

STEP Javascript - Homework 25

Adaptação dos dois exercícios anteriores de Xadrez - para JOGO DA VELHA

Nos dois homeworks anteriores (e continuando como exercício em sala de aula) vínhamos construindo um tabuleiro de xadrez com as peças em suas posições iniciais, usando somente funções anônimas e objetos (e com o requerimento de se usar arrays multidimensionais). Além disso, havíamos implementado funções simples para movimentação das pedras.

Ao final deste doc, segue o código em sua versão mais recente até a última aula.

Exercício: Adapte o código de forma que vire um jogo da velha (nota: retire os console.log() alerts depois que tiver debugado)

Caso o movimento informado seja inválido, uma mensagem deverá ser exibida.

A resolução do exercício pode ser uma adaptação simples. Porém, se você quiser sofisticar, tente aumentar a manipulação de objetos e reduzir o acesso direto a posições de array (na forma a[i,j]). Tente também em adição, se quiser, fazer algumas funções anônima que não sejam autoexecutáveis. As funções anônimas tem muitas utilidades além do formato especial de auto-execução na carga da página.

=====

Entregue A PASTA (diretório) com os arquivos .html, .js e outros se houverem. Não entregue .txt, .docx, etc. Faça um zip da pasta e upload deste zip

=====

=== Código do exercício do xadrez, última posição ===

index.html

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="estilo.css">
```

```
<script defer src="codigo.js" ></script>
<title>testoso</title>
</head>

<body>
  <div id="externa"></div>
</body>
</html>
```

estilo.css

```
.casa {
  display: inline-block;
  padding: 0;
  height: 30px;
  width: 30px;
  position: relative;
  text-align: center;
  color: rgb(255, 0, 0);
  cursor: pointer;
}

.branco {
  background-color: rgb(241, 230, 216);
}

.preto {
  background-color: rgb(82, 81, 81);
}
```

codigo.js

```
// pecaAserJogada = null

// var flag = false;

// class Casa {
//   peca = []
```

```
// }
```

```
class Peca {  
    constructor(nome, sigla, color) {  
        this.nome = nome;  
        this.sigla = sigla;  
        this.cor = color  
        this.posicao = ""  
    }  
}
```

```
class Torre extends Peca {  
    constructor(core) {  
        super("torre", "T", core)  
    }  
}
```

```
class Cavalo extends Peca {  
    constructor(core) {  
        super("cavalo", "C", core)  
    }  
}
```

```
class Bispo extends Peca {  
    constructor(core) {  
        super("Bispo", "B", core)  
    }  
}
```

```
class Rei extends Peca {  
    constructor(core) {  
        super("rei", "Re", core)  
    }  
}
```

```
class Rainha extends Peca {  
    constructor(core) {
```

```

        super("rainha", "Ra", core)
    }
}

class Peao extends Peca {
    constructor(core) {
        super("peao", "P", core)
    }
}

tabuleiro = [
    [new Torre("branco"), new Cavalo("branco"), new Bispo("branco"), new
Rei("branco"), new Rainha("branco"), new Bispo("branco"), new Cavalo("branco"),
new Torre("branco")],
    [new Peao("branco"), new Peao("branco"), new Peao("branco"), new
Peao("branco"), new Peao("branco"), new Peao("branco"), new Peao("branco"), new
Peao("branco")],
    ["", "", "", "", "", "", "", ""],
    ["", "", "", "", "", "", "", ""],
    ["", "", "", "", "", "", "", ""],
    ["", "", "", "", "", "", "", ""],
    [new Peao("preta"), new Peao("preta"), new Peao("preta"), new Peao("preta"),
new Peao("preta"), new Peao("preta"), new Peao("preta"), new Peao("preta")],
    [new Torre("preta"), new Cavalo("preta"), new Bispo("preta"), new
Rei("preta"), new Rainha("preta"), new Bispo("preta"), new Cavalo("preta"), new
Torre("preta")]
];

(function exibeTabuleiro() {
    var cor = "branco";
    for (var i = 0; i < 8; i++) {
        for (var j = 0; j < 8; j++) {
            posicao = document.createElement("div");

            if (tabuleiro[i][j] == "") {
                posicao.innerHTML = "&nbsp;";
            }
        }
    }
}());

```

```

    } else {
        posicao.innerHTML = tabuleiro[i][j].sigla
        tabuleiro[i][j].posicao = (i, j)
    }
    paramIJ = i + "," + j

    casaCor = "casa " + cor
    posiIJ = "" + i + j
    console.log(casaCor)
    posicao.setAttribute("class", casaCor)
    posicao.setAttribute("id", posiIJ)
    posicao.setAttribute("onclick", "clicar(" + paramIJ + ")")
    document.getElementById("externa").appendChild(posicao)

    cor = trocaCor(cor);
}
cor = trocaCor(cor);
document.getElementById("externa").innerHTML += "<br>"
}
})();

```

```

function trocaCor(cor) {
    var color;
    if (cor == "branco") {
        color = "preto"
    } else {
        color = "branco";
    }
    return color;
}

```

```

var globalVariavelEscrota = false
var x = null;

```

```

function clicar(i, j) {

```

```
// x é temp (swap)
// class1 e 2 são posições na verdade
if (globalVariavelEscrota == true) {
    window.alert("Vc escolheu agora pa onde essa peça foi")
    tabuleiro[i][j] = tabuleiro[x[0]][x[1]]
    console.log("" + i + j)
    tabuleiro[x[0]][x[1]] = ""
    console.log("" + x[0] + x[1])
    class1 = "" + i + j
    class2 = "" + x[0] + x[1]
    console.log("o tabuleiro[i][j].sigla é " + tabuleiro[i][j].sigla)
    document.getElementById(class1).innerHTML = tabuleiro[i][j].sigla
    document.getElementById(class2).innerHTML = "&nbsp;"
    globalVariavelEscrota = false

    console.log(tabuleiro)
} else {
    window.alert("escolheu a peça")
    window.alert("diga para onde ela vai")
    x = [i, j]
    globalVariavelEscrota = true
}
}
```