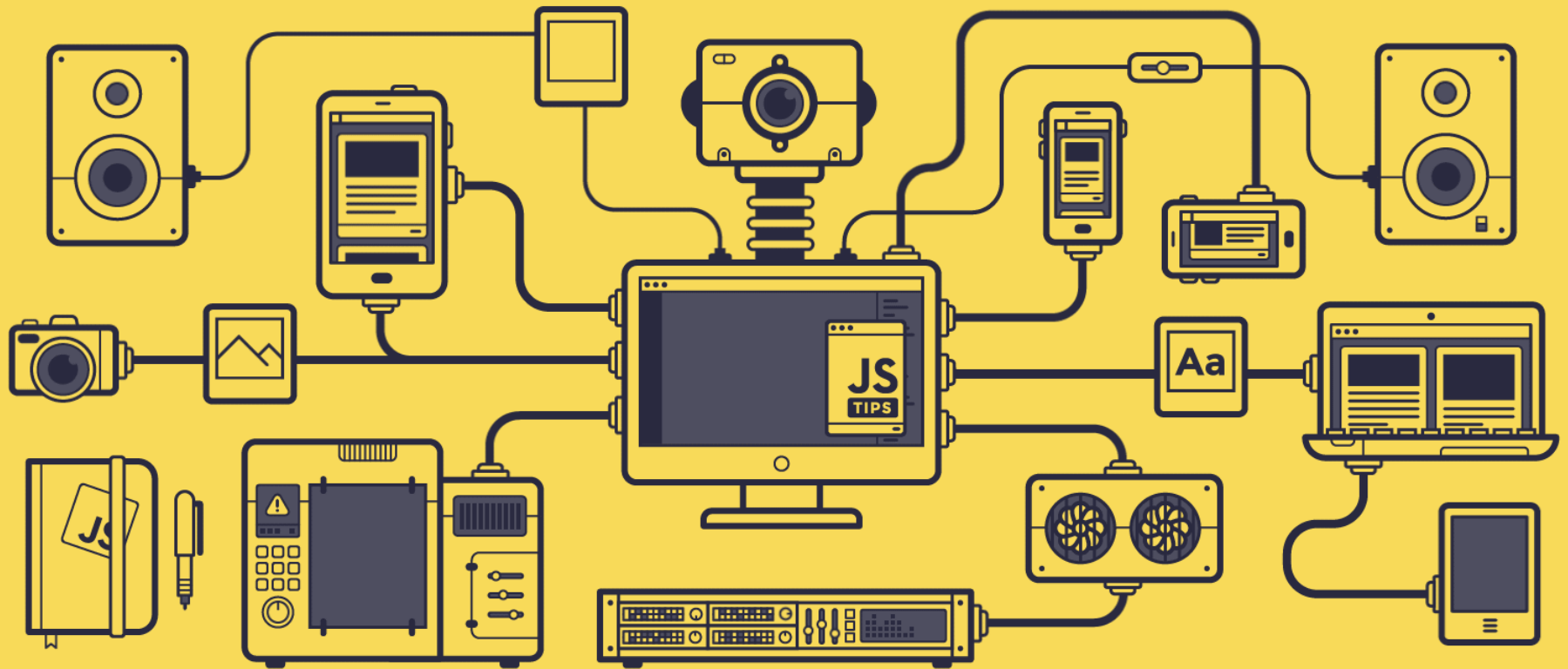


Desenvolvendo Aplicações WEB



Introdução JavaScript

JavaScript

- JavaScript é uma linguagem interpretada preparada para execução em Browser.



CRIANDO SCRIPT

JS INTERNO

Pode ser escrito antes do body ou depois, lembrando que script interno, primeiro carrega o js e depois o body, deixando o carregamento mais lento. Sendo a melhor opção o EXTERNO

```
<script>  
  seu script aqui  
</script>
```

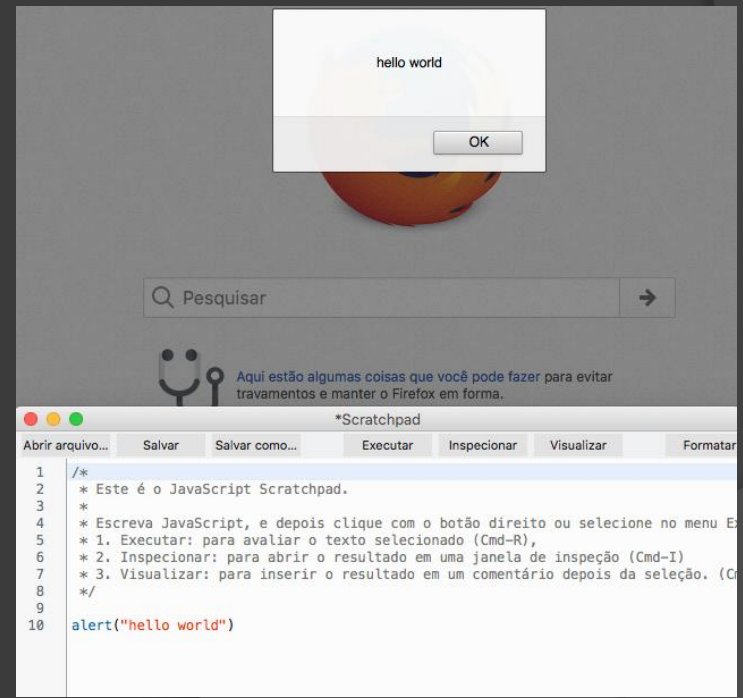
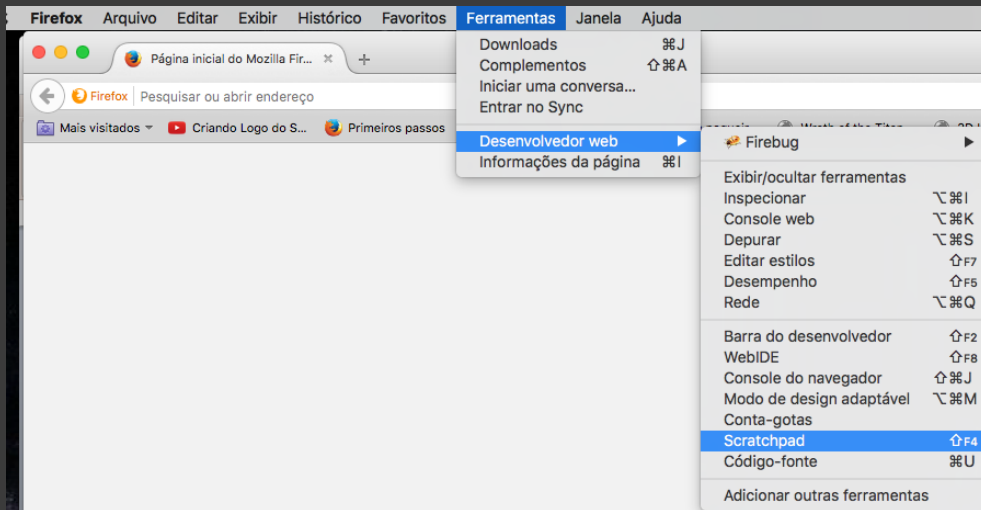
JS EXTERNO

Pode ser carregado dentro do head ou depois do Body

```
<script type="text/javascript" src="script.js"></script>
```

ONDE TESTAR E DEPURAR?

FIREFOX




ONDE TESTAR E DEPURAR?

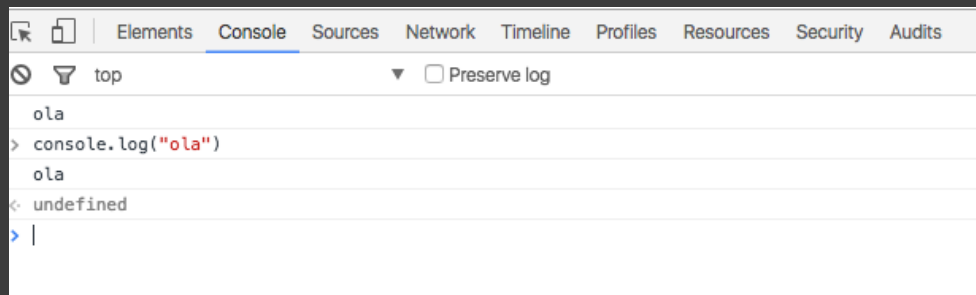
Google DevTools

Accessing the DevTools

To access the DevTools, open a web page or web app in Google Chrome. Either:

- Select the **Chrome menu**  at the top-right of your browser window, then select **Tools > Developer Tools**.
- Right-click on any page element and select **Inspect Element**.

CONSOLE



Alert – abre caixa de dialogo

Console – exibe a informação sem ter a necessidade de abrir a janela de dialogo

```
console.log ("hello word")  
alert ("ola mundo")
```

Comentário

```
// isto é um comentário
```

Exemplo - Alert

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Aula01-JS</title>
5      <meta charset="UTF-8">
6  </head>
7  <body>
8  <script>
9
10      alert ("Hello World");
11
12
13 </script>
14 </body>
15 </html>
```

Exemplo - Confirm

```
confirm("isto é um confirma!")
```

```
confirm(message: string) -> bool
```


Usando window.prompt

```
<script type="text/javascript">  
    var nome = window.prompt("Qual o seu nome?");  
    alert ("Prazer!!!!");  
</script>
```

Exemplos – document.write

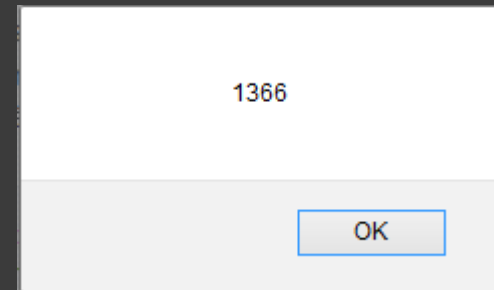
- Inserindo textos – vai introduzi-los direto no Corpo da página - html

```
document.write("Texto inserido com instruções JavaScript");
```

Exemplos –window.inner

- Largura e Altura do Browser (window)

```
<script type="text/javascript">  
var Largura = window.innerWidth;  
alert(Largura);  
</script>
```



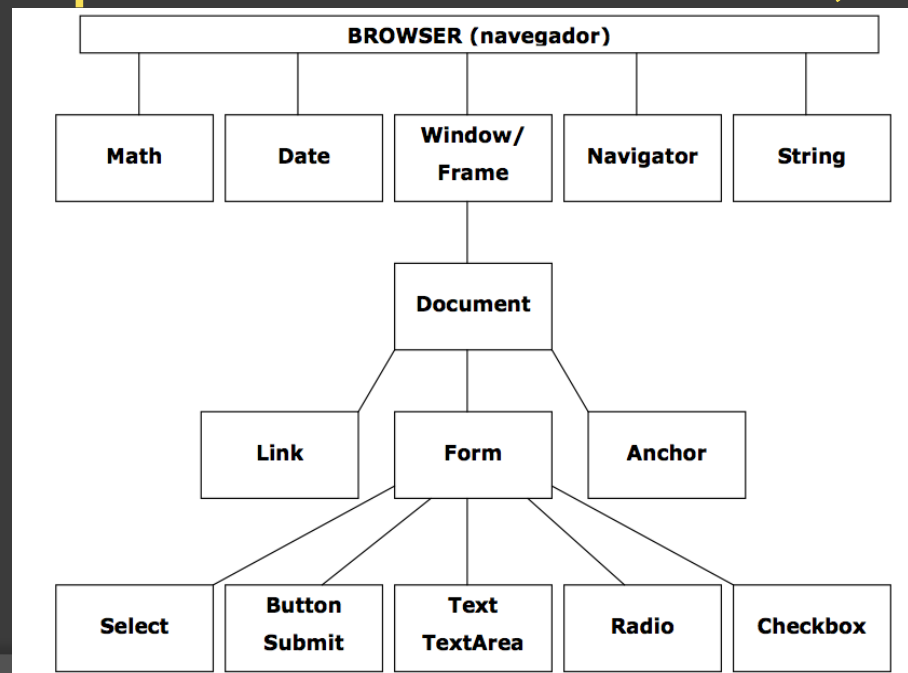
```
<script type="text/javascript">  
var altura = window.innerHeight;  
alert(altura);  
</script>
```

ORIENTAÇÃO A OBJETOS

- A linguagem JavaScript corresponde a programação orientada a objetos;
- Isto significa que todos os elementos de uma página da Web são tratados como objetos.

MANIPULAÇÃO DE OBJETO

- Os objetos são representados por uma hierarquia, fazendo com que alguns objetos se tornem propriedades de outros;



PROPRIEDADES DE OBJETOS

- A utilização de propriedades se dá acompanhada de seu objeto sendo separados por um ponto apenas.

nomeObjeto.propriedade

EX:

```
document.getElementById("texto")
```

Quando uma função parte de um objeto tem que ter a referência do objeto e passar um parâmetro

Método

MÉTODOS DE OBJETOS

- Na maioria das vezes os métodos são usados para alterar o valor de uma propriedade ou executar uma tarefa específica.

`nomeObjeto.método(argumento)`

DOM – Document Object Model

- Os *documentos* html assim como os seus *elementos* são representados por objeto que representa o documento html através da variável *document*

DOM – getElementById

- Podemos buscar elementos do HTML por ID pelo método `getElementById`, ele devolve null se não existir elemento html como indentificador.

```
var elemento = window.document.getElementById('
    conteudo');
;

</script>
<body>
    <div id="conteudo">teste</div>
```

DOM – getElementByIdName

- Podemos buscar elementos do HTML pelo valor do atributo NAME (nos formulários) pelo método getElementByName

```
var elemento = window.document.getElementsByName('categoria');
```

```
<button onclick="getElementById('demo').innerHTML=Date()">The time is?</button>
<p id="demo"></p>
```

DOM – `getElementsByTagName`

- Podemos buscar elementos do HTML de m determinado tipo através do método `getElementsByTagName`

```
var elemento = window.document.getElementsByTagName('p');
```

DOM – `getElementsByClassName`

- Podemos buscar elementos do HTML pelo valor do atributo CLASS através do método `getElementsByClassName`

```
var array = document.getElementsByClassName('confirmado');
```

EXERCICIO

- Baseado nesse modelo, crie o seu

```
<body>
  <h1>Aperte o botão e veja!</h1>
  <p id="demo"> JS troca texto HTML</p>
  <button type="button" onclick="document.getElementById('demo').
    innerHTML='Hello!'">
    Click Me!</button>
</body>
```

EXERCICIO

- Baseado nesse modelo, crie o seu

```
<script>
function ligar () {
    window.document.getElementById("imagem").src="img/js1.png";
    pegar o elemento pela ID e troca o endereço de imagem//
}
</script>
</head>
<body>
    
```

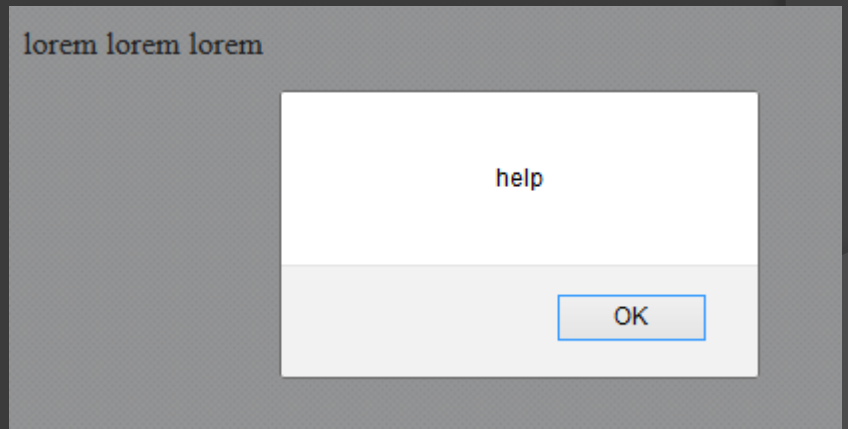
Interatividade entre o mouse, digitação do usuário, carregamento da página, redimensionamento da mesma entre outros - lista de vários eventos: <https://www.w3.org/TR/2013/WD-DOM-Level-3-Events-20131105/>

Eventos

EVENTO

- Ao clicar no parágrafo a palavra Help será exibida no console do navegador

```
<p onclick="alert('help')">  
  lorem lorem lorem  
</p>
```



EVENTOS com Função

- Em linguagens orientadas a objetos é comum a manipulação de eventos que é qualquer reação ou ação que executará determinado procedimento.

<TAG nomeEvento="Instruções JavaScript">

```
<button onclick="funcao()">Click Me</button>
```

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.

EVENTOS

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.

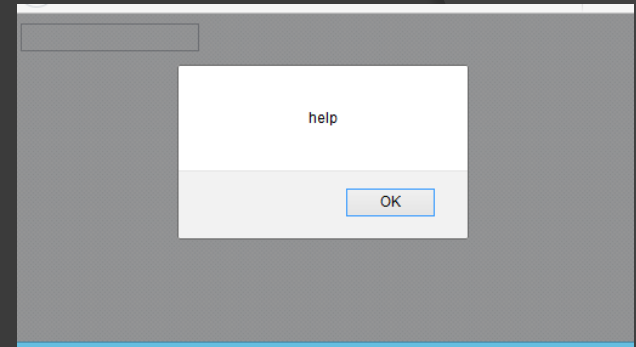
EVENTOS

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.

EVENTOS

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.

EVENTOS



```
<body onload="alert('Exemplo')">
```

```
<!-- ao fazer resize na janela exibe a caixa de dialogo -->  
<body onresize="alert('Exemplo')">
```

```
<!-- disparado quando a barra de rolagem é acionada podendo ser  
usada em elementos fo html: section, div, textarea -->  
<body onscroll="alert('Exemplo')">
```

```
<!-- disparado quando um determinado elemento ganha foco  
poder ser usado nas tags: input, select, textarea (formularios) -->  
<body>  
    <input onfocus="alert('help')"...>  
</body>
```

EVENTOS

```
<input onchange="alert('blur')"...>  
<input onblur="alert('blur')"...>  
<input onselect="alert('blur')"...>  
<form onsubmit="alert('blur')"...>  
<form onreset="alert('resetando')"...>
```

```
<!-- ocorre quando alguém clica na seguinte ordem a seguir  
pode ser usado nas tags: p, div, img, table -->
```

```
<p onmousedown="alert('1')"  
    onmouseup="alert('2')"  
    onclick="alert('3')">
```

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do  
eiusmod  
</p>
```


EVENTOS

```
<p onclick="alert('1')">click duplo</p>
<p onmousemove="alert('1')">movimentar mouse sobre elemento</p>
<p onmouseover="alert('1')">ponteiro do mouse sobre elemento</p>
<p onmouseout="alert('1')">ponteiro do mouse deixa de estar sobre</p>
```

```
<!-- usado nas tags input e textare, quando a tecla enter é pressionada -->
<input onkeydown="alert('1')"
      onkeypress="alert('2')"
      onkeyup="alert('3')"
>
```

São objetos, você pode armazená-los em variáveis, arrays e outros objetos.

Funções

Criando Funções

```
<script>
//escrevendo uma funcao com parametro
function mult (x,y){
    return (x*y);
}
//armazenando na variavel a funcao
var mult = function (x,y){
    return (x*y);
}
</script>
```

```
function dividir(x,y){
    alert(x/y);
}
dividir(6,2); |
```

Criando Funções – com onclick

```
<script>
function PrimeiraFuncao()
{

alert ("Isto é uma função JavaScript");

}

</script>

</head>

<body>
<input name="cliqueaqui" type="button" value="Clique aqui" onclick="
PrimeiraFuncao()" />
```

Função – com onload

```
1  <HTML>
2  <HEAD>
3      <TITLE>Manipuladores de Eventos</TITLE>
4  </HEAD>
5  <body onload="funcao()">
6
7  <script>
8      function funcao(){
9          alert ("Pagina com ONLOAD")
10     }
11 </script>
12 </body>
13 </HTML>
14
```

VARIÁVEIS

- Um nome que podemos aplicar, um valor para um recipiente. O conteúdo pode ser atribuído ou vir de um resultado de uma dada expressão ou função (como já vimos).

```
var nome = "jonas"; //string
var idade = 31; // numero inteiro/fracionario
var horas = 20.05;
var casado = true; // booleano (v/f)
```

- Ponto flutuante 2.34 é mult por 10 à quarta potência, ou $2.34 * 10000$

VARIÁVEIS

- **Variável Global** : MinhaVariavel = ""
- **Variável Local** : var MinhaVariavel = ""
- BOOLEANOS – valores lógicos, sendo TRUE (1) e FALSE (2)
- STRING – dentro das aspas , até mesmo números, ex: “2”.
- No próximo slide, tabela com caracteres especiais.

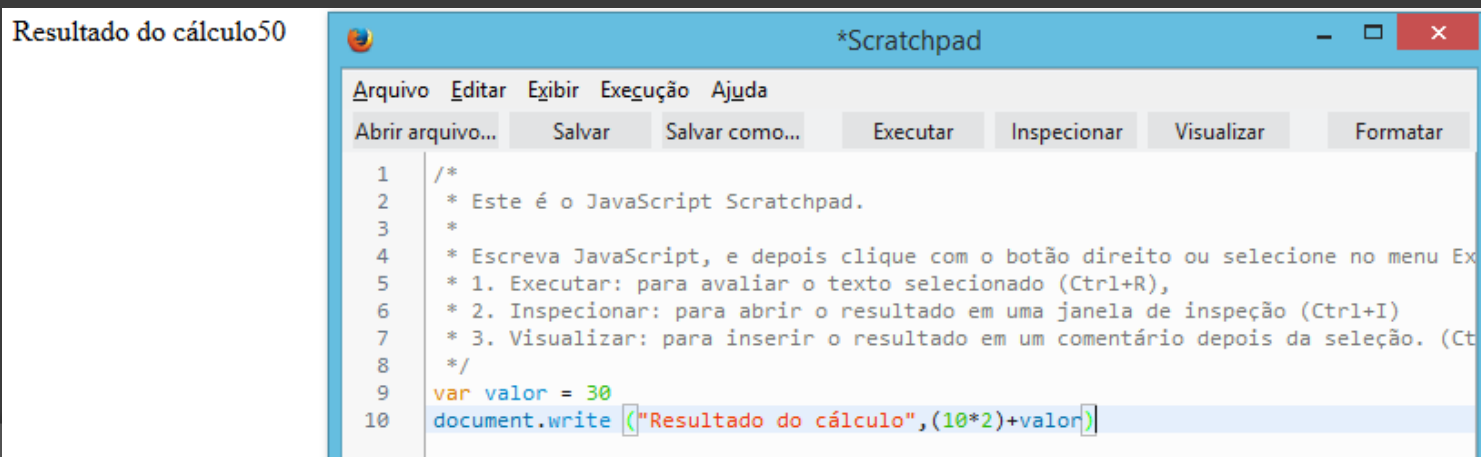
Sequência de escape	Significado	Categoria
\b	Backspace	
\t	Tabulação	Espaço em branco
\n	Alimentação de linha (nova linha)	Terminador de linha
\v (Consulte a observação após esta tabela.)	Tabulação vertical	Espaço em branco
\f	Avanço de página	Espaço em branco
\r	Retorno de carro	Terminador de linha
	Espaço	Espaço em branco
\"	Aspas duplas (")	
\'	Aspas simples (')	
\\	Barra invertida (\)	

EXPRESÕES

- É a combinação de variáveis, literais, métodos, funções, operadores que retornam qualquer valor

OPERADORES

```
aritmético : + - * / % (mod)
atribuição: = += -= *= /= %= ++ --
relacional: == != < <= > >=
lógico && ||
```



OPERADORES DE ATRIBUIÇÃO

Operadores de atribuição

[↑Topo](#) • [Fim↓](#)

Operador	Descrição	Exemplo(s)
=	Atribui o valor do operando esquerdo ao operando direito.	<code>x = 3;</code> <code>a = b + c;</code>
+=	Soma 2 valores e atribui o resultado ao primeiro valor.	<code>x += 3;</code> Se x era 1, passa para 4.
-=	Subtrai 2 valores e atribui o resultado ao primeiro.	<code>x -= 3;</code> Se x era 1, passa para -2.
*=	Multiplica 2 valores e atribui o resultado ao primeiro.	<code>x *= 2;</code> Se x era 4, passa para 8.
/=	Divide 2 valores e atribui o resultado ao primeiro.	<code>x /= 2;</code> Se x era 4, passa para 2.
%=	Calcula o resto da divisão de 2 valores e atribui o resultado ao primeiro.	<code>x %= 2;</code> Se x era 3, passa para 1.
&=	Computa E lógico bit-a-bit de 2 valores e atribui o resultado ao primeiro.	<code>x = 15;</code> <code>x &= 3; // faz x igual a 3</code> Equivale a <code>x = x & 3;</code>
=	Computa OU lógico bit-a-bit de 2 valores e atribui o resultado ao primeiro.	<code>x = 15;</code> <code>x = 3; // faz x igual a 15</code> Equivale a <code>x = x 3;</code>

OPERADORES ARITMÉTICOS

Operador	Descrição	Exemplo(s)
+	Soma valores.	<code>a = 2 + 3;</code> <code>b = b + 1;</code>
-	Subtrai valores (como operador binário).	<code>x = x - 5;</code> <code>x = a - b</code>
-	Muda sinal (como operador unitário).	<code>x = -x;</code> <code>x = -(a + b) ;</code>
*	Multiplica valores.	<code>a = 2 * 3;</code> <code>b = c * 5;</code>
/	Divide valores.	<code>a = 50 / 3;</code> <code>b = b * 4;</code>
%	Resto da divisão.	<code>d = 5 % 3;</code> d assume valor 2.

OPERADORES RELACIONAIS

Operador	Descrição	Exemplo(s), supondo a = 3 e b = 5
<code>==</code>	Verdadeiro se os operandos são iguais. Se não são do mesmo tipo, a linguagem tenta converter para a correta comparação.	<code>a == 3; // retorna verdadeiro</code> <code>a == b; // retorna falso</code>
<code>!=</code>	Verdadeiro se os operandos não são iguais. Se não são do mesmo tipo, a linguagem tenta converter para a correta comparação.	<code>a != 3; // retorna falso</code> <code>a != b; // retorna verdadeiro</code>
<code>===</code>	Verdadeiro se os operandos são iguais e do mesmo tipo.	<code>a === 3; // retorna verdadeiro</code> <code>a === "3"; // retorna falso</code>
<code>!==</code>	Verdadeiro se os operandos não são iguais ou não são do mesmo tipo.	<code>a !== b; // retorna verdadeiro</code> <code>a !== "3"; // retorna verdadeiro</code>
<code>></code>	Verdadeiro se o operando esquerdo é maior que o direito.	<code>a > b; // retorna falso</code> <code>b > a; // retorna verdadeiro</code>
<code>>=</code>	Verdadeiro se o operando esquerdo é maior ou igual ao direito.	<code>a >= 3; // retorna verdadeiro</code> <code>b >= 7; // retorna falso</code>
<code><</code>	Verdadeiro se o operando esquerdo é menor que o direito.	<code>a < b; // retorna verdadeiro</code> <code>b < a; // retorna falso</code>
<code><=</code>	Verdadeiro se o operando esquerdo é menor ou igual ao direito.	<code>a <= 3; // retorna verdadeiro</code> <code>a <= 0; // retorna falso</code>

```
var x = 6;
alert (x<6); /*false*/

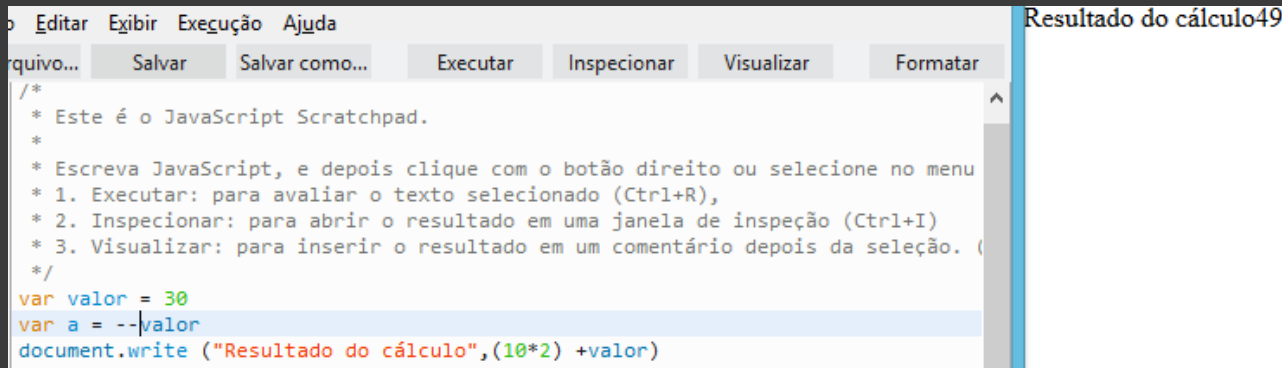
var x = 6;
alert (x==6); /*true*/
```

true

janelas de confirmação desta página?

OK

OPERADORES DE INCREMENTO



```
Editar Exibir Execução Ajuda
Arquivo... Salvar Salvar como... Executar Inspeccionar Visualizar Formatar

/*
 * Este é o JavaScript Scratchpad.
 *
 * Escreva JavaScript, e depois clique com o botão direito ou selecione no menu
 * 1. Executar: para avaliar o texto selecionado (Ctrl+R),
 * 2. Inspeccionar: para abrir o resultado em uma janela de inspeção (Ctrl+I)
 * 3. Visualizar: para inserir o resultado em um comentário depois da seleção. (
 */
var valor = 30
var a = --valor
document.write ("Resultado do cálculo", (10*2) + valor)
```

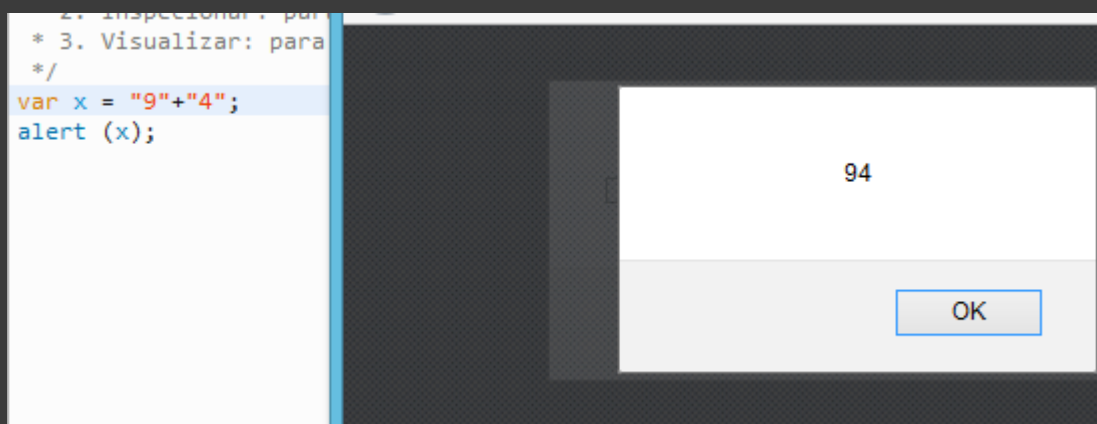
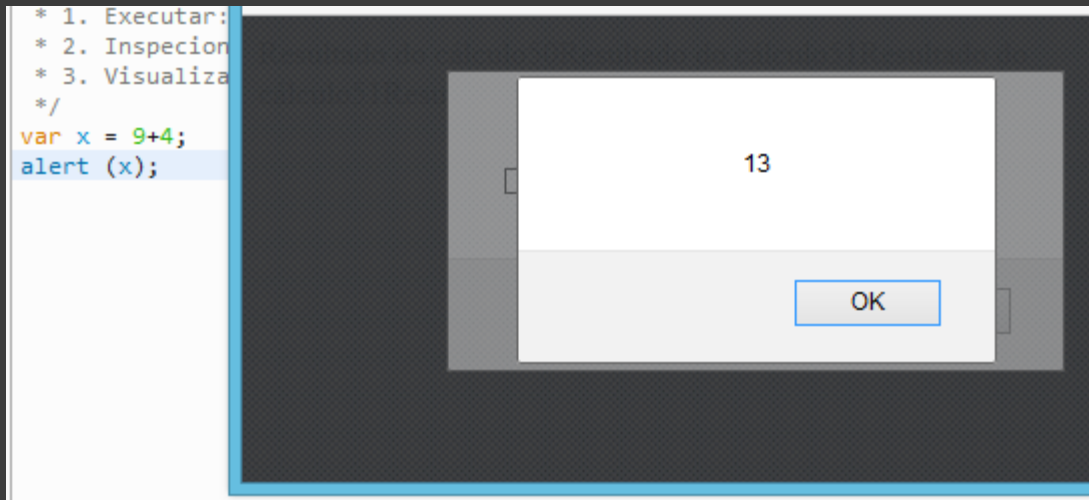
Resultado do cálculo49

- Operador incremental é representado pelo duplo sinal de adição ++ e o decremental pelo –
- Exemplo:
- Var X = 10
- Var z = X++ ou var z = ++X
- Resultado é 11

OPERADORES DE STRING

Operador	Descrição	Exemplo(s)
+	Concatenar strings.	<pre>str_1 = "Bom"; str_2 = str_1 + " dia";</pre> str_2 contém "Bom dia"
+=	Concatenar e atribuir o resultado ao operando da esquerda.	<pre>str_1 = "Bom"; str_1 += " dia";</pre> str_1 contém "Bom dia"

OPERADORES CONCATENAÇÃO +



- STRINGS

OPERADORES LÓGICOS

Em geral são usados com expressões que retornam valores booleanos, isto é, verdadeiro ou falso.

Operador	Descrição	Exemplo(s), supondo a = 3 e b = 5
&&	E lógico: retorna verdadeiro se ambas as expressões são verdadeiras e falso nos demais casos	<code>a==3 && b<10 // retorna verdadeiro</code> <code>a!=3 && b==5 // retorna falso</code>
 	OU lógico: retorna verdadeiro se pelo menos uma das expressões é verdadeira e falso se todas são falsas	<code>a==3 b<10 // retorna verdadeiro</code> <code>a!=3 b==5 // retorna verdadeiro</code> <code>a==1 b==3 // retorna falso</code>
!	NÃO lógico: retorna verdadeiro se o operando é falso e vice-versa	<code>! (a==3) // retorna falso</code> <code>! (a!=3) // retorna verdadeiro</code>

OPERADORES LÓGICOS

Em geral são usados com expressões que retornam valores booleanos, isto é, verdadeiro ou falso.

Operador	Descrição	Exemplo(s), supondo a = 3 e b = 5
&&	E lógico: retorna verdadeiro se ambas as expressões são verdadeiras e falso nos demais casos	<code>a==3 && b<10 // retorna verdadeiro</code> <code>a!=3 && b==5 // retorna falso</code>
 	OU lógico: retorna verdadeiro se pelo menos uma das expressões é verdadeira e falso se todas são falsas	<code>a==3 b<10 // retorna verdadeiro</code> <code>a!=3 b==5 // retorna verdadeiro</code> <code>a==1 b==3 // retorna falso</code>
!	NÃO lógico: retorna verdadeiro se o operando é falso e vice-versa	<code>! (a==3) // retorna falso</code> <code>! (a!=3) // retorna verdadeiro</code>

OUTROS OPERADORES

Operador	Descrição	Exemplo(s)
?:	Operador condicional na forma: condição? exp_1 : exp_2 Se condição é verdadeira, retorna exp_1 e, se é falsa, retorna exp_2.	<pre>val = com_frete? "110,00" : "100,00"; document.write("Preço: " + val);</pre> Se <i>com_frete</i> é verdadeiro, escreve: "Preço: 110,00"
,	Operador vírgula na forma: exp_1, exp_2 Calcula ou avalia ambas as expressões.	Uso comum é em loops tipo for: <pre>for(m=0,n=0; m<5; m++,n++){...</pre>
delete	Exclui um objeto, uma propriedade de objeto, um elemento de uma array ou uma variável explicitamente declarada com <code>var</code> . Para elemento de array, não altera o seu tamanho. Apenas o elemento fica indefinido.	<pre>val = new Array(10); delete val[4]; delete val; var nivel = 20; delete nivel;</pre>
new	Cria um novo objeto entre aqueles já existentes na linguagem (str = new String("abc"), etc) ou cria um objeto definido pelo usuário.	<pre>function prod(nome,unidade,valor){ this.nome = nome; this.unidade = unidade; this.valor = valor; } P01 = new prod("banana","kg",2.00);</pre>
this	Faz referência ao objeto em uso.	<pre>Digite 6 a 10 caracteres <input type="text" name="ab" size=10 onChange="verificar(this)"></pre> A função <i>verificar</i> (a definir) recebe como parâmetro a caixa de texto, o que permite verificar a quantidade de caracteres digitados.
typeof	Retorna uma string indicativa do tipo de operando.	<pre>var val = 10; typeof val retorna "number"</pre> <pre>str = "abc"; typeof str retorna "string"</pre>
void	Indica uma expressão para ser avaliada mas sem retornar nenhum valor.	O código abaixo cria um hyperlink nulo, isto é, nada abre: <pre>Este link nada abre</pre>

EXERCICIOS – REPITA E TESTE NO FIREFOX

```
/*
    autor: Adriane
    descricao: estudo de operadores somente com numericos  nao strings
*/
var x = 9 + 4; /*dados com tipo numerico*/
var xx = 5 % 3; /*fara a divisao e retornara 2 o resto da divisao*/
var y = "9" + "4"; /*dados com tipo strings com o sinal de + significa
q esta concatenando ou seja unindo 2 strings*/
var unario = -(-4); /*ex d operador unario, o resultado vai ser positivo*/
/*exemplo de incremento*/
var a = 6;
var b = ++a;
alert("a="+a);
alert("b="+b);
alert (xx);

// alert(y) ;
```

CONTROLE, CONDICIONAIS E LOOPS

DECLARAÇÕES E ESTRUTURAS

ESTRUTURAS DE CONTROLE

Existem algumas estruturas de controle que lhe permitem modificar o fluxo de execução de um programa. Estas estruturas permitem executar o código baseado em condições lógicas ou um número determinado de vezes.

FOR

FOR IN...

IF

IF...ELSE

WHILE....

Loop - FOR

- Recebe 3 argumentos
- Variável (inicialização), operador (condição), incremento (ação)

Repete uma seção do código um determinado número de vezes. Consiste de uma declaração que define as condições da estrutura e marca seu início

Estabelece um contador inicializando uma variável com um valor numérico. O contador é manipulado através da variável especificada no comando toda a vez que o loop alcança seu fim, permanece nesse loop até que a condição seja satisfeita ou a instrução Break seja executada.

```
número: 1
número: 2
número: 3
número: 4
número: 5
número: 6
número: 7
número: 8
número: 9
número: 10
```

```
Scratchpad
Arquivo  Editar  Exibir  Execução  Ajuda
Abrir arquivo...  Salvar  Salvar como...  Executar

1  /*
2   * Este é o JavaScript Scratchpad.
3   *
4   * Escreva JavaScript, e depois clique com o
5   * 1. Executar: para avaliar o texto selecionado
6   * 2. Inspecionar: para abrir o resultado em
7   * 3. Visualizar: para inserir o resultado em
8   */
9  for (i=1 ; i<=10 ; i++){
10     document.write("número: " + i + "<br>");
11 }
12
```

```
bomdia
bomdia
bomdia
```

```
9  for (contador = 0; contador < 100; contador ++){
10     document.write("bomdia" + "<br>");
11 }
```

```
var texto = "";  
for (var i=0; i<10; i++)  
{  
    texto += i + ",";  
}  
alert(texto);
```

0,1,2,3,4,5,6,7,8,9,

OK

Laço de Repetição WHILE

```
<script >
function condicao(){
    while (form1.nome.value=="")
    {
        alert("Favor preencher o campo");
        form1.nome.value=prompt("digite seu nome agora","");
    }
    alert("obrigado,"+form1.nome.value);
}

</script>
</head>
<body>
<form name=form1>
    Nome: <input type="text" name="nome" onBlur="condicao(this.value)">
    <button type="submit">enviar</button>
</form>
</body>
</html>
```

Nome:

Favor preencher o campo

IF/ELSE

```
<script>
  var idade = 10;
  if (idade<18){
    alert ("criança/adolescente");
  }
  else{
    alert("adulto");
  }
</script>
```

```
<script>
  var idade = 28;
  if (idade<=11){
    alert ("criança");
  }
  else if (idade>16 && idade <=21){
    alert("adolescente");
  }
  else if (idade>21 && idade <60){
    alert("adulto");
  }
  else {
    alert("melhor idade");
  }
</script>
```

exercício

- Faça os algoritmos criados no portugol usando condição SE na linguagem javascript

Condição IF - formulário

```
<script >
function condicao(){
    if (form1.nome.value=="")
    {
        alert("Favor preencher o campo");
        form1.nome.focus();
    }
    alert("obrigado,"+form1.nome.value);
}

</script>
</head>
<body>
<form name=form1>
    Nome: <input type="text" name="nome" onBlur="condicao(this.value)">
    <button type="submit">enviar</button>
</form>
```

Exercício

- Fazer um formulário com 4 campos e exigir que o usuário digite todos os campos;

Criando Formulário de Matrícula

```
<body>
<form name="inscricao" id="inscricao">
  <label for="nome">Nome:</label>
  <input type="text" name="nome"><br>
  <label for="tipo">Tipo de participante:</label><br>
  <select name="participante" id="participante" onChange="ocultadiv (
matricula, aluno)">
    <option value="aluno" id="aluno">Aluno</option>
    <option value="visitante" id="visitante">Visitante</option>
    <option value="" selected="selected"></option>
  </select><br>
  <!-- conteudo vai ficar oculto até que escolham aluno -->
  <div id="matricula" style="display:none">
    Matricula:<input type="text" name="matricula"><br>
    RG:<input type="text" name="rg"><br>
    E-mail:<input type="email" name="email"><br>
    <button type="submit"> enviar</button>
  </div>
</form>
</body>
</html>
```

Criando Função

```
<script>
function ocultadiv (matricula,aluno){
    valor =document.getElementById("participante").value;
    if (valor=="aluno"){
        document.getElementById("matricula").style.display="block";
    }
    else{
        document.getElementById("matricula").style.display="none";
    }
}
</script>
```



NOME:

TIPO DE PARTICIPANTE:

MATRICULA:

RG:

E-MAIL:

Referência

- JavaScript – Aplicações Interativas para a Web do autor ADRIANO GOMES LIMA.
- MSPC → <http://goo.gl/6k49s>
- MDN – mozilla → <https://goo.gl/g3u51q>
- K19 – Javascript – apostila → <http://goo.gl/URLuon>