores negativos e terá a cor amarela.

Como fazer uma sombra interna em um elemento?

box-shadow: inset 0 0 5px black;

Primeriamente, precisamos compreender quantos pixels existem na tela para trabalhamos em uma melhor exibição. Atualmente, nossa tela se adapte ao tamanho disponibilizado pelo navegador, sendo o mínimo 940px. No caso de celulares o panorama é diferente: mesmo que a resolução do dispositivo seja de 2000px de largura, a quantidade de pixels existentes na tela sempre será diferente. Esses valores são chamados de DPI, isto é, a densidade de pixels por polegada.

Geralmente, os celulares possuem 320px de largura de tela,alguns maiores oscilam entre 350px e 480px. Portanto se o mínimo é 940px, teremos um valor discrepante. Uma informação crucial é saber quantos pixels possui a tela do usuário.

Trabalharemos com uma nova tag <meta>, mas dessa vez com propriedade e valor name="viewport" embutidas. Queremos, ainda, saber a largura do dispositivo, e para isso usaremos a propriedade content com o valor width=device-width.