

DESAFIO ESTRELAS FORA DA CAIXA

Objetivo

O objetivo do desafio é construir uma página que contém 4 personagens do Rick and Morty e seus respectivos nomes e sempre que atualizar a página é retornado outros personagens diferentes.

Desenvolvimento

Vamos começar a desenvolver o desafio primeiro através do HTML criando o nosso esqueleto para que traga todos os elementos da página.

Na área de trabalho eu criei uma pasta chamada desafio para salvar todo o projeto dentro.

Para dar inicio ao projeto no programa Visual Studio Code eu criei um arquivo html salvando na pasta criada.

No inicio da página HTML foi criado o cabeçalho incluindo as Tags necessárias para que ao carregar a página os metadados funcionem corretamente.

Ainda no cabeçalho foram declarados o link do arquivo CSS "style.css" para formatar os elementos que foram incluídos no HTML.

Logo em seguida também foi lincado no cabeçalho o arquivo em Java Script "script.js" que vai fazer a conexão e interatividade com a página.

```
<body>
  <h1>Personagens
    Rick and Morty</h1>
  <main>
    <section>
      <div id="ladoalado">
        <img id="imagem1"/>
        <p id="nomeDoPersonagem1"></p>
        <img id="imagem2"/>
        <p id="nomeDoPersonagem2"></p>
      </div>
      <img id="imagem3"/>
      <p id="nomeDoPersonagem3"></p>
      <img id="imagem4"/>
      <p id="nomeDoPersonagem4"></p>
    </section>
  </main>
```

No corpo da página foram adicionadas 3 Tags (<body>, <main> e <section>). Na Tag <body> eu coloquei um título na minha página através da <h1>. Na section eu começo colocando uma tag (<div id= "ladoalado") é um elemento de divisão que utilizei para agrupar as imagens na página.

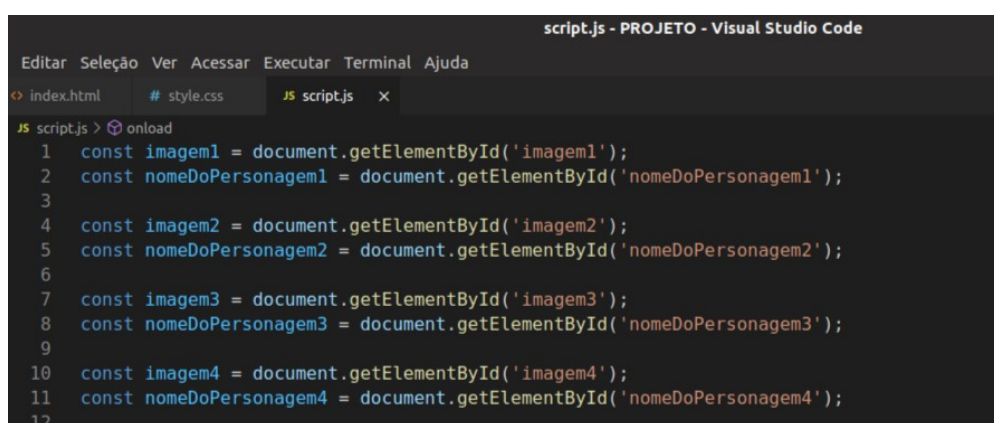
Na próxima linha foram declaradas quatro tags declarando a variável "id" que busca os parâmetros da imagem que eu quero. Também foram utilizadas quatro tags <p> uma para cada imagem que também declaram a variável "id" que vai retornar o nome dos personagens.

Na próxima parte eu criei um arquivo em CSS com o nome de #style.css que já foi vinculada ao arquivo HTML. Nessa parte eu desenvolvi um estilo para a minha página dando um visual melhor utilizando cores, formatando texto e a disposição de cada elemento na página.

Eu adicionei uma formatação para a tag *body*, para o meu título na tag *body >h1*, para a tag *main*, para a tag *section img*, tag *section p*, e *#ladoalado*.

O terceiro passo foi a criação de um arquivo Java Script com o nome de script.js que também foi vinculada no arquivo *index.html*. Sem esse vínculo a página que foi criada não funciona ou seja não vai haver uma interatividade.

No Java Script no início foram criadas quatro variáveis *const* imagens e cada imagem com uma *const* que vai retornar o nome do personagem na parte abaixo de cada figura através do método "getElementById". Esse método vai pegar o componente "id" que foi criado no HTML e colocar em uma variável constante para que possamos conseguir manipular ele.



```
script.js - PROJETO - Visual Studio Code
Editar Seleção Ver Acessar Executar Terminal Ajuda
index.html # style.css JS script.js x
JS script.js > onload
1  const imagem1 = document.getElementById('imagem1');
2  const nomeDoPersonagem1 = document.getElementById('nomeDoPersonagem1');
3
4  const imagem2 = document.getElementById('imagem2');
5  const nomeDoPersonagem2 = document.getElementById('nomeDoPersonagem2');
6
7  const imagem3 = document.getElementById('imagem3');
8  const nomeDoPersonagem3 = document.getElementById('nomeDoPersonagem3');
9
10 const imagem4 = document.getElementById('imagem4');
11 const nomeDoPersonagem4 = document.getElementById('nomeDoPersonagem4');
12
```

Na próxima parte eu criei uma função `this.onload = function(){}` para as quatro imagens e nomes dos personagens separadamente e esse evento irá fazer o carregamento da página e dentro desse onload ele dispara as funções que irão retornar as imagens sucessivamente sempre quando atualizar a página.

```
12
13  this.onload = function(){
14      capturarPersonagem(imagem1, nomeDoPersonagem1);
15      capturarPersonagem(imagem2, nomeDoPersonagem2);
16      capturarPersonagem(imagem3, nomeDoPersonagem3);
17      capturarPersonagem(imagem4, nomeDoPersonagem4);
18  };
19
```

Logo em seguida eu coloquei uma função “gerarValorAleatorio” para gerar um valor aleatório entre 1 e 671 ou seja sempre que fizer o carregamento da página vai disparar imagens de personagens diferentes e cada número corresponde a uma imagem.

Foi criado uma nova função “capturarPersonagem” que vai até a API do Rick and Morty e pega as informações da URL através da fetch e retorna para o meu arquivo.

Para fazer essa comunicação eu utilizei o método do protocolo http o “GET”. Utilizei também o “headers” que vai servir para passar o cabeçalho da requisição.

Criei a função then que vai verificar se conseguiu a resposta e transformar em Json que é uma estrutura de dados logo depois foi adicionado mais um “then” para encontrar o campo data dentro da resposta criando uma nova funcionalidade que vai poder alterar o HTML.

```
19
20  gerarValorAleatorio = () => {
21      return Math.floor(Math.random() * 671);
22  }
23
24  capturarPersonagem = (imagem,nome) => {
25      let numeroAleatorio = gerarValorAleatorio();
26      return fetch(`https://rickandmortyapi.com/api/character/${numeroAleatorio}`, {
27          method: 'GET',
28          headers: {Accept: 'application/json',
29                  "Content-type": 'application/json'
30          }
31      }).then((response) => response.json()).then((data) => {
32          imagem.src = data.image;
33          imagem.alt = data.name;
34          nome.innerHTML = data.name;
35      });
36  }
37
```

Material de apoio utilizado para o desenvolvimento do desafio:

Aulas desenvolvidas durante o projeto.

Site: <https://rickandmortyapi.com/documentation/#rest>

Site: <https://rickandmortyapi.com/documentation/#rest>

Aula disponível no You Tube do Curso em Vídeo.