

Estilos com JavaScript

Transcrição

Faremos ajustes para deixar os nossos erros mais identificáveis. Por enquanto, temos somente cinco pacientes, e para identificar um deles cujo valor de peso ou altura seja inválido, vamos procurar pela mensagem exibida na última coluna. Imagine se a tabela tiver 50 ou 100 pacientes cadastrados? Será preciso ficar *scrollando* a página e procurar pelas mensagens!

Por isso, deixaremos uma marcação visual para a nutricionista, indicando as linhas com erros. Faremos com que a linha com peso ou altura inválido exiba um fundo vermelho.

Vamos acessar o arquivo `principal.js` e analisar o trecho em que realizamos as validações:

```
var pesoEhValido = true;
var alturaEhValida = true;

if (peso <= 0 || peso >= 1000) {
  console.log("Peso inválido!");
  pesoEhValido = false;
  tdImc.textContent = "Peso inválido";
}

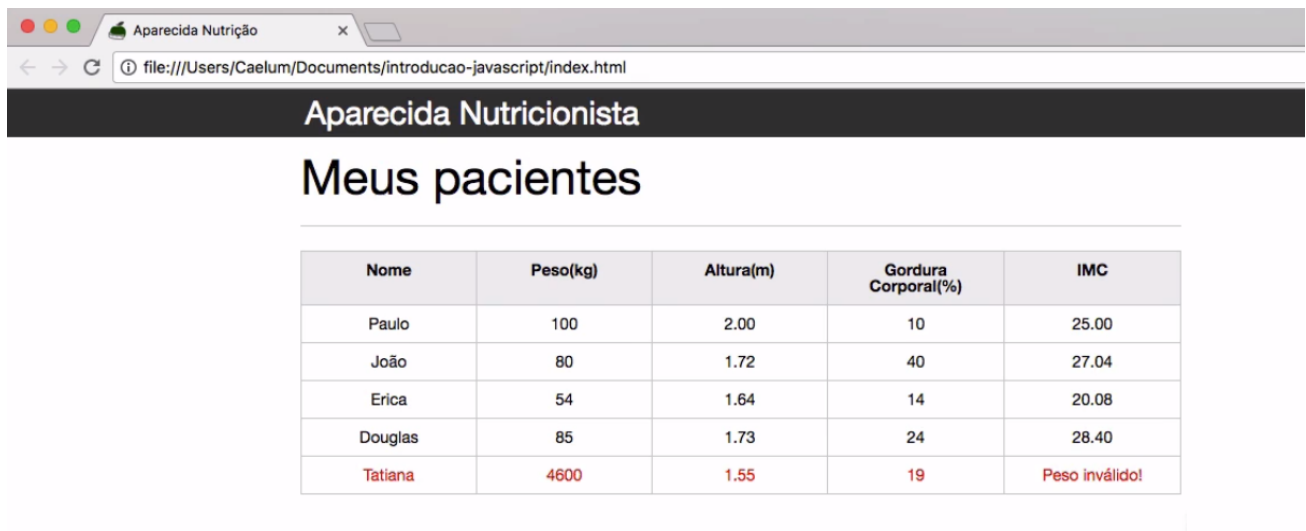
if (altura <= 0 || altura >= 3.00) {
  console.log("Altura inválida!");
  alturaEhValida = false;
  tdImc.textContent = "Altura inválida";
}
```

Para alterarmos a cor do fundo, selecionaremos a linha (cada tag `tr`) - o que já fizemos anteriormente e será representada pela variável `paciente`. Nosso objetivo é modificar o **estilo** do elemento, portanto utilizaremos a propriedade `style`, com os estilos aplicados. Ou seja, vamos alterar o CSS aplicado. Se adicionarmos `style.color`, poderíamos alterar a cor da fonte na linha.

```
var pesoEhValido = true;
var alturaEhValida = true;

if (peso <= 0 || peso >= 1000) {
  console.log("Peso inválido!");
  pesoEhValido = false;
  tdImc.textContent = "Peso inválido";
  paciente.style.color = "red";
}
```

Quando atualizarmos a página veremos que a cor da fonte foi alterada.

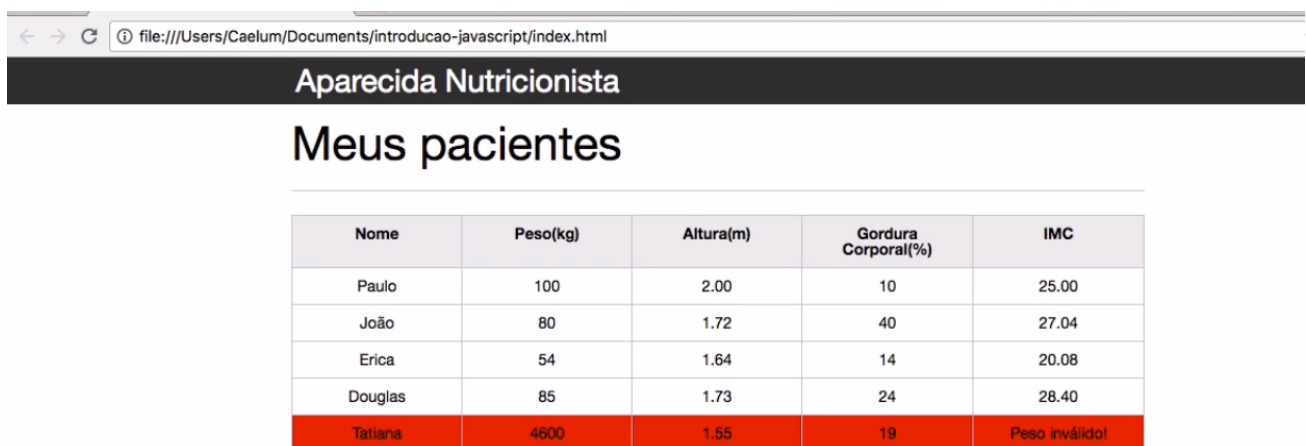


Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	100	2.00	10	25.00
João	80	1.72	40	27.04
Erica	54	1.64	14	20.08
Douglas	85	1.73	24	28.40
Tatiana	4600	1.55	19	Peso inválido!

Assim, por conseguirmos alterar a cor da fonte usando a propriedade `color`, poderemos modificar o fundo da linha ao adicionarmos a propriedade `backgroundColor`. Porém, se tentarmos escrever da mesma forma como é feito no CSS (`background-color`), nosso código não funcionará. Quando queremos modificar um estilo com duas palavras, o hífen (-) não é aceito pela linguagem JavaScript. Teremos que utilizar o padrão ***Camel Case*** - no qual as palavras compostas ou frases são iniciadas com maiúsculas e unidas sem espaços.

```
if (peso <= 0 || peso >= 1000) {  
  console.log("Peso inválido!");  
  pesoEhValido = false;  
  tdImc.textContent = "Peso inválido";  
  paciente.style.backgroundColor = "red";  
}
```

Sempre que houver um erro, ele ficará em evidência na tabela.



Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	100	2.00	10	25.00
João	80	1.72	40	27.04
Erica	54	1.64	14	20.08
Douglas	85	1.73	24	28.40
Tatiana	4600	1.55	19	Peso inválido!

Vamos alterar a cor para um tom mais claro: `lightcoral`.

```
if (altura <= 0 || altura >= 3.00) {  
  console.log("Altura inválida!");  
  alturaEhValida = false;  
  paciente.style.backgroundColor = "lightcoral";  
}
```

file:///Users/Caelum/Documents/introducao-javascript/index.html

Aparecida Nutricionista				
Meus pacientes				
Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	100	2.00	10	25.00
João	80	1.72	40	27.04
Erica	54	1.64	14	20.08
Douglas	85	1.73	24	28.40
Tatiana	4600	1.55	19	Peso inválido!

Em seguida, vamos usar a propriedade `style` para sinalizar o problema na altura.

```
if (altura <= 0 || altura >= 3.00) {  
  console.log("Altura inválida!");  
  alturaEhValida = false;  
  paciente.style.backgroundColor = "lightcoral";  
}
```

file:///Users/Caelum/Documents/introducao-javascript/index.html

Aparecida Nutricionista				
Meus pacientes				
Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	100	5.00	10	Altura inválida!
João	80	1.72	40	27.04
Erica	54	1.64	14	20.08
Douglas	85	1.73	24	28.40
Tatiana	4600	1.55	19	Peso inválido!

Sempre que ocorre um erro, ele ficará fácil de ser identificado na tabela. Conseguiremos fazer a linha ficar com o fundo vermelho quando algum dado for inválido.

No entanto, imagine a situação em que o designer decide que a cor de fundo não será mais vermelha, e sim laranja. Devemos nos lembrar de modificá-la nas duas verificações.

O local correto para definirmos o estilo da página é no arquivo CSS. Novamente, é importante separarmos esses ajuste do arquivo HTML.

É importante mantermos essa separação dos "mundos": o HTML é utilizado para marcar os elementos, o JavaScript para adicionar dinamismo e manipular a página, enquanto o CSS é responsável por estilizar. Logo, o ideal não é modificar o estilo da linha no código JavaScript, então qual é o meio termo entre não modificar o estilo diretamente, mas conseguir alterá-lo, no JavaScript?

É uma boa prática, em vez de adicionarmos um estilo, incluirmos uma classe ao elemento. Então criaremos uma nova classe em `index.css` com dados do paciente inválido. A classe `paciente-invalido` irá indicar com quais cores

trabalharemos nestes casos:

```
.paciente-invalido {  
  background-color: lightcoral;  
}
```

De volta ao JavaScript, assim como temos o `style` para acesso aos estilos e o `textContent` para acessarmos o conteúdo de texto de um elemento, teremos a `classList` para acessarmos as classes. E usando o método `add`, adicionaremos a classe desejada, no caso, `paciente-invalido`:

```
var pesoEhValido = true;  
var alturaEhValida = true;  
  
if (peso <= 0 || peso >= 1000) {  
  console.log("Peso inválido!");  
  pesoEhValido = false;  
  tdImc.textContent = "Peso inválido";  
  paciente.classList.add("paciente-invalido");  
}
```

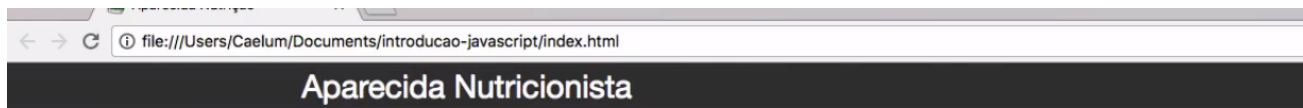
Se o designer quiser modificar a cor de fundo, deverá modificá-la no CSS e não mais no JavaScript. Inspeccionando-se uma linha cujo fundo possui alteração de cor, verificaremos no console que a classe `paciente-invalido` foi adicionada.

O próximo passo será incluir `classList.add("paciente-invalido")` também na validação da altura:

```
if (altura <= 0 || altura >= 3.00) {  
  console.log("Altura inválida!");  
  alturaEhValida = false;  
  tdImc.textContent = "Altura inválida";  
  paciente.classList.add("paciente-invalido");  
}
```

Agora, sempre que for necessária uma alteração na parte visual da página, elas serão feitas no arquivo `index.css`. Caso queiramos que o fundo da linha com o dado inválido fique com a cor azul, por exemplo, devemos alterar a classe `.paciente-invalido` no código CSS.

```
.paciente-invalido{  
  background-color: blue;  
}
```



Meus pacientes

Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	100	5.00	10	Altura inválida!
João	80	1.72	40	27.04
Erica	54	1.64	14	20.08
Douglas	85	1.73	24	28.40
Tatiana	4800	1.55	19	Peso inválido!

Não precisaremos localizar todos os pontos onde escrevemos `lightcoral` no código. Evitaremos o risco de esquecermos a alteração em algum trecho. Ao adicionarmos uma classe no CSS, nosso código fica organizado.

Vamos deixar a cor do fundo da linha com o erro como estava:

```
.paciente-invalido{
  background-color: lightcoral;
}
```

Encerramos mais uma aula do curso de JavaScript, no qual apresentamos como fazer o cálculo do IMC para todos os pacientes da tabela, em vez de um. Mostramos como selecionar vários elementos do HTML simultaneamente, usando a função `querySelectorAll()`. Com ela podemos trazer uma lista de objetos selecionados contendo a classe `.paciente`. Depois, percorremos essa lista com o laço `for`. Trabalhamos também com a propriedade `length`, que nos retorna o tamanho do array, além de aprendermos como acessar uma posição específica do array utilizando o `[]`.

Por fim, conseguimos calcular o IMC dos pacientes e arredondamos os resultados com valores decimais, fazendo uso do método `toFixed()`. E manipulamos o estilo de um objeto no JS, por meio da propriedade `style` - que não deve ser utilizada com frequência. Na aula, explicamos a importância de não misturarmos os "mundos" de linguagens diferentes, no caso, quando queremos alterar o visual da página, devemos manipular as classes do arquivo CSS. A nova classe será adicionada com `classList.add()` no JavaScript. Este tipo de ajuste é uma boa prática que deve ser adotada por facilitar a manutenção do código.

Obrigado por ter chegado até aqui e espero você na próxima aula!