Koding

MATERIAL DE APOIO:

JAVASCRIPT

por: Koding

SUMÁRIO

JAVASCRIPT 3

CAPÍTULO 1: JAVASCRIPT – INTRODUÇÃO 3

Aula 00 - Fundamentos do JavaScript 3

Aula 01 - Manipulação do Documento com JavaScript 7

Aula 02 - Introdução a Funções e Eventos 8

CAPÍTULO 2: JAVASCRIPT - PROGRAMAÇÃO ASSÍNCRONA 10

Aula 00 - Introdução à Programação Assíncrona em JavaScript 10

Aula 01 - Promises em JavaScript 11

Aula 02 - Async/Await e Fetch API 11

CAPÍTULO 3: JAVASCRIPT - FRAMEWORKS FRONT-END 13

Aula 00 - Introdução a Frameworks Front-end em JavaScript 13

Aula 01- Componentes e Roteamento em Frameworks Front-end 14

Aula 02 - Comunicação com APIs e Estado da Aplicação 15

GABARITO DO JOGO 17

REFERÊNCIAS 18

JAVASCRIPT

JavaScript é uma linguagem de programação usada por desenvolvedores para fazer páginas interativas da Internet. As funções de JavaScript podem melhorar a experiência do usuário durante a navegação em um site, como, por exemplo, desde a atualização do feed na página da mídia social até a exibição de animações e mapas interativos.

CAPÍTULO 1: JAVASCRIPT – INTRODUÇÃO

Neste capítulo seu filho irá aprender sobre os fundamentos, manipulação do documento, funções e eventos em JavaScript.

Aula 00 - Fundamentos do JavaScript

JavaScript é uma linguagem de programação usada por desenvolvedores para fazer páginas interativas da Internet. As funções de JavaScript podem melhorar a experiência do usuário durante a navegação em um site, como, por exemplo, desde a atualização do feed na página da mídia social até a exibição de animações e mapas interativos.

VARIÁVEIS:

As variáveis podem ser usadas para armazenar dados em um programa. Em JavaScript, existem três tipos de variáveis diferentes: var , let e const .

var - Tem escopo global ou escopo de função/local

O escopo é global quando uma variável var é declarada fora de uma função. var tem escopo de função quando é declarado dentro de uma função;

let – Tem escopo de bloco

Um bloco é uma porção de código cercado por {}. Um bloco vive dentro dessas chaves. Tudo o que estiver cercado por chaves é um bloco. Assim, uma variável declarada com let em um bloco estará disponível apenas dentro daquele bloco;

const - Tem escopo de bloco

Variáveis declaradas com const mantêm valores constantes. Declarações com const compartilham algumas semelhanças com as declarações com let. Assim, as declarações de const somente podem ser acessadas dentro do bloco onde foram declaradas e não pode ser atualizado nem declarado novamente.

TIPOS DE DADOS:

Strings - Strings são usadas para representar texto e devem ser declaradas entre aspas simples ou duplas;

Números - Números são usados para representar valores numéricos e podem ser inteiros ou de ponto flutuante;

Valores booleanos - Valores booleanos podem ser true ou false e são usados em expressões lógicas e estruturas de controle;

Undefined - Undefined é usado para variáveis que foram declaradas, mas não possuem um valor atribuído;

Null - Null é usado para indicar a ausência de valor;

Objetos - Objetos são estruturas de dados com múltiplas propriedades e podem ser acessados utilizando a sintaxe de ponto ou de colchetes;

Arrays - Arrays são usados para armazenar coleções de valores em uma única variável;

Funções - Funções são blocos de código que podem ser reutilizados e executados quando necessário.

OPERADORES:

Operadores De Comparação

Igual (==) - Retorna verdadeiro caso os operandos sejam iguais;

Não igual (!=) - Retorna verdadeiro caso os operandos não sejam iguais;

Estritamente igual (===) - Retorna verdadeiro caso os operandos sejam iguais e do mesmo tipo;

Estritamente não igual (!==) - Retorna verdadeiro caso os operandos não sejam iguais e/ou não sejam do mesmo tipo;

Maior que (>) - Retorna verdadeiro caso o operando da esquerda seja maior que o da direita;

Maior que ou igual (>=) - Retorna verdadeiro caso o operando da esquerda seja maior ou igual ao da direita;

Menor que (<) - Retorna verdadeiro caso o operando da esquerda seja menor que o da direita;

Menor que ou igual (<=) - Retorna verdadeiro caso o operando da esquerda seja menor ou igual ao da direita.

Operadores Aritmiéticos

Módulo (%) - Retorna o inteiro restante da divisão dos dois operandos;

Incremento (++) - Adiciona um ao seu operando. Se usado como operador prefixado (++x), retorna o valor de seu operando após a adição. Se usado como operador pósfixado (x++), retorna o valor de seu operando antes da adição;

Decremento (--) - Subtrai um de seu operando. O valor de retorno é análogo àquele do operador de incremento;

Negação (-) - Retorna a negação de seu operando;

Adição (+) - Tenta converter o operando em um número, sempre que possível;

Operador de exponenciação (**) - Calcula a base elevada à potência do expoente.

Aula 01 - Manipulação do Documento com JavaScript

Nesta aula, seu filho aprenderá a interagir dinamicamente com o conteúdo HTML usando JavaScript. Serão abordados conceitos como seleção de elementos e modificação de atributos.

SELEÇÃO DE ELEMENTOS:

getElementById() - Retorna um elemento correspondente ao id passado como parâmetro. No entanto, caso mais de um elemento possua o id passado como parâmetro, a função retorna o primeiro elemento encontrado;

getElementsByClassName() - Essa função vai retornar os elementos que possuem uma mesma classe passada como parâmetro em formato string, retornando uma coleção de elementos;

querySelector() - O querySelector() recebe o mesmo seletor que seria utilizado em CSS, no formato string. É importante lembrar que assim como no CSS, a utilização de "." para indicar classes bem como a utilização de "#" para indicar ids é obrigatória;

querySelectorAll() - Retorna uma NodeList com todos os elementos que correspondem ao seletor criado. Assim como nas coleções, podemos acessar os elementos através da notação de colchetes.

MODIFICAÇÃO DE ATRIBUTOS:

No JavaScript, existem quatro métodos para modificar atributos do elemento:

hasAttribute() - Devolve valores lógicos boolianos do tipo true ou false;

getAttribute() - Retorna o valor de um atributo especificado ou null;

setAttribute() - Adiciona ou atualiza o valor de um atributo especificado;

removeAttribute() - Remove um atributo de um elemento.

Aula 02 - Introdução a Funções e Eventos

FUNÇÕES:

Funções são blocos de construção fundamentais em JavaScript. A definição da função consiste no uso da palavra-chave *function*, seguida por:

Nome da Função.

Lista de argumentos para a função, entre parênteses e separados por vírgulas.

Declarações JavaScript que definem a função, entre chaves { }.

As variáveis definidas no interior de uma função não podem ser acessadas de nenhum lugar fora da função, porque a variável está definida apenas no escopo da função. No entanto, uma função pode acessar todas variáveis e funções definida fora do escopo onde ela está definida.

EVENTOS:

Os eventos nada mais são do que ações que deverão ser disparadas quando acontecer alguma coisa pré-definida na página.

onClick - é disparado quando um elemento é clicado uma vez pelo usuário;

onDoubleClick - é disparado quando um elemento é clicado duas vezes pelo usuário;

onKeyDown - é disparado quando o usuário pressiona uma tecla do teclado, estando com o foco naquele elemento;

onKeyUp - é disparado quando o usuário solta a tecla do teclado que havia sido pressionada, estando com o foco naquele elemento;

onFocus - é disparado quando um elemento recebe o foco;

onBlur - é disparado quando um elemento perde o foco;

onLoad - é disparado quando o elemento é carregado na página.

Para usar um evento JavaScript é muito simples, basta adicionar o atributo da tag, com o evento correspondente e, no valor do atributo, colocar o que deve acontecer quando o evento for disparado.

CAPÍTULO 2: JAVASCRIPT - PROGRAMAÇÃO ASSÍNCRONA

Neste capítulo seu filho verá sobre programação assíncrona, promises, Async/Await e Fetch API em JavaScript.

Aula 00 - Introdução à Programação Assíncrona em JavaScript

CALLBACK FUNCTIONS:

Callbacks garantem que uma função não seja executada antes que uma tarefa seja concluída, mas logo depois dessa tarefa ser concluída. Elas ajudam a desenvolver código JavaScript assíncrono e evitam que ocorram problemas e erros.

Em JavaScript, o jeito de criar uma função de callback é passá-la como um parâmetro para outra função, chamando-a novamente em seguida, logo depois que algo aconteça ou que alguma tarefa seja concluída.

Exemplo com arrow function:

```
1  setTimeout(() => { console.log("Essa
2  mensagem é exibida após 3 segundos");
3  }, 3000);
4
5
6
7
```

Aula 01 - Promises em JavaScript

Exploração do uso de Promises para lidar com operações assíncronas de maneira mais eficiente e legível.

PROMISES:

O objeto Promise representa a eventual conclusão (ou falha) de uma operação assíncrona e seu valor resultante.

Uma Promise está em um destes estados:

pending - estado inicial, nem cumprido nem rejeitado;

fulfilled - significa que a operação foi concluída com sucesso;

rejected - significa que a operação falhou.

Aula 02 - Async/Await e Fetch API

Introdução às palavras-chave async e await para simplificar a escrita de código assíncrono, juntamente com o uso da Fetch API para fazer requisições HTTP.

ASYNC/AWAIT:

async - faz com que uma função automaticamente retorne uma Promise;

await - basicamente pausa a função async até que a Promise dentro de uma função seja resolvida.

FETCH API:

A Fetch API é uma interface moderna do JavaScript para fazer requisições HTTP assíncronas, oferecendo uma maneira mais flexível e poderosa de interagir com recursos remotos em comparação com as abordagens mais antigas.

Sintaxe:

```
fetch('https://exemplo.com/dados')
.then(response => {
// Lógica para lidar com a resposta
})
.catch(error => {
// Lógica para lidar com erros
});

```

CAPÍTULO 3: JAVASCRIPT - FRAMEWORKS FRONT-END

Neste capítulo seu filho irá ver sobre frameworks, comunicação APIs e estado da aplicação em JavaScript.

Aula 00 - Introdução a Frameworks Front-end em JavaScript

Nesta aula, seu filho será introduzido ao mundo dos frameworks front-end em JavaScript e entenderá sua importância no desenvolvimento web moderno.

FRAMEWORKS FRONT-END:

Frameworks front-end são conjuntos de ferramentas e bibliotecas que fornecem estruturas e abstrações para o desenvolvimento de interfaces de usuário em aplicações web, simplificando tarefas comuns

React:

O React é uma biblioteca de JavaScript para construção de interfaces de usuário. Ele usa um conceito chamado de "componentes" para construir interfaces modulares e reutilizáveis. O React é conhecido por sua eficiência e desempenho, bem como por sua comunidade ativa.

Angular:

O Angular é um framework completo para a construção de aplicações web. Ele utiliza TypeScript e segue uma abordagem de programação orientada a objetos. O Angular oferece uma estrutura robusta para o

desenvolvimento de aplicações complexas e inclui muitos recursos integrados.

Aula 01- Componentes e Roteamento em Frameworks Front-end

COMPONENTES:

Os componentes são blocos de construção fundamentais. Eles representam partes autônomas e reutilizáveis da interface do usuário. Um componente pode encapsular HTML, CSS e lógica JavaScript relacionada, fornecendo modularidade e facilitando a manutenção do código.

Reutilização - Componentes podem ser reutilizados em diferentes partes da aplicação, promovendo a consistência visual e de comportamento.

Encapsulamento - Um componente encapsula seu próprio estado e comportamento, o que facilita o desenvolvimento e a manutenção de código.

ROTEAMENTO:

Refere-se à navegação entre diferentes páginas dentro de uma aplicação sem a necessidade de recarregar a página inteira. Isso é alcançado mantendo o estado da aplicação e manipulando dinamicamente o conteúdo exibido.

Experiência do Usuário: A navegação sem recarregar a página proporciona uma experiência de usuário mais suave e rápida.

Single Page Application (SPA): Muitos frameworks front-end são usados para construir SPAs, onde o roteamento desempenha um papel crucial.

Aula 02 - Comunicação com APIs e Estado da Aplicação

COMUNICAÇÃO COM APIS:

A comunicação com APIs (Interface de Programação de Aplicações) é fundamental para obter e enviar dados entre o front-end e o back-end. Isso permite que a aplicação web interaja dinamicamente com serviços externos, como bancos de dados e serviços web.

Algumas abordagens comuns incluem o uso de funções assíncronas, como fetch em JavaScript puro, ou bibliotecas específicas como axios para React.

ESTADO DA APLICAÇÃO:

O estado da aplicação refere-se às informações dinâmicas que a aplicação mantém durante sua execução. Em frameworks front-end, o estado é frequentemente gerenciado por meio de variáveis ou objetos especiais que podem ser observados para reagir a alterações.

Frameworks fornecem soluções para gerenciar o estado de maneira eficiente. No React, por exemplo, o estado pode ser gerenciado por meio do hook useState.

Em alguns casos, é necessário persistir o estado da aplicação para que ele não seja perdido durante recargas de página. Técnicas como armazenamento local (localStorage), cookies ou, em aplicações mais complexas, frameworks de gerenciamento de estado externos, podem ser utilizados para essa finalidade.

GABARITO DO JOGO

FASE 1:

Pergunta: **Pensando nos eventos do JavaScript, complete o evento que é disparado quando um elemento recebe o foco;**

Resposta: **onFocus** - é disparado quando um elemento recebe o foco.

FASE 2:

Pergunta: **Qual a linguagem que está sendo referida abaixo?**

Resposta: Complete a frase: React é uma biblioteca de **JavaScript**.

REFERÊNCIAS

https://aws.amazon.com/pt/what-is/javascript/#:~:text=O%20JavaScript%20surgiu%20como%20uma,layout%20do%20conte%C3%BAdo%20na%20p%C3%A1gina.

https://www.freecodecamp.org/portuguese/news/var-let-e-const-qual-e-a-diferenca/

https://awari.com.br/os-8-tipos-de-dados-em-javascript-guia-completo-para-iniciantes/?utm_source=blog&utm_campaign=projeto+blog&utm_medium=Os%208%20tipos%20de%20dados%20em%20JavaScript:%20guia%20completo%20para%20iniciantes

https://blog.cod3r.com.br/selecionando-elementos-html-no-javascript/

https://www.digitalocean.com/community/tutorials/how-to-modify-attributes-classes-and-styles-in-the-dom-pt

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Functions

https://wallky.com.br/tech/blog/o-que-sao-eventos-javascript-js.php

https://www.freecodecamp.org/portuguese/news/funcoes-de-callback-em-javascript-o-que-sao-e-como-usa-las/

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Promise#construtor

https://www.treinaweb.com.br/blog/usando-o-async-await-do-javascript

https://pt-br.legacy.reactjs.org/

https://angular.io/