Sistema FIEB SENAI

Federação das Indústrias do Estado da Bahia

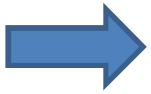
Introdução

- JavaScript
  - Linguagem criada inicialmente para validação de formulários no lado do cliente e interação com a página;
  - Fracamente tipada;
  - É interpratada;
  - Suporte a expressões regulares;

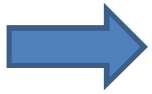
Introdução

•Diferença entre Compilação e Interpretação de Códigos

CÓDIGO FONTE ...
SOMA = VAR1 + TOTAL ...
(linguagem de programação)



TRADUTOR



CÓDIGO OBJETO ...
00010110111001011001011010 ...
("executável")

- MONTADOR (assembler):
  - Tradutor para linguagens de 2ª geração.
- COMPILADOR:
  - Traduz todo o programa de uma vez.
- INTERPRETADOR:
  - Traduz o programa instrução por instrução.
  - A verificação da existência de erros de sintaxe ou estrutura só ocorre em runtime e o código fonte do programa tem de ser sempre fornecido ao utilizador final.

História

- É uma linguagem inicialmente criada para o browser;
- Considerada como a mais popular do mundo;
- Criada em 1995 para o Netscape 2.0;
- A Microsoft adotou em 1996 no IE 3.0;

Ferramentas Necessárias

- Editor de Texto;
- Browser;
- Conexão com a internet;

Testando...

index.html

```
<html>
    <head>
        <title>Olá</title>
        <script type="text/javascript">
                alert("123 Testando");
        </script>

    </head>
    <body>
            <h1>Titulo</h1>

    </body>
</html>
```

Testando...

index.html

```
<html>
    <head>
        <title>Olá</title>
    </head>
    <body>
            <h1>Titulo</h1>
            <script type="text/javascript">
                alert("123 Testando");
            </script>

    </body>
</html>
```

Testando...

index.html

```
<html>
    <head>
        <title>Olá</title>
</head>
    <body>
            <h1>Título</h1>
             <script src="programa.js"> </script>

    </body>
</html>
```

programa.js

```
document.writeln('123 Testando');
```

Fonte: **JavaScript: The Good Parts [CROCKFORD,2008]**

Janelas de diálogo

- Alerta:
  - É uma janela onde mostra informações ao usuário;
  - alert(“Informacao”);

Janelas de diálogo

- Confirmação:
  - É uma janela que dá opção ao usuário clicar em OK ou Cancelar;
  - var resposta = confirm(“Pergunta”);

Janelas de diálogo

- Prompt:
  - É uma janela que dá opção ao usuário clicar em OK ou Cancelar. Além disso, ela permite ao usuário digitar uma entrada;
  - var resposta = prompt(“Pergunta”,”Sugestão”);

Exercícios

- Elabore scripts usando a função prompt que:
  - Leia um valor e imprima os resultados: “É maior que 10” ou “Não é maior que 10” ou ainda “É igual a 10”
  - Some dois valores lidos e imprima o resultado
  - Leia 2 valores e a operação a ser realizada (+, -, * ou /) e imprima o resultado (use um switch)
  - Leia um nome e um valor **n** e imprima o nome **n** vezes usando o laço **for**

Variáveis

- Var nomeDaVariavel = expressão;
- Exemplo:
  - var idade = 15;
  - var cor = “azul”;
- Nome das variáveis
  - Inicia-se com uma letra seguida, opcionalmente, de outras letras, números ou underlines;
    - c
    - c1
    - data_de_nascimento
    - DataDeNascimento

Tipos de Variáveis

- Booleano;
- Número;
- String;
- Arrays;

Booleano

- Exemplo:
  - var a = true;
  - var b = false;

Números

- Inteiros:
  - 3942
  - -3942
- Frações:
  - 7.34
- Exponenciais:
  - 3.14E10
- Hexadecimais
  - 0xf56a2

Operadores Aritméticos

- Soma -> **+**
- Subtração -> **-**
- Divisão -> ***/***
- Multiplicação -> *******
- Resto da Divisão -> **%**
- **++, --, +=, -=, *=, /=, %=**

Funções Matemática

abs	y = Math.abs(x);	Retorna o valor absoluto de um número (x no exemplo).
ceil	i = Math.ceil(x);	Retorna o menor inteiro maior ou igual ao número dado. Por exemplo: para x = 30.75 retorna 31 e para x = -30.75 retorna -30.
cos	y = Math.cos(x);	Retorna o co-seno de um número que representa um ângulo em radianos (x no exemplo). O resultado, conforme definição matemática da função, está na faixa de -1 a 1.
exp	y = Math.exp(x);	Retorna o número e (base dos logaritmos naturais) elevado ao argumento (x no exemplo).
floor	i = Math.floor(x);	Retorna o maior inteiro menor ou igual ao número dado. Por exemplo: para x = 30.75 retorna 30 e para x = -30.75 retorna -31.
log	y = Math.log(x);	Retorna o logaritmo natural (base e) de um número (x no exemplo). Se x = 0, retorna -Infinity. Se x < 0, retorna NaN porque está fora da faixa.
pow	p = Math.pow(x,y);	Retorna a base elevada ao expoente. No exemplo dado abaixo, x é a base e y é o expoente, isto é, p = xy.
sqrt	y = Math.sin(x);	Retorna o seno de um número que representa um ângulo em radianos (x no exemplo). O resultado, conforme definição matemática da função, está na faixa de -1 a 1.

Atividade 1

- Tendo como dados de entrada a altura de uma pessoa, construa um programa que calcule e mostre seu peso ideal, utilizando a seguinte fórmula:
  - peso ideal = (72.7*h) - 58

Atividade 2

- Calcule quantos galões de tinta são necessários e qual o preço para a realização de pinturas em tanques cilíndricos de combustível, onde a altura(h) e o raio(r) desse cilindro são dados como entrada.
  - Sabendo que:
    - O galão de tinta custa R$20,00;
    - cada galão contém 5 litros;
    - cada litro de tinta pinta 3 metros quadrados.
  - Sabendo que:
    - Área do cilindro=3,14*raio^2 + 2*3,14*raio*altura;

Operadores Relacionais

- > Maior
  - >= Maior ou igual
- < Menor
  - <= Menor ou igual
- == Igual
- != Diferente
- === Igual
- !== Diferente

Operadores Relacionais

- Diferença entre == e ===

```
var x = 43
x == 43 //retorna true
x == “43” //retorna true
x == 67 //retorna false
```

```
var x = 43
x === 43 //retorna true
x === “43” //retorna false
67 === “67” //retorna false
```

Operadores Lógicos

- **&&** e lógico (AND)
- **||** ou lógico (OR)
- **!** não lógico (not)

Operadores Lógicos

-

Tabela E	Tabela OU	Tabela NÃO
V e V → V	V ou V → V	Não V → F
V e F → F	V ou F → V	Não V → F
F e V → F	F ou V → V	
F e F → F	F ou F → F	

Comentários

- De linha
  - //Comentário
- De bloco
  - /* Comentário de uma ou mais linhas */

IF

– Exemplo:

```
var saldo = 10.0;
// Verificando se o saldo é suficiente
if (saldo < 0) {
   document.writeln (“Saldo insuficiente”);
} else {
   document.writeln (“Operação realizada”);
}
```

SWITCH

–Exemplo:

```
var farol = “azul”;
switch (farol){
        case “vermelho”:
                alert(“pare”);
                break;
        case “amarelo”:
                alert(“atencao”);
                break;
        case “verde”:
                alert(“Siga”);
                break;
        default:
```

FOR

–Exemplo:

```
var contador = 1;
for(i=0; i<contador; i++) {
  document.writeln(i);
}
var vetor = new Array();
vetor[“car”]=“carro”;
vetor[“house”]=“casa”;
vetor[“pen”]=“caneta”;
for(chave in vetor){
        alert(chave+“=>”+vetor[chave]);
}
```

While

–Exemplo:

```
var contador = 10;
// Contagem regressiva
while (contador >= 0) {
  document.writeln(contador);
  contador--;
}
```

Try/catch

```
•try{
        var x = u;
        var y = x+2;
        alert (y);
        }
   catch (e){
          alert(e.message+“\n” +e.name+
“\n”+e.toString());
        };
```

Strings

- São utilizadas para manipulação de textos;
- O JavaScript armazena as Strings como objetos do tipo String;

Strings

- Formas de criação de Strings em JavaScript
  - Aspas:
    - teste = “Tudo aqui é uma string”
  - Apóstrofos:
    - teste = ‘Tudo aqui é uma string’
  - new
    - teste = **new** String(“Tudo aqui é uma string”);

Strings

- Operações com Strings
  - x = “”;
  - x += “Olá ”;
  - x += “Mundo ”;
  - x += “Cheguei ”;
  - x += “hoje.”;
  - alert (x);

Strings

- Propriedades:
  - length
- Métodos:
  - indexOf(str);
    - Exemplo:
    - var str = “Olá, visite o SENAI.";
      - var n = str.indexOf(“SENAI");
      - n será igual a 14;

Strings

- lastIndexOf(str,offset);
  - Exemplo:
  - var str = “Seja bem vindo ao SENAI! Visite o SENAI CIMATEC.";
    - var n = str.indexOf(“SENAI");
- charAt(x)
  - Exemplo:
  - var str = “Seja bem vindo ao SENAI! Visite o SENAI CIMATEC.";
    - var n = str. charAt(1);

Strings

- toUpperCase();
- var str = “Visite o CIMATEC!";
  - var res = str.toUpperCase();
  - res será igual “VISITE O SENAI!”
- toLowerCase();
- var str = “Visite o CIMATEC!";
  - var res = str.toLowerCase();
  - res será igual “visite o cimatec!”

Strings

- anchor("name")
  - var str = "Visite o CIMATEC!";
  - Str.anchor("link");
- big()
  - var str = "Visite o CIMATEC!";
  - str.big();
- small()
  - var str = "Visite o CIMATEC!";
  - str. small();

Strings

- replace(a,b);
- var str = “Visite o CIMATEC!";
  - var res = str.replace("CIMATEC ", “SENAI");
  - res será igual “Visite o SENAI!”
- split(delimitador);
- var str = “Visite o CIMATEC!";
  - var res = str.split(“ ");
  - res será igual “Visite,o,CIMATEC!”
- substring(inicio, fim);
- var str = “Vamos verificar?";
  - var res = str.substring(1, 4);
  - res será igual “amo”

Strings

Caractere	Descrição
\n	Insere uma quebra de linha.
\t	Insere uma tabulação.
\r	Insere um retorno.
\f	Insere um caractere de barra.
\t	Tabulação.
\’	Apóstrofo.
\”	Aspas.
\\	Barra Invertida.
\XXX	Caractere representado pela codificação **Latin-1**. Exemplo **\251** representa o caractere de copyright ©.

OBS: As letras dos operadores devem apresentar-se em letras minúsculas.

Operandos

- 8 + 4 = 12 //número
- “8” + “4” = “84” // string
- “8.56” + 4 = “8.564” // string
- “8” * 4 = 32 //número
- 8 / “4” = 2 //número
- 8 + true = 9 //número
- “olá” + true = olátrue //string
- 8 * “olá” = //NaN não é número
- 8 + “olá” = “8olá” //string

Arrays

- Trata-se de automatizar a declaração de um grande número de dados;
- As variáveis assim declaradas se acessam através de um índice.

Arrays

- Vazio:
  - [ ]
  - New Array();
- Não vazio:
  - [‘João´, ´Maria´]
- Exemplo 1:
  - var carros = new Array(2);

carros[0]= “Ferrari”;
carros[1]= “BMW”;

Arrays

- Exemplo 2:
  - var carros = new Array(“Ferrari”, “BMW”);
- Curiosidade:
  - nome = “João da Silva Santos”
  - vetor = nome.split(“ “);
  - total = vetor.join(“ “);

Date

- É um objeto que facilita a manipulação com horas e datas em JavaScript;
- Os valores do mês são contados de 0 até 11 e os dias da semana de 0 a 6 da seguinte forma
- Instanciação:
  - variavel = new Data();
  - variavel = new Data(1988,11,30);
  - variavel = new Data("Nov 30,1988 20:10:10");

Date

MÉTODOS	DESCRIÇÃO
getDate	Dia da semana (0=Domingo).
getDay	Dia do mês.
getHours	Horas (0 a 23).
getMinutes	Minutos (0 a 59).
getMonth	Mês do ano (0=janeiro).
getSeconds	Segundos (0 a 59).
getTime	Milissegundos desde 1 de janeiro de 1990 (00:00:00).
getTimezoneOffset	Diferença de fuso horário em minutos para o GMT.
getYear	2 digitos do ano até 1999. Após 2000, 4 dígitos.

Date

- parse();
  - retorna o valor de milissegundos a partir de 1 de janeiro de 1970, 00:00:00
- toGMTString()
  - Converte um objeto Date para uma string usando convenções GMT;

Funções

- São blocos de código reutilizados;
- São executadas apenas quando chamadas;
- Podem ter parâmetros de entrada e de saída
- Podemos ter vários parâmetros de entrada separados por vírgulas
- Podemos retornar valor através da instrução **return**

Funções

```
•function maiuscula (objeto){
        objeto.value = obj.value.toUpperCase();
}
•function mes_extenso (mes){
var extenso = "";
switch (mes){
        case 1: extenso="janeiro";break;
        case 2: extenso="fevereiro";break;
        case 3: extenso="marco";break;
        case 4: extenso="abril";break;
        case 5: extenso="maio";break;
        case 6: extenso="junho";break;
        case 7: extenso="julho";break;
        case 8: extenso="agosto";break;
        case 9: extenso="setembro";break;
        case 10: extenso="outrubro";break;
        case 11: extenso="novembro";break;
        case 12: extenso="dezembro";break;
}
return extenso;
}
```

Funções

```
•function mensagem(){
        for (i=0;i<mensagem.arguments.length;i++){
                alert(mensagem.arguments[i]);
        }
•}
```

Funções

- O JS possui funções internas, que não estão vinculadas a nenhum objeto;
- São elas:
  - toFixed(casasDecimais)
    - Exemplo:
    - var num = 5.56789;
      - var n = num.toFixed(2);
      - Resultado => 5.57
  - isNaN(número)
    - Exemplos:
      - var b = isNaN(-1.23) => **false**
      - var c = isNaN("Olá") => **true**

Funções

- parseInt(string, radix);
  - Converte strings ou números nas bases binário, octal ou hexadecimal para valores decimais;
  - Em caso da função encontrar valores diferentes de +, - , números de 0 a 9, ponto decimal ou expoente, será retornado o valor até o ponto desse caractere, o resto será ignorado;
  - Em caso do primeiro caractere não puder ser convertido é retornado NaN;

Funções

- setTimeout("função",intervalo);
  - Exemplo:
  - var num = setTimeout(funcao,4000);
- clearTimeout();
  - Exemplo:
    - clearTimeout(num);

Funções

- setInterval(“função”,intervalo);
  - Exemplo:
  - var num = setInterval (funcao,4000);
- clearInterval();
  - Exemplo:
    - clearInterval(num);

Eventos

- São reações a ações do usuário ou da própria página

ou:

- São ocorrências ou acontecimentos dentro de uma página. Ex:
  - Carregar uma página;
  - Clicar em um botão;
  - Modificar o texto de uma caixa de texto;
  - Sair de um campo texto;
  - etc;

Eventos

- Ex. : Passando parâmetros

```
...
<form>
        <input type = “button" value = "Chamar função"
                onclick = “saudacao('jose');"/>
</form>
...
```

saudacao.html

```
function saudacao(nome) {
        alert("Olá, " + nome);
}
```

saudacao.js

Eventos

- Ex. 4: Passando parâmetros de campos de formulário

```
...
<form>
        <input type="text" name="txtNome" id = "txtNome"/>
        <input type="button" name="btn_saudacao"
            onclick = “saudacao(document.getElementById('txtNome').value);"/>
</form>
...
```

```
...
<form name = "frm">
        <input type="text" name="txtNome"/>
        <input type="button" name="btn_saudacao"
            onclick = “saudacao(txtNome.value);"/>
</form>
...
```

Eventos

- **onclick**: ocorre quando o usuário clica sobre algum elemento da página
  - Ex.: Chamar a função
- **onload** e **onunload**: ocorrem respectivamente quando o objeto que as possuem são carregados (criados) e descarregados
  - Ex.: <body onload = "bemvindo();" onunload = "adeus();">

Eventos

- **onmouseover**: é acionado quando o mouse se localiza na área de um elemento
- **onmouseout**: ocorre quando o mouse sai da área de um elemento
- Ex.: <a href = "#" onmouseover = "mouseSobre();"
  - onmouseout = “mouseFora();">
  Passe o mouse

Eventos

- **onfocus**: ocorre quando um controle recebe o foco através do mouse ou do teclado
- **onblur**: ocorre quando um controle perde o foco

```
...
<input type="text" name="txt1" id = "txt1"
                onfocus = "trataEntrada('txt1')"
                onblur = "trataSaida('txt1')"/>
<input type="text" name="txt2" id = "txt2"
                onfocus = "trataEntrada('txt2')"
                onblur = "trataSaida('txt2')"/>
...
```

Eventos

- **onkeydown** e **onkeypress**: são semelhantes e ocorrem quando uma tecla é pressionada pelo usuário em seu teclado.
- **onkeyup**: é executado quando a tecla é liberada, ou seja, ela foi pressionada e em seguida liberada.
- ...

```
<input type="text" name="txtOrigem" id = "txtOrigem"
                onkeydown = "copiaTexto()"/>
<input type="text" name="txtDestino" id = "txtDestino" />
        ...
```

Eventos

```
function mouseSobre() {
        var divResultado = document.getElementById("resultado");
        divResultado.innerHTML = divResultado.innerHTML +

        "mouse sobre.<br/>";
}

function mouseFora() {
         var divResultado = document.getElementById("resultado");
        divResultado.innerHTML = divResultado.innerHTML +

        "mouse fora.<br/>";
}
```

Eventos

- **onAbort**: é executado quando o usuário cancela o carregamento de uma imagem.

- ...

```
<IMG name="foto" src = "foto.jpeg"                    onAbort
= "alert('Foto não carregada!')"/>
        ...
```

Eventos

- **onsubmit**: usado para chamar a validação de um formulário (ao enviar os dados)
- Para validar um formulário, chamamos uma função por nós definida:
  - Ao chamar a função, usamos a palavra reservada return
- A função, por sua vez, deve retornar true ou false, representando se os dados devem ou não serem enviados. Ex:

```
<form name="frmBusca"
        action="http://www.google.com/search"
    method="get" onsubmit = "return validaCampo()">
        Termo: <input type="text" name="q" id = "q" />
        <input type="submit" name="btnBuscar" value="Buscar"/>
</form>
```

Eventos

```
function validaCampo() {
        var valor =
document.getElementById("q").value;

        if ((valor == null) || (valor == "")) {
                        alert("Preencha o campo de
busca");
                        return  false;
        }
        return true;
}
```

Eventos

```
function trataEntrada(id) {
        var div = document.getElementById("resultado");
        div.innerHTML = div.innerHTML + id +" ganhou o
foco.<br/>";
}

function trataSaida(id) {
        var div = document.getElementById("resultado");
        div.innerHTML = div.innerHTML + id +" perdeu o
foco.<br/>";
}
```

Eventos

```
function copiaTexto(idOrigem,idDestino) {
        var txtOrigem = document.getElementById(idOrigem);
        document.getElementById(idDestino).value =
                        txtOrigem.value;
}
```

Eventos

- Ex. :

```
...
<form>
        <input type = “button" value = "Chamar função" onclick = "alo();"/>
</form>
...
```

alomundo.html

```
function alo() {
        alert("Link clicado!");
}
```

alomundo.js

Eventos

EVENTO	MANIPULADOR	DESCRIÇÃO
blur	onBlur	Ocorre quando o usuário retira o foco de um objeto de formulário.
change	onChange	Ocorre quando o usuário muda o valor de um objeto de formulário.
click	onClick	Ocorre quando o usuário clica sobre o objeto.
focus	onFocus	Ocorre quando o usuário focaliza o objeto.
load	onLoad	Ocorre quando o usuário carrega a página.
unload	onUnload	Ocorre quando o usuário abandona a página.
mouseOver	onMouseOver	Ocorre quando o ponteiro do mouse passa sobre um link ou âncora. **Válidos apenas para hiperlinks**.
select	onSelect	Ocorre quando o usuário seleciona um elemento de um formulário.

Eventos

EVENTO	MANIPULADOR	DESCRIÇÃO
submit	onSubmit	Ocorre quando o usuário envia um formulário.
mouseDown	onMouseDown	Ocorre quando o botão do mouse é pressionado.
mouseMove	onMouseMove	Ocorre quando o ponteiro do mouse se movimenta sobre o objeto.
mouseOut	onMouseOut	Ocorre quando o ponteiro do mouse afasta de um objeto. **Válidos apenas para hiperlinks**.
mouseUp	onMouseUp	Ocorre quando o botão do mouse é solto.
keyDown	onKeyDown	Ocorre quando uma tecla é segurada.
keyPress	onKeyPress	Ocorre quando uma tecla é pressionada.
keyUp	onKeyUp	Ocorre quando uma tecla é solta.

Prática

1.Elabore um formulário HTML que tenha como entrada 3 valores para lados de um triângulo e escreva uma função de nome **tipoTriangulo** que receba 3 parâmetros esses lados de um triângulo e imprima o tipo dele (equilátero, isósceles ou escaleno).

A passagem dos parâmetros deve ser feita de forma simplificada dentro do HTML no evento onclick de um botão ou link da seguinte forma:

Prática

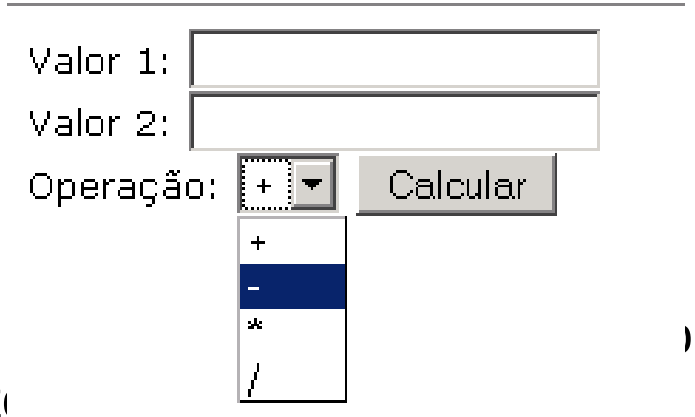
2. Deseja-se calcular a conta de consumo de energia elétrica de uma casa. Para isso, elabore um formulário em HTML que leia a quantidade de Kwh consumidos e o valor unitário do Kwh.

Escreva uma função em JavaScript que faça o cálculo (valor = quantidade x valor unitário) e, caso a quantidade de Kwh ultrapasse 100, o valor do Kwh deve ser acrescido em 25%. Caso ultrapasse 200, o mesmo valor deve ser acrescido em 50%.

Os valores devem ser repassados para para uma função em JavaScript

Prática

•Cria uma página semelhante à figura abaixo e implemente em JS uma calculadora com as 4 operações fundamentais



•O valor da caixa select ...rma que se obtém o valor das caixas de text...

•O resultado do cálculo deve ser exibido com uma função alert

•*Use a função parseFloat para converter números reais e caso não seja um número enviar um alerta*

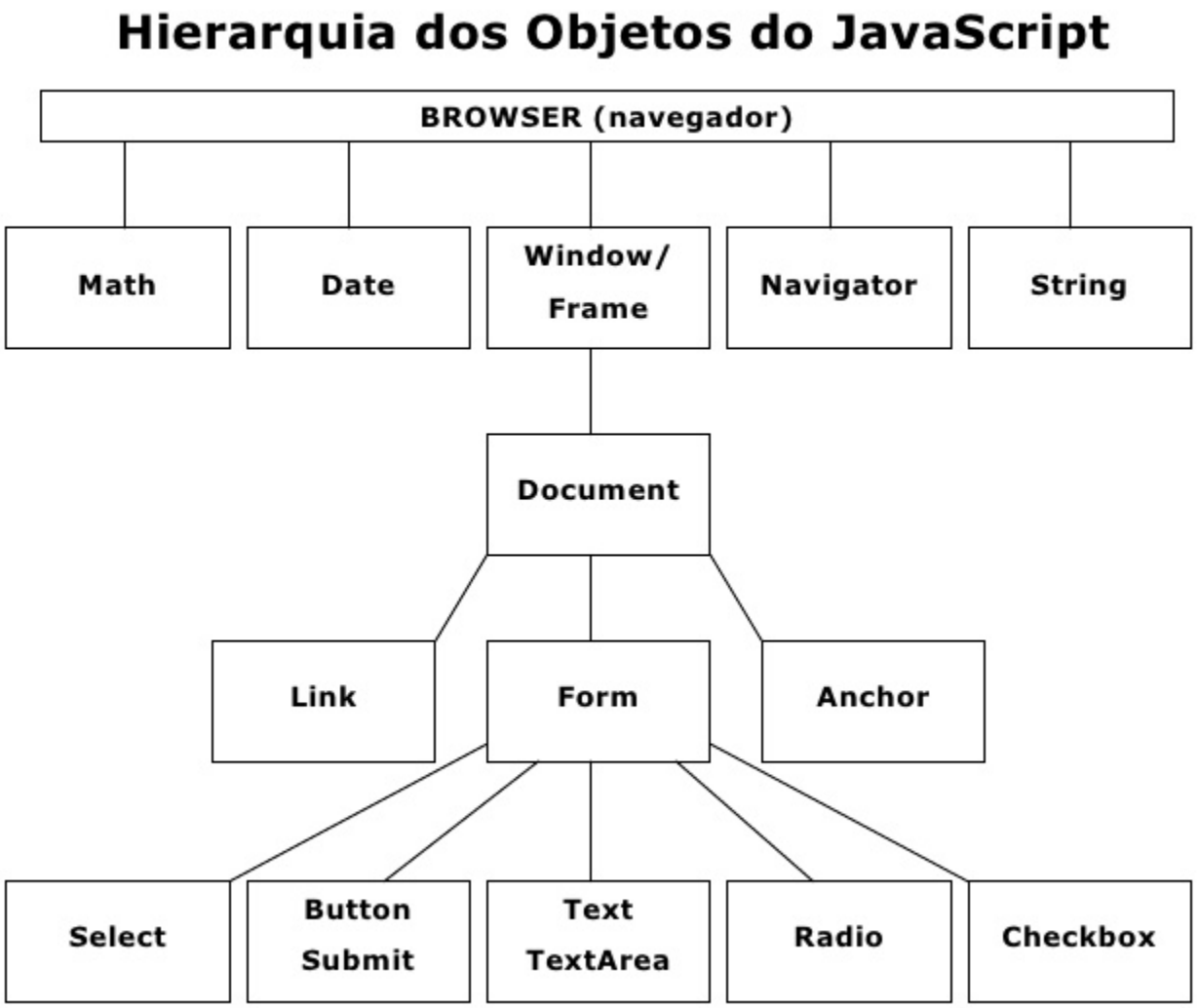
Prática

Crie um relógio em JS que mostre em uma página HTML a data e hora atual;

DOM

- O DOM (Modelo de Objeto de Documentos) é uma API para definição da estrutura lógica dos documentos HTML e/ou XML e o meio pelo qual o documento é acessado;
- Cada elemento do HTML é representado no DOM por um objeto;
- O DOM possibilita o JavaScript:
  - Acessar individualmente elementos do HTML;
  - Mudar elementos HTML individualmente sem necessitar carregar a página do navegador;

DOM


Hierarquia dos Objetos do JavaScript
BROWSER (navegador)
Math
Date
Window/
Frame
Navigator
String
Document
Link
Form
Anchor
Select
Button
Submit
Text
TextArea
Radio
Checkbox

DOM

- Principais métodos para manipualção:
  - getElementById
    - Exemplo:
      - document.getElementById(“id”).innerHTML;
  - getElementsByTagName
    - Exemplo:
      - valores = document.getElementsByTagName('ul')[0].getElementsByTagName('li');
  - getElementsByName
    - Exemplos
      - valores = document.getElementsByName(‘atributoname');
  - write()
    - Escreve um texto HTML no ponto da página onde é chamado;

DOM

Propriedade	Descrição
documentElement	Captura o elemento raiz <html> de um documento HTML.
createElement	Cria um nodo elemento na página.
createAttribute	Cria um nodo atributo na página.
createTextNode	Cria um nodo texto na página.
appendChild	Insere um novo elemento filho.
removeChild	Remove um elemento filho.
parentNode	Retorna o nodo pai de um nodo.

Validações de formulários

- Os dados de um formulário devem ser enviados para um servidor.
- Pode-se suavizar o trabalho de um servidor efetuando-se algumas validações no próprio cliente (navegador) com JavaScript
  - Nota:

    É importante também haver a validação no servidor.

    A validação com JavaScript serve apenas para amenizar o tráfego de rede com validações simples como campos não preenchidos, caixas não marcadas e etc.

Validações de formulários

- Algumas dicas:
  - Ao se validar um campo, procure sempre obtê-los pelo atributo id
  - Quase todos os elementos do formulário possuem sempre um atributo **value**, que pode ser acessado como uma String
  - Para verificar um caractere em especial dentro de um valor, use **[ ]**, pois as Strings são arrays de caracteres
  - As Strings também possuem um atributo **length** que assim como os arrays, representam o tamanho

Validações de formulários

- Alguns exemplos de validação:
  - Campos de texto não preenchidos
  - Campo de texto com tamanho mínimo e máximo
  - Validação de campo de e-mail
  - Campos com apenas números em seu conteúdo
  - Seleção obrigatória de radio buttons, checkboxes e caixas de seleção

Validações de formulários

- Validação de campo de texto com preenchimento obrigatório:
  - Deve-se validar se:
    - O valor é nulo
    - O valor é uma String vazia
    - O valor é formado apenas por espaço
  - A validação feita para um campo do tipo **text** serve também para um **textarea** e para um **password**
  - Validar espaços pode ser feito usando expressões regulares

Validações de formulários

•Validação de campo de texto com preenchimento obrigatório:

```
function validaCampoTexto(id) {
        var valor = document.getElementById(id).value;
//testa se o valor é nulo, vazio ou formado por apenas espaços em branco
if ( (valor == null) || (valor == "") || (/^\s+$/.test(valor)) ) {
        return false;
}
return true;
}
```

Validações de formulários

- Validação de tamanho em campos de texto:
  - É importante validar primeiramente se o campo tem algo preenchido (validação anterior)
  - Pode-se limitar o campo a um tamanho mínimo ou máximo
  - Usa-se o atributo **length** para se checar o tamanho do campo valor do componente do formulário

Validações de formulários

Validação de tamanho em campos de texto:

```
function validaCampoTextoTamanho(id, minimo, maximo) {
        var valor = document.getElementById(id).value;
        if (!validaCampoTexto(id)) {
                return false;
        }

        if ( (valor.length < minimo) || (valor.length > maximo)) {
                return false;
        }
        return true;
}
```

Validações de formulários

- Validar para que um campo tenha apenas números:
  - Pode-se validar um campo que deva ser numérico usando-se a função isNaN que retorna verdadeiro se um parâmetro não é um número
  - Também é aconselhável validar se o campo contém algum valor

Validações de formulários

- Validar para que um campo tenha apenas números:

```
function validaCampoNumerico(id) {
        var valor = document.getElementById(id).value;
        if (isNaN(valor) ) {
                return false;
        }
        return true;
}
```

Validações de formulários

- Validar se um item foi selecionado numa caixa de seleção ou combo box:
  - Deve-se obter o índice do elemento selecionado através do atributo **selectedIndex**
  - **selectedIndex**: começa do 0 e tem o valor -1 se não houver seleção
  - O índice pode ser nulo se o componente não for do tipo select

Validações de formulários

- Validar se um item foi selecionado numa caixa de seleção ou combo box

```
function validaCampoSelect(id) {
        var indice = document.getElementById(id).selectedIndex;
        if ( (indice == null) || (indice < 0) ) {
                return false;
        }
        return true;
}
```

Validações de formulários

- Validar se uma caixa de checagem (checkbox) está marcada:
  - Deve-se consultar o atributo **checked** do componente

```
function validaCampoCheckbox(id) {
        var elemento = document.getElementById(id);
        if (!elemento.checked) {
                return false;
        }
        return true;
}
```

Validações de formulários

- Validar se pelo menos um botão de radio de um conjunto foi selecionado:
  - Os campos radio funcionam em conjunto desde que possuam o mesmo atributo name, portanto não se deve consultar pelo id e sim pelo nome pelo método:
    - document.getElementsByName(nome);
  - **getElementsByName(nome)** retorna um array de elementos com o mesmo nome.
  - Esse array deve ser percorrido verificando-se no atributo **checked** se pelo menos um dos botões de radio foram marcados

Validações de formulários

•Validar se pelo menos um botão de radio de um conjunto foi selecionado:

```
function validaCamposRadio(nome) {
        var opcoes = document.getElementsByName(nome);
        var selecionado = false;
        for(var i = 0; i < opcoes.length; i++) {
                if(opcoes[i].checked) {
                        selecionado = true;
                        break;
                }
        }
        if(!selecionado) {
                return false;
        }
        return true;
}
```

Atividade

- Nas atividades seguintes:
  - Use uma página HTML e um arquivo de scripts
  - Use o evento **onsubmit** do formulário e uma função de validação que retorne **true** ou **false**
  - Utilize uma página qualquer como **action** do formulário.

Atividade

- Copie o valor de um campo texto para outro caso o campo de origem não esteja vazio. Use o evento onblur do campo de origem
- Valide um campo senha de acordo com seu tamanho:
  - < 3: segurança fraca
  - Entre 3 e 5: segurança média
  - >= 6: segurança forte
- Valide se dois campos do tipo **password** são idênticos
- Valide 3 campos texto que representem dia, mês e ano:
  - Dia: entre 1 e 31
  - Mês: entre 1 e 12
  - Ano: > 1949