Aula Prática 12 – JavaScript

Professor: Luiz Carlos Felix Carvalho

JavaScript é uma linguagem de script orientada a objetos que pode ser utilizada tanto no cliente (navegador web) quanto no servidor (Node.js). Nessa disciplina, o uso de JavaScript será abordado no lado cliente.

JavaScript Sintaxe

JavaScript possui sintaxe baseada no Java, AWK, Perl e Python. Algumas características da sintaxe:

- Case-sensitive
- Utiliza caracteres Unicode
- Instruções são chamadas de declarações e são separadas por ';'

Comentários

```JS
// comentário de uma linha

/* isto é um comentário longo
   de múltiplas linhas.
 */

/* Você não pode, porém, /* aninhar comentários */ SyntaxError */
```

Declarações

var: Declara uma variável, opcionalmente, inicializando-a com um valor.

let: Declara uma variável local de escopo do bloco, opcionalmente, inicializando-a com um valor.

const: Declara uma constante de escopo de bloco, apenas de leitura.

Vetor / Array

```
var myArray = [];
```

Tipos de Dados Primitivos

- Boolean
- null
- undefined
- Number
- String
- Symbol
- Object

Condicional

```
if (condicao) {
  declaracao_1;
} else if (condicao_2) {
  declaracao_2;
} else if (condicao_n) {
  declaracao_n;
} else {
  declaracao_final;
}
```

Switch case

```
switch (expressao) {
   case rotulo_1:
      declaracoes_1
      [break;]
   case rotulo_2:
      declaracoes_2
      [break;]
   ...
   default:
      declaracoes_padrao
      [break;]
}
```

Declaração for

```
for ([expressaoInicial]; [condicao]; [incremento])
  declaracao
```

```
for (var i = 0; i < selectObject.options.length; i++) {
  if (selectObject.options[i].selected) {
    numeroSelecionadas++;
  }
}
```

Declaração do...while

```
do
  declaracao
while (condicao);
```

```
do {
  i += 1;
  console.log(i);
} while (i < 5);
```

Declaração while

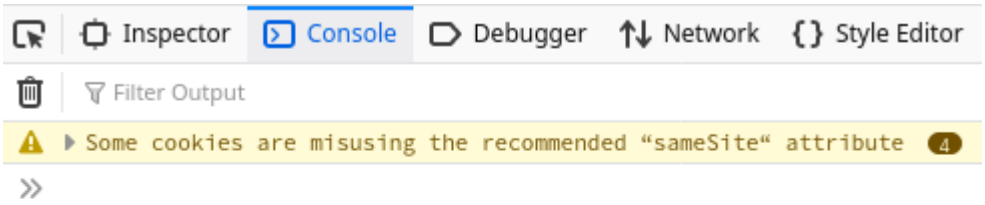
```
while (condicao)
  declaracao
```

```
n = 0;
x = 0;
while (n < 3) {
  n++;
  x += n;
}
```

Console

Uma ferramenta muito útil para explorar o JavaScript é o JavaScript Console (às vezes chamado de Web Console, ou apenas o console): é uma ferramenta que permite inserir JavaScript e executá-lo na página atual.

Para acessar o console em um navegador, acesse a ferramenta do desenvolvedor (F12 no Chrome e Firefox) e localize a aba “Console”.



O console aparece na parte inferior da janela do navegador. Na parte inferior do console há uma linha de entrada que você pode usar para inserir JavaScript, e a saída aparece no painel acima.

O console funciona exatamente da mesma forma que eval: a última expressão digitada é retornada.

Obs.: A função Eval avalia o expressão de cadeia de caracteres e retorna seu valor. Por exemplo, Eval("1 + 1") retorna 2. Se você passar para a função Eval uma cadeia de caracteres que contém o nome de uma função, a função Eval retornará o valor de retorno da função.

JS

```
console.log(eval("3 + 5"));
```

Várias Linhas no Console

- Se a string digitada estiver incompleta (por exemplo, você digitou function foo() {), o console tratará Enter como uma quebra de linha e permitirá que você digite outra linha.
- Se você segurar Shift enquanto pressiona Enter, o console tratará isso como uma quebra de linha e permitirá que você digite outra linha.
- Somente no Firefox, você pode ativar modo de entrada multi-linha, em que você pode inserir várias linhas em um mini-editor e, em seguida, executar tudo quando estiver pronto.

Teste o Console

Abra o console e digite:

```
/* Ínicio do seu código */
function greetMe(seuNome) {
  alert(`Olá ${seuNome}`);
}

greetMe("Mundo");
/* Fim do seu código */
```

Depois tecle Enter.

Obs.: utilize console.info() para imprimir um texto (será impresso no console)

Vetores (Arrays)

São objetos que possuem vários (um ou mais) valores armazenados em sequência, identificados (cada um) por um índice.

Criando vetores:

1. Criando uma lista de times:

```
let times = [ "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca" ];
```

2. Outras possibilidades:
   Em JavaScript o vetor pode armazenar qualquer tipo de dado (número, string, objeto etc.) e pode armazenar, em um mesmo vetor, diferentes tipos de dados:

```
let timesRanking = [ "Corinthians", 1, "Uberlândia", 3, "XV de Piracicaba", 4, "Asa de Arapiraca", 2 ];
```

Acesso e modificação:

1. Considere que o vetor criado anteriormente, denominado timeRanking, possui o nome de um time e, em sequência, seu ranking nessa lista. Vamos alterar o valor do ranking do Uberlândia e do Asa de Arapiraca, fazendo uma inversão dos valores. Para tal utilizaremos uma variável auxiliar, aux;

```
let timesRanking = [ "Corinthians", 1, "Uberlândia", 3, "XV de Piracicaba", 4, "Asa de Arapiraca", 2 ];

let aux = timesRanking[ 3 ];
timesRanking[ 3 ] = timesRanking[ 7 ];
timesRanking[ 7 ] = aux;
```

2. Quando utilizamos vetor dentro de vetor, dizemos que temos um vetor multidimensional.

```
let timesRanking2 = [ [ "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa
de Arapiraca" ], [ 1, 3, 4, 2 ] ];

//Acessar nome e ranking do primeiro time
timesRanking2[0][0]; // Corinthians
timesRanking2[1][0]; // 1

//Acessar nome e ranking do segundo time
timesRanking2[0][1]; // Uberlândia
timesRanking2[1][1]; // 3
```

Observação: lembre-se que uma string pode ser tratada como um vetor, assim um vetor de strings, pode ser tratado como um vetor multidimensional.

Tamanho de um vetor

Propriedade **length:**

```
timesRanking.length; // retorna 8
```

Exemplo de iteração usando a propriedade length:

```
timesRanking = [ "Corinthians", 1, "Uberlândia", 3, "XV de Piracicaba", 4, "Asa de
Arapiraca", 2 ];

for ( let i = 0; i < timesRanking.length; i++ ) {

    console.log( timesRanking[ i ] );
}
```

Métodos úteis de um vetor

Split

Divide uma string, segundo o caracter informado:

```
let timesString = "Corinthians,Uberlândia,XV de Piracicaba,Asa de Arapiraca";

let times3 = timesString.split( "," );

times3; //[ "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca" ];
```

Join

“Junta” um vetor em uma string

```
let timesString3 = times3.join( "," );

timesString3; //"Corinthians,Uberlândia,XV de Piracicaba,Asa de Arapiraca";
```

toString

Converter vetor em uma string.

```
timesString3 = times3.toString();

timesString3; //"Corinthians,Uberlândia,XV de Piracicaba,Asa de Arapiraca";
```

Adicionando e removendo itens de um vetor - push() e pop()

Push

Adiciona elementos no fim de um vetor, retornando o novo tamanho do vetor.

```
let novoTamanho = times3.push( "Uberaba" );
times3; //[ "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca",
"Uberaba" ];

novoTamanho = times3.push( "Sinop", "Remo" );
times3; //[ "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca",
"Uberaba", "Sinop", "Remo" ];
```

Pop

Remove e retorna o último elemento de um vetor.

```
let itemRemovido = times3.pop();
itemRemovido; // Remo

let itemRemovido2 = times3.pop();
itemRemovido2; // Sinop
```

Adicionando e removendo itens de um vetor - unshift() e shift()

Unshift() e shift() funcionam da mesma forma do que push() e pop(), respectivamente, porém considerando o início do vetor.

```
times3.unshift( "Sinop", "Remo" );
times3; //[ "Sinop", "Remo", "Corinthians", "Uberlândia", "XV de Piracicaba", "Asa
de Arapiraca", "Uberaba" ];

itemRemovido = times3.shift();
itemRemovido; // Sinop

itemRemovido2 = times3.shift();
itemRemovido2; // Remo
```

Funções

A declaração de uma função consiste no uso da palavra chave function (função), seguida

pelo: 1) nome da função, 2) lista de argumentos entre parênteses e separados por vírgula e 3) pelo código JavaScript que define a função, entre chaves { }.

Exemplo:

```
function square(numero) {
  return numero * numero;
}
```

Observações:

1) Parâmetros primitivos (como um número) são passados para as funções por valor; o valor é passado para a função, mas se a função altera o valor do parâmetro, esta mudança não reflete globalmente ou na função chamada.
2) Se você passar um objeto (ou seja, um valor não primitivo, tal como vetor (Array) ou um objeto definido por você) como um parâmetro e a função alterar as propriedades do objeto, essa mudança é visível fora da função.

Execute o seguinte exemplo e veja o comportamento explicado nas observações anteriores.

```
function campeaoMundial( time ) {
    time.titulosMundiais++;
}

var corinthians = { nome: "Corinthians", titulosMundiais: 1 };

var mundiaisAntes2012 = corinthians.titulosMundiais;

//2012
campeaoMundial( corinthians );

var mundiais2013 = corinthians.titulosMundiais;

console.info( mundiaisAntes2012 );
console.info( mundiais2013 );
```

Exercícios:

Exercícios de 1 a 6: vamos criar alguns algoritmos em JavaScript:
Obs.: você pode utilizar o console para construir os algoritmos. Assim, utilize console.info() para imprimir alguma informação.

1. Faça um “Olá Mundo” utilizando o método **alert** para imprimir a mensagem.

2. Faça um código que, a partir de três números, descubra qual o maior.

3. Implemente um código que calcule a média de três números.

4. Crie um programa que exiba todos os números primos que estão entre 0 e 200.

5. Cálculo do Delta (Δ) de uma função de segundo grau.

Sabendo que:

a = 2, b = 8 e c = – 24

O valor de delta é dado pela seguinte expressão: $\Delta = b^2 – 4ac$, em que a, b e c são coeficientes da equação e Δ é delta.

Tomando o exemplo anterior, na equação $2x^2 + 8x – 24 = 0$, delta vale:

$\Delta = b^2 – 4ac$
$\Delta = 8^2 – 4 \cdot 2 \cdot (– 24)$
$\Delta = 64 + 192$
$\Delta = 256$

Crie um programa JavaScript que faça esse cálculo, mostrando o cálculo passo a passo e o resultado final, a partir de números declarados no próprio código.

6. Implemente um algoritmo para calcular o fatorial de um número. Declare o número no próprio código. Lembre-se de que: 0!=1.

7. Veja o seguinte código apresentado anteriormente:

```
timesRanking = [ "Corinthians", 1, "Uberlândia", 3, "XV de Piracicaba", 4, "Asa de
Arapiraca", 2 ];

for ( let i = 0; i < timesRanking.length; i++ ) {

    console.log( timesRanking[ i ] );
}
```

A variável timeRanking é um vetor que possui nas posições pares o nome de um time e nas posições ímpares o ranking de tal time da posição anterior. Altere o código, para que possa imprimir na tela o ranking seguido do nome do time, da seguinte forma:
1 – Corinthians
3 – Uberlândia
4 – XV de Piracicaba
2 – Asa de Arapiraca

8. Crie uma função JavaScript que receba um vetor e retorne um objeto contendo o primeiro e último elementos de tal vetor.

Entrada: ["Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca"]
Retorno: { posicao1: “Corinthians”, posicaoFinal: “Asa de Arapiraca” }
Obs.: vetor não é alterado

9. Reimplemente o exercício anterior, mas, agora, removendo os elementos do vetor.

Entrada: ["Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca"]
Retorno: { posicao1: “Corinthians”, posicaoFinal: “Asa de Arapiraca” }
Obs.: vetor de parâmetro: [“Uberlândia", "XV de Piracicaba"]

10. Crie uma função JavaScript que receba um vetor e um objeto, de modo que esse

objeto tenha as mesmas propriedades do objeto retornado no item 2. O comportamento da função deve ser o contrário do item 3, adicionando os elementos no vetor.

Entrada: [“Uberlândia", "XV de Piracicaba"]
{ posicao1: “Corinthians”, posicaoFinal: “Asa de Arapiraca” }
Retorno: ["Corinthians", "Uberlândia", "XV de Piracicaba", "Asa de Arapiraca"]