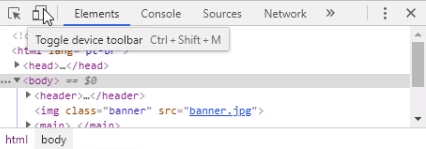
TAG META VIEWPORT – ADAPTAÇÃO DE CONTEUDO WEB PARA OS CELULARES

Nesta aula, discutiremos sobre as versões mobile do nosso site, isto é, as adaptações necessárias para que o site seja visualizado corretamente em dispositivos móveis. Hoje em dia, a maior parte do fluxo de usuários na internet é proveniente de celulares, portanto um site não responsivo a tal necessidade é um problema.

Para testarmos a usabilidade do site, abriremos a ferramenta do desenvolvedor por meio do atalho "Ctrl + Shift + I" . No painel de ferramentas, teremos um ícone que representa tablets e celulares, a "Toggle device toolbar" , que também pode ser acessada pelo atalho "Ctrl + Shift + M".



No momento em que habilitamos essa opção, nosso site passa a se comportar como se estivesse sendo visualizado via celular: o conteúdo é compartimentando em 940px, o mínimo para esse tipo de plataforma. Atualmente, temos alguns problemas nesse tipo de visualização: a logo marcar está pequena,o menu está pouco legível assim como o conteúdo da lista de benefícios.



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.

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width">

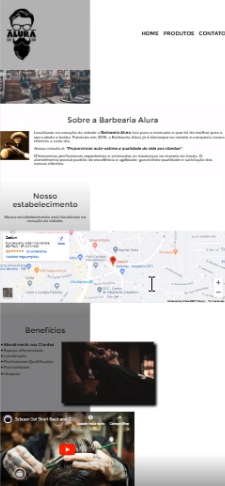
<title>Barbearia Alura</title>

<link rel="stylesheet" href="style-home.css">

<link href="https://fonts.googleapis.com/css?family=Montserrat&display=swap" rel="stylesheet">

</head>COPIAR CÓDIGO

Ao reabrirmos o navegador, veremos que o site se comporta de maneira diferente. A forma como os elementos etão distribuídos em relação ao fundo, o conteúdo é de 940px, mas a visualização mobile é de 375px, que corresponde a um IphoneX. Na pare superior da tela ainda podemos rotacionar a tela para horizontal e ver como a página se comporta.



Precisaremos, então, adaptar o CSS para que ele funcione nessa tela. Precisamos entregar para telas até 450px um estilo diferente, e faremos isso por meio de *media queries*. Perguntaremos ao navegador qual é o tamanho da tela em questão e, se for aquele que desejamos, entregaremos o estilo correto.

Para realizar isso, utilizaremos o @media, com o valor do tipo de tela screen e a pesquisa. Todas as telas que tenham até 450, terão esse estilo diferente, reescrito. Para exempliciar, colocaremos no background um estilo da cor vermelha.

@media screen and (max-width: 480px) {

body {

background: red;

}

}COPIAR CÓDIGO

Ao regarregarmos a página veremos a modificação aplicada.



Sabemos como o CSS pode responder a situações mobile, nas próximas aulas adaptaremos nosso conteúdo para entregar um site responsivo.

De maneira bem objetiva e dinâmica, adaptaremos nosso site para que ele se torne responsivo, ou seja, que se comporte de maneira adequada no desktop e em dispositivos móveis.

Primeiramente nos atentaremos para o cabeçalho, que possui 940px de altura. Se inserirmos uma altura automática, o posicionamento dos elementos que compõe a sessão se deforma, portanto essa não é uma boa saída para solucionar a questão. Em style.css mudaremos a altura da caixa para auto. Quando elaboramos layouts responsivos, precisamos trabalhar exclusivamente no elemento que será modificado.

Faremos o mesmo procedimento para principal, conteudo-beneficios, mapa-conteudo e video.

@media screen and (max-width: 480px) {

.caixa, .principal, .conteudo-beneficios, .mapa-conteudo, .video {

width: auto;

}

}COPIAR CÓDIGO

Precisaremos ainda fazer uma pequena alteração na classe video, e anterar o width para 100%, assim evitamos uma deformação no layout.

Feito isso, faremos as alterações necessárias no cabeçalho da página, e a primeira adaptação que faremos é centralizar o <h1> Em nosso CSS, escreveremos:

h1 {

text-align: center;

}COPIAR CÓDIGO

Feito isso, queremos que o menu - que até o presente momento está em posição absoluta - volte a ser estático. Então em nosso CSS escreveremos:

nav {

position:static;

}COPIAR CÓDIGO

Por fim, faremos alterações na lista de benefícios, de forma que a imagem e texto fiquem melhor posicionados.

.lista-beneficios, imagem-beneficios {

width: 100%

}COPIAR CÓDIGO

Há ainda melhorias possíveis, como alteração de espaçamentos, um respiro do texto em relação à borda da página, e assim por diante. Contudo, a página está funcional para mobile!

**Vw (viewport width)**

Essa medida faz parte das medidas mais atuais e do futuro do CSS. Viewport units.

Como escrito no título, **vw** significa viewport width, mas o que é viewport?

Viewport nada mais é que a área visível de uma página web para o seu usuário, essa viewport pode variar de acordo com o dispositivo, sendo menor em celulares e maior em desktops.

Antigamente, quando não existiam tablets e celulares capazes de acessar sites, todas as web pages eram pensadas para a tela de um computador, com tamanho fixo e design estático. Com a chegada desses dispositivos móveis, essas páginas eram grandes demais para serem exibidas nesses aparelhos, o que tornava muito difícil a navegação.

A primeira solução partiu dos browsers desses dispositivos, eles adotavam um comportamento de retirar o zoom de forma que o site inteiro coubesse na tela do aparelho,definitivamente não era o ideal, mas uma solução rápida. No HTML5, foi introduzido uma maneira para que os desenvolvedores conseguissem alterar a viewport através da **tag**, corrigindo esse problema de usabilidade relacionado aos dispositivos móveis, mas isso é assunto para outra postagem!

Voltando para o nosso querido **vw**, essa unidade se relaciona diretamente com a largura da viewport, onde **1vw** representa **1%** do tamanho da largura dessa área visível. A diferença entre **vw** e a **%** é bem semelhante a diferença entre **em** e **rem**, onde a **%** é relativa ao contexto local do elemento e o **vw** é relativo ao tamanho total da largura da viewport do usuário.

**Vh (viewport height)**

Essa unidade funciona da mesma forma que o **vw**, porém dessa vez, a referência será a altura e não a largura. Existem diversos exemplos práticos e interessantes de uso dessas duas unidades. Você pode ver alguns usos nesse [link](https://css-tricks.com/fun-viewport-units/), provavelmente mais para frente postarei alguns exemplos bacanas. Me cobrem!

**Vmin (viewport minimun)**

Essa unidade também se relaciona com as dimensões da viewport, mas com um porém. Anteriormente quando vimos **vh** e **vw** precisávamos escolher se gostaríamos de nos basear na altura (vh) ou na largura (vw) da viewport.

Diferentemente das anteriores, o **vmin** utilizará como base a menor dimensão da viewport (altura x largura), vamos ao exemplo.

Imagine que estamos trabalhando com uma viewport de 1600px de altura e 900px de largura. Nesse caso, **1vmin** terá o valor de **9px** (1% da menor dimensão!), caso tenhamos **100vmin**, esse será igual a **900px**! Interessante né?

No caso acima, a menor dimensão foi a da largura, porém se tivéssemos 300px para altura e 1400px para largura, nosso valor de referência seria o 300px! Sempre a **menor** dimensão!

**Vmax (viewport maximum)**

Seguindo a mesma base lógica da unidade anterior, o **vmax** terá como valor de referência a maior dimensão da viewport. Ou seja, utilizando o mesmo exemplo, se tivermos 1600px de altura e 900px de largura, **1vmax** será equivalente a **16px**!

No segundo exemplo ocorrerá a mesma inversão, tendo 300px para altura e 1400px para largura, **1vmax** será equivalente a **14px**. Dessa vez sempre será a **maior** dimensão!

**Alerta!!**

Na versão 9.0 do IE, o **vmin** foi implementado com um nome diferenciado (vm) Algumas versões do IE ainda não oferecem suporte para o **vmax**, nas novas versões isso foi corrigido.

Algumas plataformas como o windows tendem a ser inconsistentes na maneira de como eles contam o tamanho da barra de scroll dos browsers. (diminui o viewport ou não) Sempre verifique o suporte dessas medidas, tenha sempre [esse link](https://caniuse.com/#search=vw) no cinto de utilidades.