Nesta aula, aprendemos:

* A definir a estrutura básica do HTML
  + Com a *tag* DOCTYPE, definimos qual versão do HTML estamos utilizando
  + A *tag* **<html>**, que marca o conteúdo a ser renderizado no navegador
    - Dentro desta *tag*, podemos definir a linguagem da página, através da propriedade **lang**
* Como passar as informações do *encoding* da nossa página para o navegador, através da *tag* **<meta>** e da propriedade **charset**
* Como definir o título de uma página, através da *tag* **<title>**
* Como separar as informações que estão sendo passadas para o navegador, utilizando a *tag* **<head>**
* Como separar o conteúdo da página, utilizando a *tag* **<body>**

Existem três formas de se utilizar o CSS:

A primeira utilizada no curso é o “CSS in line”, onde os parâmetros de estilização são colocados na linha dentro do próprio documento HTML.

Nesta aula, aprendemos:

* A mexer na apresentação dos textos
  + No alinhamento deles (text-align)
  + No tamanho da fonte (font-size)
  + Na cor de fundo (background)
  + Na cor do texto (color)
* CSS *inline*
  + Na linha onde temos a nossa *tag*, adicionamos a propriedade do CSS
* A tag **<style>**
  + Dentro da *tag*, podemos colocar marcações de CSS referentes aos elementos que temos no nosso HTML
* A apresentação do CSS com um arquivo externo
* Como funciona o estilo em cascata do CSS
* Como importar um arquivo externo de CSS dentro da nossa página HTML
* Como representar cores no CSS
  + Através do nome da cor
  + Através do seu hexadecimal
  + Através do seu RGB

Nesta aula, começamos a mexer na apresentação dos textos, no alinhamento deles e no tamanho da fonte. Na próxima, vamos ver como fazer isso em grandes quantidades de texto e de forma muito mais organizada. Te espero lá.

Nesta aula, aprendemos:

* Como reestruturar o nosso código, removendo os CSS *inline* e colocando-os no arquivo CSS externo
* Como criar um identificador para marcar especificamente um elemento
  + Como fazer referência a esse identificador no CSS
* Como adicionar uma imagem à nossa página
* Como ajustar a altura do elemento, através da propriedade height
* Como ajustar a largura do elemento, através da propriedade width
* Como ajustar o espaçamento interno do elemento, através da propriedade padding
* Como ajustar o espaçamento externo do elemento, através da propriedade margin
* Como se comporta um time de front-end hoje em dia

A tag para a lista é a ul e cada item da lista precisa ser marcado com um li.

No caso da tag li, passa a ocupar toda a largura da página naquela linha e quebrar a linha depois dela, ou seja, o próximo item começa na linha debaixo. O comportamento de uma tag ocupar toda a largura de uma página é chamado “BLOCK”. Ele bloqueia esse conteúdo dessa linha.

Uma imagem não bloqueia toda a largura da página e esse comportamento é chamado de “INLINE”.

A diferença entre eles é que mesmo que eu diminua o tamanho, diminua a largura de um elemento block, ele vai sempre ocupar aquela linha, mesmo preenchendo só metade.

Um elemento inline não me deixa alterar, por exemplo, o espaçamento externo e interno dele.

 Para isso, existe uma terceira característica, quando o elemento possui as duas condições. Ele é inline e block ao mesmo tempo. Ou seja, ele bloqueia uma largura, mas essa largura é fixa. Sou eu que dou o tamanho. E ele me deixa também mexer na largura e nos espaçamentos interno e externo.

A tag ol serve para listas com dados ordenados, como numa receita de bolo. Cada passo precisa ser executado numa ordem pré-definida e cada item da lista precisa ser marcado com um li.

A tag div é a tag específica para dividir o conteúdo. Ela não tem outra função além dessa.

A class serve para replicarmos a aplicação do nosso CSS, sem precisar copiar e colá-lo em vários elementos. Aplicamos a classe para isso.

Nesta aula, aprendemos:

* A trabalhar com listas não-ordenadas e listas ordenadas
  + Para cada um dos itens da lista, utilizamos a *tag* **<li>**
* O conceito das **classes** no CSS
  + Elas servem para marcar itens, que são repetíveis
* Como referenciar uma classe no CSS
* Divisões de conteúdo, utilizando a *tag* **<div>**
* Os comportamentos ***inline*** e ***block***

A tag header é uma tag de conteúdo do HTML, e não uma tag de estrutura. Não confunda head com header

A tag <a> serve justamente como uma âncora, um link, que vai nos levar para outro ponto na internet.

A propriedade text-transform é a responsável pela transformação do texto para maiúsculo, quando usamos o valor uppercase

Nesta aula, aprendemos:

* A criar links para outras páginas, sejam elas do nosso projeto ou páginas externas
* Um reforço aos estilos **inline** e **block**
* Como transformar o texto para ter todas as letras maiúsculas
* Como deixar o texto em negrito com CSS
* Como remover a decoração do texto

O navegador cria o estilo para todas as tags que ele acha adequadas, então precisamos removê-lo, para que o estilo que criamos não seja influenciado pelo estilo que o navegador adiciona.

 Com o position: absolute, eu consigo posicionar meu elemento em qualquer lugar da página.

Usamos a tag ***margin*** ao delegarmos para o navegador fazer a conta de calcular a largura que resta, dividir por dois, e adicionar metade em cada um dos lados.

Nesta aula, aprendemos:

* Como remover os estilos que o navegador cria automaticamente
* Como funciona os posicionamentos ***static***, ***relative*** e ***absolute*** dos elementos
* Como posicionar o cabeçalho da nossa página

Por que devemos usar as tags semânticas, como a <main>?

As tags semânticas fazem com que o navegador entenda melhor o nosso conteúdo, e com isso carregue melhor a nossa página.

Quando tenho uma lista com itens complexos, qual a formatação correta?

<li>

<h2>Cabelo</h2>

<p>R$ 40,00</p>

</li>

 Um item de lista, em uma lista complexa, pode ter qualquer conteúdo.

Como se comporta um elemento que tem o seu display com o valor inline-block?

O tamanho pode ser ajustado, tanto na largura, quanto na altura.

Qual o comportamento do elemento quando adiciono espaçamento interno, o padding?

O elemento aumenta o seu tamanho. Quando declaro que existe um espaçamento interno, estou aumentando o conteúdo.

Nesta aula, aprendemos:

* A *tag* main, para o conteúdo principal da nossa página
* A criar listas complexas, com títulos, imagens e parágrafos
* A utilizar o inline-block
* A praticar e estilizar o conteúdo principal da nossa página

Quando adiciono a seguinte declaração de borda:

border: 3px solid #000000;

Qual o comportamento esperado?

Na estrutura de "caixas" do CSS, a borda fica posicionada entre o padding e a margin.

Como usar o CSS para fazer os desenhos de cantos arredondados das bordas?

Com a propriedade border-radius, podemos desenhar o canto arredondado dos elementos.

Nesta aula, aprendemos:

* Através do CSS, aplicar bordas nos elementos.
* Os diferentes tipos de bordas.
* A deixar a borda arredondada.

Quando podemos utilizar o comportamento do hover em um elemento?

 No hover, mapeamos o comportamento do mouse por cima do elemento, e podemos usar isso para destacar visualmente o elemento em questão.

Nesta aula, aprendemos:

* Algumas **pseudo-classes** CSS
  + **hover**, quando o usuário passa o cursor sobre o elemento
  + **active**, quando um elemento está sendo ativado pelo usuário
* A mudar a cor do texto e/ou da borda de um elemento, quando o usuário passar o cursor sobre o mesmo
* A mudar a cor da borda de um elemento, quando o mesmo estiver sendo ativado pelo usuário

Como adicionamos uma imagem de fundo em um elemento?

No CSS, ao referenciarmos o endereço de uma imagem, usamos o valor url.

Nesta aula, aprendemos:

* A *tag* footer, para o rodapé da nossa página
* Que, com CSS, podemos colocar uma imagem de fundo em um elemento
  + Quando colocamos uma imagem de fundo em um elemento, o CSS, por padrão, copia e cola a imagem diversas vezes até ocupar todo o espaço do elemento
* A tabela **Unicode**

Nesta aula, aprendemos:

* A criar um formulário HTML
  + A *tag* que o representa é a <form>
* A *tag* <input>, para a entrada de dados do usuário
* A criar uma *etiqueta* para o input, com a *tag* <label>
* A conectar um input com o seu label
  + Colocamos um id para o input e associamos esse id ao atributo for do label
* Alguns tipos de input, como text e submit
* Que label possui o display inline e o input possui display inline-block
* A estilizar o nosso formulário

Qual a propriedade de um input do tipo radio que preciso criar para que eles façam parte do mesmo grupo?

A propriedade name só pode ser "preenchida" uma única vez, por isso que, quando eu seleciono um dos itens, ele desmarca o outro, mantendo somente um selecionado.

Qual a melhor abordagem quando temos uma configuração de CSS aplicada para um item e queremos replicar ela para um item parecido?

Essa é uma forma simples de fazermos a mesma configuração de CSS funcionar para os dois itens, sendo eles seletores dos elementos, de ids ou de classes.

Quando temos um estilo que pode ser repetido, uma excelente estratégia é extrair isso para uma classe e usar a classe para aplicar esses estilos nos nossos elementos.

Quando temos um estilo que pode ser repetido, uma excelente estratégia é extrair isso para uma classe e usar a classe para aplicar esses estilos nos nossos elementos.

Considerando o código abaixo:

<p class="paragrafo">

p {

color: blue;

}

p.paragrafo {

color: red;

}

.paragrafo {

color: purple;

}

Qual será a cor final do meu parágrafo?

O seletor p.paragrafo é o mais forte, e a cor aplicada é a vermelha (red).

No último vídeo, foi vista a tag **<option>**. Qual a sua utilidade?

 A estrutura da tag <select> é composta de um ou mais <option>.

Nesta aula, vimos:

* O textarea, para entradas de texto de mais de uma linha
* O input do tipo radio
* Como agrupar vários input do tipo radio, impedindo que mais de um input seja selecionado
* O input do tipo checkbox
* Que podemos criar um input dentro de um label, assim associando-os
* Mais estilizações para a nossa página
* Como funciona a hierarquia no CSS
* O select, que é seletor, um campo de seleção de um item, e o option, que representa cada opção do seletor

Por quais motivos é importante usar os tipos corretos de <input> para os usuários que acessam a página via celular?

É extremamente importante facilitarmos a vida dos nossos usuários sempre.

É muito melhor termos dados corretos quando temos o nosso formulário preenchido.

Como transformamos um campo comum em um de preenchimento obrigatório?

Quando adicionamos a propriedade required, o navegador nos ajuda a validar se aquele campo está preenchido.

Qual tag usamos para marcar um título de um grupo de campos do formulário?

É a tag <legend> que usamos para um título de um grupo de campos em qualquer formulário. E esse título vai dentro da divisão <fieldset> .

Nesta aula, aprendemos:

* Alguns tipos de *inputs* para celular: email, tel, number, password, date, datetime, month e search
* Como não permitir que um campo não seja preenchido, através do atributo required
* Como exibir uma sugestão de preenchimento para os campos, através do atributo placeholder
* Como deixar uma opção marcada por padrão nos nossos input radio e checkbox, através do atributo checked
* Como estruturar melhor o nosso código com fieldset e legend
* Como adicionar uma alternativa à imagem, descrevendo-a, com o atributo alt

Como configurar a transição visual de um elemento, que dure 3 segundos?

Utilizamos a propriedade transition para prover uma transição visual de um elemento.

Qual propriedade do CSS usamos para aumentar um elemento proporcionalmente?

Utilizamos a propriedade transform: scale() para aumentar um elemento proporcionalmente.

Nesta aula, aprendemos:

* Como estilizar o botão de envio de formulário
* A realizar transições nos nossos elementos, com a propriedade CSS **transition**
* A modificar o estilo do ponteiro do mouse, quando passar por cima de determinado elemento, através da propriedade CSS **cursor**
* A realizar transformações nos nossos elementos, como aumentar proporcionalmente a escala de determinado elemento ou rotacioná-lo, através da propriedade CSS **transform**

Qual tag uso para criar linhas em uma tabela?

 Utilizamos a tag <tr> para marcar uma linha de uma tabela.

Como dividir uma tabela de forma mais semântica?

As tags <thead>, <tbody> e <tfoot> ajudam a deixar o conteúdo da tabela mais bem dividido e mais semântico.

As tabelas também nos oferecem a possibilidade de juntar células e montar um visual diferente. Por exemplo, quando uma linha, que deveria ter 5 células, passa a mostrar só "uma célula".

Esse efeito é conseguido através da propriedade colspan=X, onde X é o número de células que você quer agrupar.

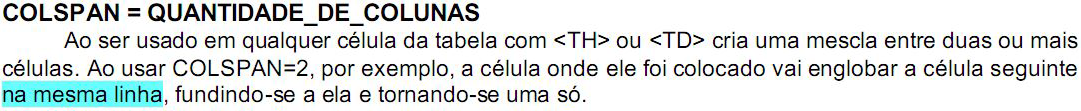
Portanto, em uma tabela de 5 colunas, para ter uma célula única na linha, usamos um código assim:

<tr>

<td colspan="5">Rio de Janeiro</td>

</tr>

Para entendermos o que aconteceu, precisamos considerar o seguinte:



\*Extraído da apostila "HTML HiperText Markup Language - Prof. Fabio Miyasaki

Como a tabela tem apenas 2 coluna no cabeçalho (thead), colspan não foi executado aí nesta área da tabela.



Já no corpo da tabela (tbody), na primeira linha em que o colspan aparece, o programa não encontrou a quinta célula na mesma linha, então foi passando para as linhas seguintes até chegar na primeira da última linha. Aí, ele uniu esta com a célula seguinte, ocupando uma só coluna. Ou seja, só o último colspan foi executado, com as células que haviam.

Não foi possível a execução de todos os outros colspan ou por não haver a quinta célula ou por não existir uma célula seguinte a ela para ser unificada.

Segue um outro exemplo de código utilizando colspan:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

</head>

<table border="1" width="500">

<thead>

<tr>

<th>Seg</th>

<th>Ter</th>

<th>Qua</th>

<th>Qui</th>

<th>Sex</th>

<th>Sab</th>

<th>Dom</th>

</tr>

</thead>

<tbody>

<tr>

<td colspan="5">aberto</td>

<td colspan="2">fechado</td>

</tr>

</tbody>

</table>

</html>

Nesta aula, aprendemos:

* A criar uma tabela HTML
  + A *tag* table, que representa a tabela
  + A *tag* tr, que representa a linha da tabela
  + A *tag* td, que representa a célula da tabela
  + A *tag* thead, que representa o cabeçalho da tabela
  + A *tag* tbody, que representa o corpo da tabela
  + A *tag* th, que representa a célula do cabeçalho da tabela
  + A *tag* tfoot, que representa o rodapé da tabela
* A estilizar a tabela

Qual a diferença entre usar uma <div> ou uma <section>?

Para um bloco onde o conteúdo tenha o mesmo significado, o mesmo sentido, usamos uma <section>.

Ao criar um estilo qualquer, qual o melhor padrão para marcar o nosso conteúdo, ao criar uma classe?

 Um balanço é a melhor alternativa. Nem tão específica, para poder repetir as classes, e nem tão genérica, para poder não precisar combinar classes.

Para que serve a propriedade **float**?

 Tanto o float:left quanto o float: right servem para que o elemento se destaque na tela, deixe de ocupar o espaço em que estava, para que os outros elementos possam se posicionar ao redor dele

Nesta aula, aprendemos:

* A ajustar a página principal para utilizar os mesmos padrões da página de produtos
* Medidas proporcionais com CSS
* Como funciona a *flutuação* dos elementos e como modificá-la, com a propriedade float do CSS
* Como *limpar* o float, com a propriedade clear do CSS

Analise os seguintes benefícios de usar fontes externas:

**1)** Padronizar o nosso conteúdo em todos os navegadores

**2)** Deixar o site mais bonito, com visual mais exclusivo.

**3)** Termos mais alternativas de fonte

Quais das afirmativas acima realmente são benefícios de usar fontes externas?

Usar fontes externas é uma alternativa muito funcional para termos mais opções, que deixam o nosso site mais bonito e exclusivo, e também padronizado em todos os navegadores.

Nesta aula, aprendemos:

* A utilizar fontes externas nas nossas páginas
* Como incorporar um mapa à nossa página
* Como incorporar um vídeo à nossa página

No CSS temos algo chamado **pseudo-elementos**. Conhecemos alguns deles, como :hover, :active, :visited, :required. Tais recursos são utilizados para marcar melhor nossos elementos e gerar um comportamento mais interessante em nosso site.

 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.

.mapa {

padding:3em 0;

background: linear-gradient(45deg, orange, blue, red, black);

}

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%.

.mapa {

padding:3em 0;

background: linear-gradient(45deg, orange 50%, blue, red, black);

}

Além do degradê linear(linear-gradient) temos ainda a opção radial(radial-gradient), que organiza a transição de cores de maneira circular. Nosso site terá uma gradação de cinza linear:

.mapa {

padding:3em 0;

background: linear-gradient(#FEFEFE, #888888);

}

Como fazer um degradê de cinco cores em um elemento?

Usamos dentro do mesmo parênteses, separando as cores por vírgulas.

background: linear-gradient(gray, yellow, red, orange, blue)

Como criar um elemento no HTML antes do meu item, através do CSS?

Quando queremos criar um elemento na página, via CSS, antes do elemento do HTML, usamos a propriedade :before.

Nesta aula, aprendemos:

* A melhorar mais ainda a semântica da página principal, com novas divisões, classes, etc
* Novas pseudo-classes
* Como aplicar um *background* gradiente na página
* Pseudo-elementos

Usando essa estrutura:

<h2>

<section>

<h2>

<p>

</p>

<h2>

Como fazemos para selecionar o último <h2>?

Section > p + h2

O último h2 é filho direto da section e irmão do p.

“Em nosso CSS, verificaremos que o tamanho da imagem é de 120px, mas como podemos fazer com que esse tamanho seja proporcional? Basta mudar para percentual, isto é, 20% de largura tendo a tela como referência.

Para que a imagem sempre ocupe a medida correta em outros dispositivos, utilizamos a propriedade calc. O calculo 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.

.utensilios {

width: calc(40% - (26px \* 4);

float: left;

Margin: 0 20px 20px

}

A propriedade calc nos dá o poder de fazer com o que cremos valores proporcionais específicos.

Como usar as medidas proporcionais para deixar um elemento de 100% de largura com o equivalente a um terço do elemento pai, menos 10px?

*width: cal( (100% / 3) – 10px )*

Essa é uma forma mais complexa, funciona, porém não precisa dessa primeira conta.

*width: calc( 33% - 10px )*

Essa é a forma mais direta.

Nesta aula, aprendemos:

* Seletores avançados CSS
  + Seletor >, para acessar os filhos de determinado elemento. Por exemplo, para acessar todos os p dentro de main:
* main > p {
* }

Seletor +, para acessar o primeiro irmão de determinado elemento. Por exemplo, para acessar o primeiro p após um img:

img + p {

}

Seletor ~, para acessar todos os irmãos de determinado elemento. Por exemplo, para acessar todos os p após um img:

img ~ p {

}

Seletor not, para acessar os elementos, exceto algum. Por exemplo, para acessar todos os p dentro de main, exceto o p que tem id missao:

main p:not(#missao) {

}

* Como fazer contas com CSS, com a propriedade calc

Quais tipos de elementos podem receber uma camada de opacidade?

 Todos os elementos e todas as cores podem ter uma camada de opacidade.

Como fazer uma sombra interna em um elemento?

*box-shadow: inset 0 0 5px black;*

Sombra interna, preta, com 5px de espalhamento.

<https://developer.mozilla.org/pt-BR/docs/Web/CSS/box-shadow> (documentação)

Nesta aula, aprendemos:

* Como manipular a opacidade dos elementos, com a propriedade CSS opacity
* Como manipular a opacidade das cores
* Como adicionar um sombreamento em volta dos elementos, com a propriedade CSS box-shadow
* Como adicionar um sombreamento em textos, com a propriedade CSS text-shadow

Como marcar um elemento para que ele seja apresentado de outra forma, em uma tela de até 480px?

É dentro dessa media query que criamos um estilo visual que compreenda telas de até 480px.

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

    body{

        background: red;

    }

}

Qual a principal tática para não perder muito tempo criando layouts responsivos?

Usar medidas proporcionais para tudo. É hoje a melhor forma de criar qualquer conteúdo.

Criar um layout fluido que se adapte sempre. É uma forma perfeita de evitar.

* + 1. Na página **index.html**, dentro do head, adicione a tag meta viewport, logo abaixo da tag meta charset:

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

* + 1. Altere o tamanho do vídeo, para não ser mais fixo e sim ocupar toda a largura da tela, com width="100%":

<iframe width="100%" height="315" src="https://www.youtube.com/embed/wcVVXUV0YUY" frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe>

* + 1. No arquivo **style.css**, adicione um estilo diferente para dispositivos de até 480 pixels de largura:
* @media screen and (max-width: 480px) {
* .caixa, .principal, .conteudo-beneficios, .mapa-conteudo, .video {
* width: auto;
* }
* h1 {
* text-align: center;
* }
* nav {
* position: static;
* }
* .lista-beneficios, .imagem-beneficios {
* width: 100%;
* }
* }

Nesta aula, aprendemos:

* **Design responsivo**: como ajustar o estilo da nossa página de acordo com o tamanho da tela do dispositivo que a acesse
  + Meta *tag* de ***Viewport***
  + *Media Queries*