INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

AULA:

Tipos de dados e operadores

Disciplina: Programação de Sistemas para Internet

Alba Lopes, Profa.

http://docentes.ifrn.edu.br/albalopes
alba.lopes@ifrn.edu.br

Agenda

- Variáveis e Tipos de Dados
  - Tipos de dados
  - Aritmética de variáveis
- Operadores relacionais
  - Exercícios
- Arrays
  - Exercícios
  - Estudo Dirigido

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Variáveis e Tipos de Dados

- Javascript possui os seguintes tipos de dados:
  - string, number, boolean, array, object.

Exemplo

```
var idade = 11;                                    // Número inteiro
var altura = 1.55;                                 // Número flutuante
var trouxa = true;                                 // Booleano
var escola = "Hogwarts";                           // String
var feiticos = ["lumos", "alorromora"];            // Array
var x = {nome:"Hermione", sobrenome:"Granger"};    // Objeto
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Variáveis e Tipos de Dados

- Javascript é linguagem uma que possui tipagem dinâmica
  - Não é preciso definir o tipo do dado no momento da criação da variável.
  - Uma mesma variável pode armazenar números como 100 e textos como ‘Alba Lopes’.

- **Exemplo**

```
var x = 11;                                  // Número inteiro
x = 'Hogwarts';                              // String
x = true;                                    // Booleano
x = ["lumos", "alohomorra", "acio"];         // Array
```

Variáveis do tipo String podem ser definidas tanto usando aspas simples como aspas duplas.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Variáveis e Tipos de Dados

- Como Javascript é uma linguagem de tipagem dinâmica, é possível utilizar o operador **typeof** para identificar o tipo de uma determinada variável.

```
<script>
    var x = 16;
    alert(typeof(x));  //retorna number
    x = 'Hogwarts';
    alert(typeof(x));  //retorna string
    x = true;
    alert(typeof(x));  //retorna boolean
    x = ["lumos", "alohomorra", "acio"];
    alert(typeof(x));  //retorna object
</script>
```

Em Javascript, array é um tipo especial de objeto. Por isso o operador **typeof** retorna *object* para array.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Variáveis e Tipos de Dados

- Em JavaScript, uma variável sem valor algum tem seu valor definido como **undefined**

```
<script>
   var x;
   alert(x);         //retorna undefined
   alert(typeof(x)); //retorna undefined
```

Declarar uma variável e não atribuir valor nenhum a ela impossibilita definir que tipo de dado ela armazena.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

- Números
  - A aritmética usando variáveis do tipo numérico pode ser feito através dos operadores apresentados a seguir:

Operador	Operação	Exemplo	Atribuição
+	Adição	c = a + b	a += b
-	Subtração	c = a - b	a -= b
*	Multiplicação	c = a * b	a *= b
/	Divisão	c = a / b	a /= b
%	Módulo da divisão ou Resto	c = a % b	a %= b
++	Incremento	c = a++	a++
--	Decremento	c = a--	a--

  - Exemplo:

```
var x = 10;
x = x + 1;
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

- Strings
  - O operador aritmético que pode ser usado em strings é o operador de +.
  - Esse operador é usado para concatenação de valores de variáveis.

  - **Exemplo:**

```
var nome = "Hermione";
var nomeCompleto = nome + " Granger";
alert(nomeCompleto);
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

- Números e strings
  - A avaliação das operações em Javascript é feita da esquerda para a direita
  - Adicionar dois números irá retonar um número
  - Adicionar um número e uma string irá retornar uma string.

- **Exemplo:**

```
a = 5 + 5;
b = "5" + 5;
c = "Olá" + 5;
d = 5 + 5 + "Olá";
e = "Olá" + 5 + 5;
f = "5" + "5";
```

Resultado:

```
10
55
Olá5
10Olá
Olá55
55
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

Exemplo (1/2)

- Página HTML com dois campos de texto e um botão. Incluir uma ação no botão para somar os dois valores.

HTML

```
<form>
    N1: <input type="text" id="n1"/><br/>
    N2: <input type="text" id="n2"/><br/>
    <button type="button" onclick="somar()">Somar</button><br/>
    R: <input type="text" id="resultado"/><br/>
</form>
```

JAVASCRIPT

```
<head>
    <script>
        function somar(){
            var x = document.getElementById("n1").value;
            var y = document.getElementById("n2").value;
            var soma = x + y;
            document.getElementById("resultado").value = soma;
        }
    </script>
</head>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

Exemplo (2/2)

- Veja que nesse exemplo, independente do valor digitado, o código executará aritmética de strings.
- Isso acontece porque o valor do atributo value de campos de formulário serão sempre do tipo string.
- Para poder executar aritmética numérica utilizando valores de entrada do usuário, deve-se realizar uma conversão de valores (**parseInt**, **parseFloat**, **parseDouble**)
- O código então fica como o código a seguir:

JAVASCRIPT

```
<head>
    <script>
        function somar(){
            var x = document.getElementById("n1").value;
            var y = document.getElementById("n2").value;
            var soma = parseInt(x) + parseInt(y);
            document.getElementById("resultado").value = soma;
        }
    </script>
</head>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Aritmética de variáveis

- Quando um valor **undefined** é convertido para inteiro, o resultado é apresentando como **NaN**

```
<script>
   var x;
   alert(parseInt(x));          //retorna NaN
```

NaN é uma sigla que representa **Not a Number**

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Operadores Relacionais

- Em JavaScript os operadores relacionais retornam um valor booleano (true ou false) e são:

Operador	Operação	Exemplo
<	Menor	a < b
>	Maior	a > b
==	Igual	a == b
!=	Diferente	a != b
>=	Maior ou igual	a >= b
<=	Menor ou igual	a <= b

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Operadores Relacionais

- **Diferença entre os operadores == e ===**
  - Em Javascript, assim como em outras linguagens de tipagem dinâmica, há diferença entre o operador == e o operador ===

Operador	Descrição
==	Faz conversão de tipo e retorna verdadeiro se as variáveis possuem valores iguais.
===	Retorna verdadeiro apenas se as variáveis são do mesmo tipo e possuem valores iguais
!=	Faz conversão de tipo e retorna falso baseado apenas no valor das variáveis.
!==	Retorna verdadeiro se as variáveis são de tipos diferentes, mesmo que possuem diferentes iguais

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Operadores Relacionais

- **Diferença entre os operadores == e ===**
  - **Exemplo**

```
var y = 1;
var z = '1';

if(y == z){
    document.write("Os valores com == são iguais<br/>");
}else{
    document.write("Os valores com == são diferentes<br/>");
}

if(y === z){
    document.write("Os valores com === são iguais<br/>");
}else{
    document.write("Os valores com === são diferentes<br/>");
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Operadores Relacionais

- **Exemplo:**

HTML

```
<h3> O feitiço correto para acender uma luz é: </h3>
<input type="text" id="resposta"/><br/>
<button type="button" id="bresponder" onclick="corrigir()">Responder</button><br/>
```

JAVASCRIPT

```
<script>
function corrigir(){
     var resposta = document.getElementById("resposta").value;
     if (resposta == "lumos"){
          document.write("Parabéns! Você acendeu a luz.");
          document.body.style.backgroundColor = "yellow";
     }else{
          document.write("Que pena! Você continua no escuro.");
          document.body.style.backgroundColor = "gray";
     }
}
</script>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Operadores Relacionais

- **Exemplo:**

Separando completamente o Javascript do HTML com o uso do método addEventListener

HTML

```
<h3> O feitiço correto para acender uma luz é: </h3>
<input type="text" id="resposta"/><br/>
<button type="button" id="bresponder">Responder</button><br/>
```

JAVASCRIPT

```
<script>
 window.onload = function (){
     document.getElementById("bresponder").addEventListener("click", corrigir);
};

function corrigir(){
     var resposta = document.getElementById("resposta").value;
     if (resposta == "lumos"){
          document.write("Parabéns! Você acendeu a luz.");
          document.body.style.backgroundColor = "yellow";
     }else{
          document.write("Que pena! Você continua no escuro.");
          document.body.style.backgroundColor = "gray";
     }
}
</script>
```

Exercícios

- Crie páginas web utilizando HTML, JavaScript e CSS que resolva os seguintes problemas

1. Crie uma página com um cabeçalho contendo uma pergunta: “Que forma geométrica é essa?”. Sua página deve conter também uma figura de um quadrado, um campo de texto para o usuário digitar a resposta e um botão. Ao digitar um nome no campo de texto e clicar no botão, verifique se o usuário digitou quadrado. Em caso afirmativo, parabenize o usuário. Caso contrário, informe que a resposta está errada.
2. Crie uma página com um cabeçalho contendo uma pergunta: “Que forma geométrica você gostaria de conhecer?”. Sua página deve conter também, uma figura de uma interrogação, um campo de texto e um botão. No campo de texto, o usuário deverá digitar o nome de uma forma geométrica (quadrado, triângulo ou círculo). Ao clicar no botão, altere o src da imagem para exibir a imagem correspondente à forma digitada pelo usuário. Caso seja inválida, mantenha a interrogação e informe uma mensagem de erro.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

- Arrays são usados em linguagens de programação para armazenar múltiplos valores em um única variável.
  - **Declaração de múltiplas variáveis**

```
var casa1 = "Grifinória";
var casa2 = "Sonserina";
var casa3 = "Lufa-Lufa";
var casa4 = "Corvinal";
```

  - **Declaração de um array, “economizando” variáveis**

```
var casas = ["Grifinória", "Sonserina", "Lufa-Lufa", "Corvinal"];
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

- Os elementos de um array são acessados através de **índices numéricos**, iniciando a partir do índice **0**.

```
var casas = ["Grifinória", "Sonserina", "Lufa-Lufa", "Corvinal"];
alert("A casa de Harry Potter é a " + casas[0]);
```

```
alert("Em Hogwarts há " + casas.length + " casas.");
```

- A quantidade de elementos no array pode ser acessada através da propriedade **length.**

O ultimo element do array é sempre o tamanho do array – 1. Ex: casas[casas.length-1]

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Inicializando arrays vazios

- Para criar um array para posterior adição de elementos, basta abrir colchetes após o operador de atribuição

```
var grifinoria = [];
```

Adicionando novos elementos no array

- Após a inicialização, a forma mais simples de adicionar elementos no array é através do método **array.push**

```
grifinoria.push("Harry Potter");
```

- Também é possível atribuir elementos a um determinado índice do array

```
grifinoria[1] = "Hermione Granger";
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Exemplo

HTML

```
<h3>Chapéu Seletor</h3>
Novo estudante: <input type="text" id="estudante"/>
<button type="button" onclick="selecionar()">Grifinória</button><br/>
<h3>Novos alunos na Grifinória</h3>
<div id="listadealunos"></div>
```

JAVASCRIPT

```
<script>
    var grifinoria = [];

    function selecionar(){
        var novoestudante = document.getElementById("estudante").value;
        grifinoria.push(novoestudante);
        alert(novoestudante + " selecionado para Grifinória!");
        document.getElementById("estudante").value = "";
    }
</script>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Percorrendo um array

- Para percorrer um array e efetuar operações com um determinado elemento, pode-se utilizar laços de repetição.
- Javascript, assim como outras linguagens de programação oferece os laços **for**, **while** e **do...while**

HTML

```
Novo estudante: <input type="text" id="estudante"/>
<button type="button" onclick="selecionar()">Grifinória</button><br/>

<h3>Novos alunos na Grifinória</h3>
<button type="button" onclick="verLista()">Atualizar lista de alunos</button><br/>
<div id="listadealunos"></div>
```

JAVASCRIPT

```
var grifinoria = [];
function selecionar(){/*...*/}

function verLista(){
     var lista = "";
     for (i=0; i<grifinoria.length; i++){
          lista += grifinoria[i];
          lista += "<br/>";
     }
     document.getElementById("listadealunos").innerHTML = lista;
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Exemplo (1/2)

- Aprimorando a exibição dos dados anteriores, exibir nome dos estudantes em uma tabela.

JAVASCRIPT

```
function verLista(){
    var lista = "<table class='tabela'>";
    lista += "<tr>";
    lista += "    <th>Aluno</th>";
    lista += "    <th>Casa</th>";
    lista += "</tr>";

    for (i=0; i<grifinoria.length; i++){
        lista += "<tr><td>";
        lista += grifinoria[i];
        lista += "</td><td>Grifinória</td><tr>";
    }
    lista += "</table>";
    document.getElementById("listadealunos").innerHTML = lista;
}
```

CSS

```
<style>
    table, td, th{
        border-collapse: collapse;
        border: 1px solid black;
    }

    .
</style>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Exemplo (2/2)

- Alternando a cor das linhas da tabela.

JAVASCRIPT

```
function verLista(){
     var lista = "<table class='tabela'>";
     lista += "<tr>";
     lista += "    <th>Aluno</th>";
     lista += "    <th>Casa</th>";
     lista += "</tr>";

     for (i=0; i<grifinoria.length; i++){
          if (i%2==0) {lista += "<tr  class='par'><td>";}
          else {lista += "<tr class='impar'><td>";}
          lista += grifinoria[i];
          lista += "</td><td>Grifinória</td><tr>";
     }
     lista += "</table>";
     document.getElementById("listadealunos").innerHTML = lista;
}
```

CSS

```
<style>
     table, td, th{
          border-collapse: collapse;
          border: 1px solid black;
     }
     .par{
          background-color: yellow;
     }
     .impar{
          background-color: #8C1717;
     }
</style>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Arrays

Fazendo buscas em um array

- Para verificar se um determinando elemento está ou não no array, pode-se comparar o valor que deseja buscar com o valor de cada uma das posições no array.
  - Ao encontrar o valor desejado, a busca deve ser interrompida e um sinal de que o elemento foi encontrado pode ser definido.
  - Ao encerrar o laço, o sinal deve ser verificado para saber se o valor foi encontrado ou não.

HTML

```
<div id="listadealunos"></div>

<input type="text" id="busca"/><button type="button"
onclick="buscarAluno()">Grifinória</button><br/>
```

JAVASCRIPT

```
var grifinoria = [];
function selecionar(){/*...*/}

function buscarAluno(){
      var nome = document.getElementById("busca").value;
      var achou = false;
      for (i=0; i<grifinoria.length; i++){
            if (grifinoria[i] == nome){ achou = true; break; }
      }
      if (achou){ alert(nome + " está na Grifinória!"); }
      else{ alert(nome + " não está na Grifinória!"); }
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercícios

Crie páginas web utilizando HTML, JavaScript e CSS que resolva os seguintes problemas

1. Crie arquivo HTML contendo um array em Javascript, com o nome de 10 flores (orquídea, jasmim, violeta, etc). Procure na internet algumas imagens JPG e salve cada uma com o nome da flor.jpg, na mesma pasta em que está seu arquivo HTML. Em seguida, percorra o array e exiba cada uma das imagens, uma ao lado da outra, como mostrada abaixo. Defina um estilo no css de modo manter um tamanho padrão para todas as imagens.

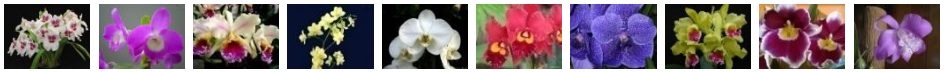
2. Crie uma página web contendo o mesmo array com nome de flores do exercício anterior. No HTML, crie um campo de texto e um botão. No campo de texto o usuário deverá digitar o nome de uma flor. Ao clicar no botão, você deverá informar se a flor que o usuário digitou está ou não está no seu array. Caso esteja, exiba a imagem correspondente à flor. Caso não esteja, exiba a mensagem “Essa flor ainda não está no meu catálogo”.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercícios

DESAFIO

Altere a sua galeria para, ao clicar em uma qualquer uma das imagem, mostrar a foto ampliada, como no exemplo abaixo.



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Estudo dirigido

- O objeto array possui alguns métodos próprios, implementados no Javascript.

1. concat()
2. every()
3. filter()
4. forEach()
5. indexOf()
6. lastIndexOf()
7. join()
8. map()
9. push() / pop()
10. reduce() / reduceRight()
11. reverse()
12. shift() / unshift()
13. slice()
14. some()
15. sort()
16. splice()
17. toLocaleString()
18. toString()

Cada aluno deve estudar um dos métodos e apresentar um exemplo de utilização na próxima aula

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

REFERÊNCIAS

- [1] W3C School. JavaScript Tutorial. Disponível em: http://www.w3schools.com/js/
- [2] MORISSON,Michael. Java Script – Use a Cabeça. Ed. 2. Rio de Janeiro: Altabooks
- [3] Manzano, José; Toledo, Suely. Guia de Orientação e Desenvolvimento de Sites – HTML, XHTML, CSS e JavaScript / Jscript. 2a. Edição

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br