AULA 5 – IMERSÃO JAVASCRIPT – COLEÇÕES E FUNÇÕES

OBJETIVO DA AULA

Praticar os conceitos vistos até o momento.

APRESENTAÇÃO

Nesta aula iremos colocar a mão na massa. Isso mesmo! Vamos fazer exemplos práticos juntos para testar os nossos conhecimentos em relação aos conceitos aprendidos nas aulas anteriores:

- Arrays;
- Arrays multifuncionais;
- Funções;
- Recursividade.

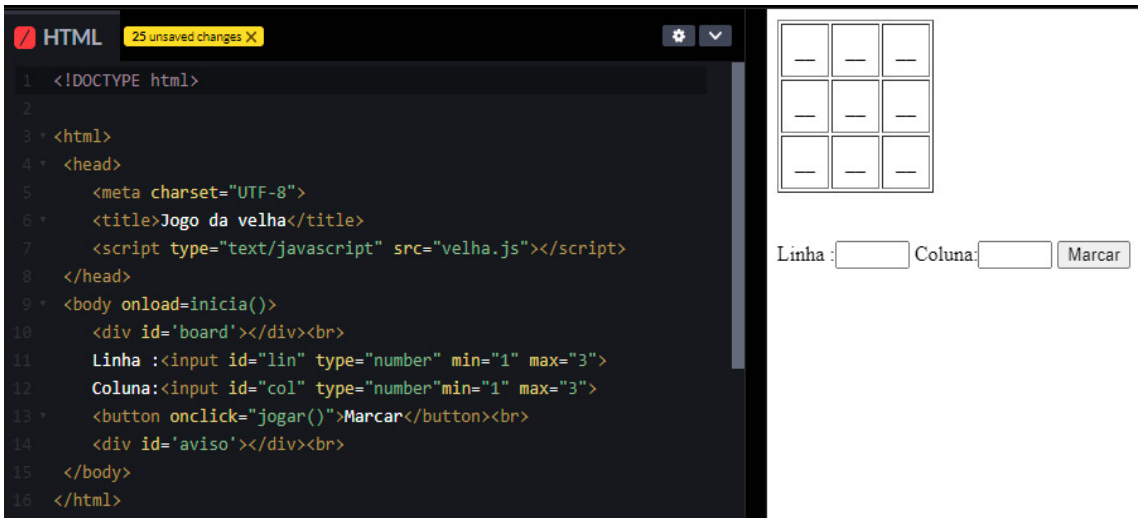
Basicamente, vamos fazer um exercício que envolva vários dos conceitos aprendidos aqui, assim teremos ideia de como esses assuntos trabalham em conjunto. A prática é muito importante nesta disciplina. Preparados? Vamos lá!

1. EXEMPLO 1 – ARRAY + ARRAY MULTIDIMENSIONAL + FUNÇÃO

Neste exemplo vamos aprender a construir um jogo da velha usando Javascript. Este exercício faz parte de um livro de De Matos (2015).

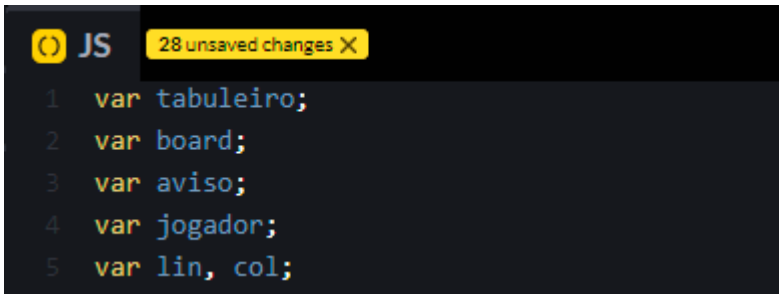
Fonte: https://www.javascriptprogressivo.net/2019/01/Codigo-Completo-Jogo-velha-JavaScript-HTML-JS.html

Como o código é relativamente grande, vamos separar o HTML do JS e começar analisando o HTML:



Link para ter acesso ao código: https://github.com/GRANCodigo/PraticaDeProgramacao/blob/8943797a40f131f9365dcae9e917018e1d707212/Unidade3Aula5a

No nosso HTML temos a tag <body> chamando a função inicial() através do evento onload. O evento onload não exige interação com o usuário, pois ele ocorre quando um objeto ou recurso tenha sido completamente carregado. Dentro do <body> temos uma div com id=board e é nela que vamos exibir o tabuleiro do jogo. Em seguida temos dois campos de formulário, onde o jogador vai indicar a linha e a coluna que deseja marcar. Como nosso tabuleiro é 3x3, os jogadores só poderão escolher números de 1 até 3. Depois de escolher o jogador clica em “Marcar” que por sua vez chama a função jogar(). A última tag do nosso body é uma div com id=aviso, onde será exibido a vez de cada jogador e também a mensagem que anunciará o jogador vencedor. Agora vamos ver o código em JavaScript que vai fazer o jogo acontecer. Vamos por partes:



O primeiro passo é declarar as variáveis que serão utilizadas fora do escopo de qualquer função, são elas:

- tabuleiro: responsável por armazenar a matriz 3x3, inicializada com todos os elementos iguais a 0. Regra: quando o primeiro jogador marca uma posição, mudamos de 0 para 1. Quando o segundo jogador marca uma posição, mudamos de 0 para -1;
- board: a variável board vai se conectar com a div board;
- aviso: a variável aviso vai se conectar com a div aviso;

- jogador: esta variável será incrementada a cada jogada. Para saber de quem é a vez de jogar, vamos utilizar o MOD(%) dela por 2, daí basta somar 1 ao resultado que teremos sempre os valores 1 e 2 se alternando;
- *lin*: responsável por pegar a posição da linha;
- *col*: responsável por pegar a posição da coluna.

A função inicia() roda apenas quando a página é carregada. Ela é responsável por criar um array de arrays (*linha 8*), ou seja, um array multidimensional. Nas *linhas 9 e 10*, as variáveis board e aviso são linkadas com o HTML. Na *linha 11*, o jogador é inicializado como 1. Em seguida, todos os elementos de array de arrays são inicializados com 0.

```
7  function inicia(){
8   tabuleiro = new Array(3);
9   board = document.getElementById('board');
10  aviso = document.getElementById('aviso');
11  jogador = 1;
12
13  for(let i=0 ; i<3 ; i++)
14   tabuleiro[i] = new Array(3);
15
16  for(let i=0; i<3 ; i++)
17   for(let j=0 ; j<3 ; j++)
18    tabuleiro[i][j]=0;
19  exibe();
20 }
```

A função exibe() é responsável por criar a estrutura da tabela (linhas, colunas) e armazenar na variável HTML. A tabela aparece na nossa página graças ao innerHTML na linha 37.

LINK

Para saber mais sobre tabelas acesse: https://www.w3schools.com/html/html_tables.asp. As linhas 27-33 exibem "—" caso encontre o valor 0 no tabuleiro, "X" caso o valor seja 1 e "O" caso o valor seja -1. Lembra da regra da variável tabuleiro descrita acima?

```
22 function exibe(){
23  HTML = '<table  cellpadding="10" border="1">';
24      for(let i=0; i<3 ; i++){
25       HTML += '<tr>';
26       for(let j=0 ; j<3 ; j++)
27        if(tabuleiro[i][j]==0)
28         HTML +='<td>   __   </td>';
29        else
30         if(tabuleiro[i][j]==1)
31          HTML +='<td> X </td>';
32         else
33          HTML +='<td> O </td>';
34       HTML += '</tr>';
35      }
36  HTML += '</table><br/>';
37  board.innerHTML = HTML
38 }
```

Primeiramente, a função jogar() exibe na tela o aviso de quem é a vez de jogar (*linha 42*). Em seguida, ela pega os valores de linha e coluna digitados pelo usuário. Lembra que tudo que é digitado num campo de formulário é considerado como string? Então, precisamos pegar essas string e converter em números (*linhas 43 e 44*). O usuário escolhe entre os números 1, 2 e 3, porém o tabuleiro contém os índices com posições 0, 1 e 2, por isso é necessário subtrair 1 do valor que foi escolhido ou digitado pelo usuário.

```
40 function jogar()
41 {
42  aviso.innerHTML='Vez do jogador: ' + ((jogador)%2 + 1);
43  lin = parseInt(document.getElementById("lin").value)-1;
44  col = parseInt(document.getElementById("col").value)-1;
45
46      if(tabuleiro[lin][col]==0)
47       if(jogador % 2)
48        tabuleiro[lin][col] = 1;
49       else
50        tabuleiro[lin][col] = -1;
51      else{
52       aviso.innerHTML='Campo ja foi marcado!'
53       jogador--;
54      }
55  jogador++;
56  exibe();
57  checa();
58 }
```

Para marcar “X” ou “O” é necessário verificar se a posição está vazia ou não tabuleiro[lin][col]==0, se estiver vazia e for o primeiro jogador coloca-se 1(“X”) na posição, se for o segundo coloca-se -1 (“O”) na posição. Caso não esteja vazio (else), a mensagem “**campo já foi marcado**” é exibida na tela e subtraímos 1 da variável jogador (*linha 52*). Ao encerrar o bloco if-else a variável jogador é incrementada para que o próximo possa realizar a sua jogada. Além disso, a função jogar() também invoca as funções exibe() e checa().

A função checa() é responsável por checar quem ganhou o jogo. Para isso, ela percorre todas as linhas, colunas e diagonais do tabuleiro para verificar se a soma é igual a 3 ou -3.

```
60  function checa()
61  {
62   var soma;
63
64       //Linhas
65       for(let i=0 ; i<3 ; i++){
66        soma=0;
67        soma=tabuleiro[i][0]+tabuleiro[i][1]+tabuleiro[i][2];
68
69        if(soma==3 || soma==-3)
70         aviso.innerHTML="Jogador " + ((jogador)%2 + 1) + " ganhou! Linha " + (i+1) + " preenchida!";
71       }
72
73       //Colunas
74       for(let i=0 ; i<3 ; i++){
75        soma=0;
76        soma=tabuleiro[0][i]+tabuleiro[1][i]+tabuleiro[2][i];
```

```
78        if(soma==3 || soma==-3)
79         aviso.innerHTML="Jogador " + ((jogador)%2 + 1) + " ganhou! Coluna " + (i+1) + " preenchida!";
80       }
81
82       //Diagonal
83       soma=0;
84       soma = tabuleiro[0][0]+tabuleiro[1][1]+tabuleiro[2][2];
85       if(soma==3 || soma==-3)
86         aviso.innerHTML="Jogador " + ((jogador)%2 + 1) + " ganhou! Diagonal preenchida!";
87
88       //Diagonal
89       soma=0;
90       soma = tabuleiro[0][2]+tabuleiro[1][1]+tabuleiro[2][0];
91       if(soma==3 || soma==-3)
92         aviso.innerHTML="Jogador " + ((jogador)%2 + 1) + " ganhou! Diagonal preenchida!";
93  }
```

O primeiro for é responsável por verificar as colunas, o segundo por verificar as linhas e o terceiro por verificar as diagonais. Se o resultado da soma for igual a 3 ou -3 é exibida uma mensagem indicando qual a linha, coluna o diagonal aquele jogador venceu. Veja:

X	X	O
_	O	X
O	O	X

Linha : 3 Coluna: 1 Marcar
Jogador 2 ganhou! Diagonal preenchida!

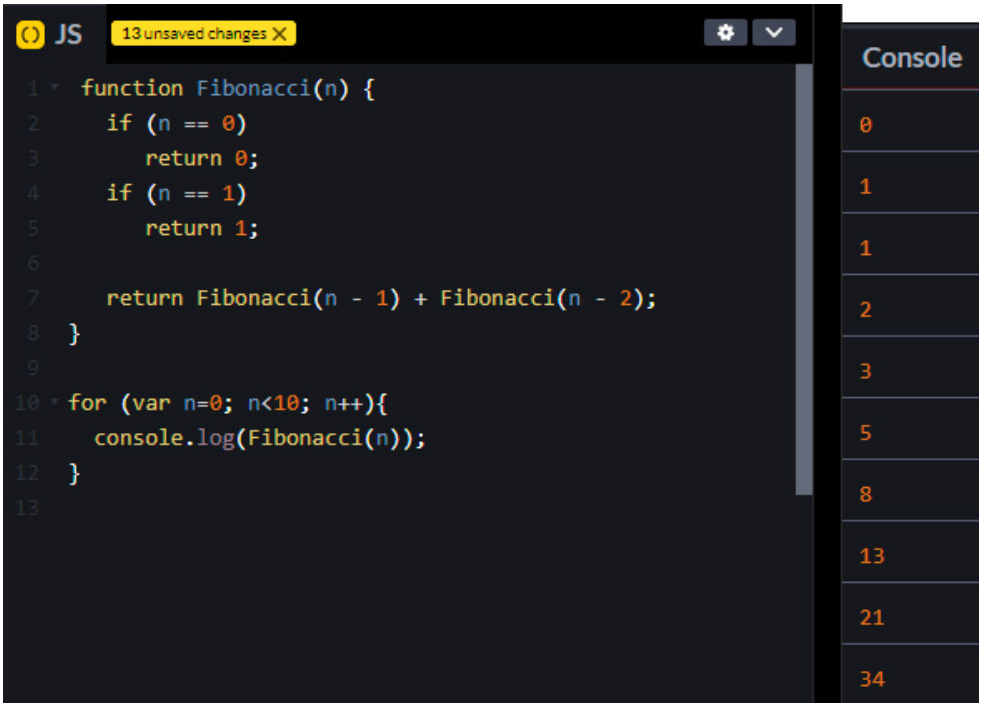
2. EXEMPLO 2 – RECURSIVIDADE

Para encerarmos esta aula, vamos fazer mais um exemplo de recursividade. Um outro exemplo clássico de recursividade é a sequência de Fibonacci. Ela é composta por uma sucessão de números que tem como primeiros termos os números 0 e 1 e, a seguir, cada termo subsequente é obtido pela soma dos dois termos predecessores:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597...

Fórmula matemática Fibonacci: Fn = Fn – 1 + Fn – 2.

Vamos criar um script para imprimir no console os 10 primeiros números da sequência de Fibonacci utilizando a recursividade.



Repare que aqui foi necessário chamar a função Fibonacci dentro do console.log, que por sua vez está dentro de uma estrutura de repetição para que a sequência fosse impressa.

CONSIDERAÇÕES FINAIS

Nesta aula tivemos a oportunidade de colocar a mão e aprender uma pouco mais sobre como implementar arrays de arrays, como definir e chamar funções e como trabalhar com recursividade. A ideia é que a partir de agora você consiga dar os seus próprios passos. Aproveite para acessar, explorar e editar o código do jogo da velha que está disponível na apostila. JavaScript não é difícil, mas a prática se faz necessário. Sendo assim, deixo a disposição de vocês um link (https://oprogramador.bsb.br/aprenderjs_exercicios.php?page=1) com exercícios resolvidos para que vocês possam testar ainda mais os conhecimentos adquiridos em nossas aulas. Bons estudos!

MATERIAIS COMPLEMENTARES

Javscript – recursão, fatorial e fibonacci:

https://youtu.be/H5u1dOoCPnc

REFERÊNCIAS

DE MATOS, F. J. M. JavaScript Progressivo: *Curso completo de JavaScript para iniciantes*, 2015. Disponível em: <https://www.javascriptprogressivo.net/>. Acesso em: 20 de nov. de 2022.