

Alexandre Meslin
meslin@puc-rio.br

Programação para a Web em Python



Front-End

Ementa

HTML

CSS

JavaScript

TypeScript

- Variáveis
- Operadores
- Comandos
- Objetos

- Variáveis
- Operadores
- Comandos
- Objetos

Referências

Bibliografia

Na Internet

- HTML/CSS
 - Criando Sites com HTML, Silva, M, Novatec
 - Design Criativo com HTML, Weinman, L e outros, Ciência Moderna
- JavaScript
 - Use a Cabeça – JavaScript, Monison, M, Altabooks
 - Só JavaScript, Yank, K, Bookman
- TypeScript
 - Getting Started with TypeScript: Includes Introduction to Angular - Thomas Claudius Huber

- <http://cursos.meslin.com.br/home/javascript>
- <http://exemplosweb.meslin.com.br> (vps)
- <http://www.w3.org>
- <http://www.w3schools.com>
- <http://learnlayout.com> (só layout - excelente)
- <http://eloquentjavascript.net/>

JavaScript



JavaScript

- Linguagem script popular
- Suportada por diversos navegadores web e outras ferramentas
- Interage com HTML, adicionando interatividade
- Normalmente as páginas HTML são estáticas
 - Janelas pop-up
 - Iteração com formulários
 - Cálculos
 - Efeitos especiais

JavaScript x Java (para não confundir nunca mais)

JavaScript	Java
Interpretado pelo computador no lado do cliente	Compilado no servidor antes de ser executado
Referências a objetos são verificados em tempo de execução	Referências a objetos são verificados em tempo de compilação
Não precisa declaração de tipos de dados	Tipos de dados precisam ser declarados
Pode acessar objetos do navegador	Não pode acessar objetos do navegador

Termos

- Objeto: dados e funcionalidade juntos
- Propriedade: atributos (valores) que são associados a alguma coisa
- Método: uma ação que um objeto pode realizar
- Evento: uma ação iniciada por um usuário ou pelo computador
- Variável: um lugar para armazenar valores em um computador (propriedade está relacionada a objeto)
- Função: uma rotina ou procedimento que realiza uma ação (método está relacionado a objeto)

Em Resumo

- HTML → estrutura
- CSS → estilo
- JavaScript → interatividade

Como incluir JavaScript em uma página HTML

- JavaScript dentro da página HTML

```
<html>
<body>
<script type="text/javascript">
    document.write("Alo mundo!");
</script>
</body>
</html>
```

- O comando `document.write()` é um comando padrão do JavaScript para escrever em uma página HTML

ATENÇÃO!!!

```
document.write("Alo mundo!");
```

- **document** → um objeto
- **. (o ponto)** → liga o objeto ao seu método
- **write()** → um método do objeto document (todo método é chamado utilizando-se parêntesis)
- **"Alo mundo!"** → uma string (escrita entre aspas)

Como incluir JavaScript em uma página HTML

- O mesmo exemplo, agora com tags HTML dentro do JavaScript

```
<html>
<body>
<script type="text/javascript">
  document.write("<h1>Alo mundo!</h1>");
</script>
</body>
</html>
```

Onde incluir o JavaScript

- JavaScript é executado durante a carga da página HTML
- Nem sempre queremos que ele execute durante
- Algumas vezes é necessário que ele execute antes ou após o carregamento da página
- Incluir o JavaScript no cabeçalho (<head>) de uma página garante que ele será executando antes da carga da página

Exemplo

```
<html>
<head>
<script type="text/javascript">
    document.write("<title>Título da página</title>");
</script>
</head>
<body>
<script type="text/javascript">
    document.write("<h1>Conteúdo da página</h1>");
</script>
</body>
</html>
```



Utilizando um arquivo JavaScript externo (a forma preferencial é essa!!!)

```
<html>
<head>
<title>Página com JavaScript externo</title>
<script type="text/javascript" src="arquivo.js">
</script>
</head>
<body>
Corpo da Página
</body>
</html>
```

```
// código javascript
/*
 * aqui eu posso colocar qualquer
 * comando JavaScript
 */
alert("Alo mundo!");
```

arquivo.js

Comandos JavaScript

- A linguagem JavaScript é case-sensitive
- Um comando JavaScript é executado pelo navegador
- Normalmente termina-se um comando JavaScript com ;
(boa prática de programação, mas desnecessário)
- O uso de ; permite escrever mais de um comando por linha

Código JavaScript

- Código JavaScript é uma sequência de comandos JavaScript
- O código é executado sequencialmente, comando por comando
- Exemplo:

```
document.write("<h1>Este é um cabeçalho</h1>");  
document.write("<p>Isto é um parágrafo.</p>");  
document.write("<p>Este é outro parágrafo.</p>");
```

Bloco de comandos

- Comandos JavaScript podem ser agrupados em blocos
- Um bloco inicia por um { e termina por um }
- A finalidade do bloco é agrupar os comandos para que eles sejam tratados como se fossem apenas um único comando (comando composto)
- Exemplo:

```
{  
    document.write("<h1>Este é um cabeçalho</h1>");  
    document.write("<p>Isto é um parágrafo.</p>");  
    document.write("<p>Este é outro parágrafo.</p>");  
}
```

Comentários

- Finalidade: documentação
- Comentários podem ser adicionados para explicar o código JavaScript
- Exemplo:

```
// escreve um cabeçalho
document.write("<h1>Este é um cabeçalho</h1>");
// escreve um parágrafo
document.write("<p>Isto é um parágrafo.</p>");
document.write("<p>Este é outro parágrafo.</p>");
```

Comentários em múltiplas linhas

- Um comentário de múltiplas linhas começa por /* e termina com */
- Exemplo:

```
/*
Este código a seguir escreverá:
- Um cabeçalho
- Dois parágrafos
*/
document.write("<h1>Este é um cabeçalho</h1>");
document.write("<p>Isto é um parágrafo.</p>");
document.write("<p>Este é outro parágrafo.</p>");
```

Comentários e código

- Podemos utilizar comentários para evitar que algum comando seja executado
- Muito utilizado durante a fase de desenvolvimento
- Exemplo:

```
document.write("<h1>Este é um cabeçalho</h1>");  
// document.write("<p>Isto é um parágrafo.</p>");  
document.write("<p>Este é outro parágrafo.</p>");
```

Comentários e comandos

- Podemos acrescentar comentários no final de uma linha de comando
- Exemplo:

```
document.write("<h1>Este é um cabeçalho</h1>"); //cabeçalho  
document.write("<p>Isto é um parágrafo.</p>"); /* parágrafo */  
document.write("<p>Este é outro parágrafo.</p>");
```

Estrutura da Linguagem

■ Valores constantes primitivos:

- "Alexandre Meslin" String entre aspas
- 'Linguagem JavaScript' String entre plices
- 8752 -2578 Inteiro na base 10
- 0257 -0752 Inteiro na base 8
- 0xAB12 -0x3CD4 Inteiro na base 16
- 3.14 -2.7 2.3E11 Número em ponto flutuante
- true false Valores booleanos
- Null Representa a ausência de valor intencionalmente
- Undefined Representa um valor não atribuído
- 8752n BigInt

Declaração e uso de variáveis

■ Nomes de variáveis

- Letras, números, _ e \$
- Não podem começar por números
- Não podem ser iguais a palavras reservadas

■ Exemplo:

```
var i;           // cria a variável (local)
i = 8752;        // cria e inicializa a variável i com 8752 (global)
var i = 8752;    // cria e inicializa a variável i com 8752 (local)
```

■ Os tipos de dados são assumidos dinamicamente

```
i = 25;
i = "vinte e cinco";
```

Declaração e uso de variáveis

- var vs let

```
<script>
let a = 8;
let b = 5;
var c = 25;
{
  let a = 7;          // Variável válida somente dentro do bloco
  var d = 78;         // Variável válida dentro e fora do bloco
}
//let b = 2;          // ERRADO!!!
var c = 87;          // Correto!!!
</script>
```

Atenção!!!

- Variáveis criadas sem o uso de `var` são sempre globais, mesmo quando criadas dentro de uma função!



Armadilhas...

- Cuidado: não utilize nomes com escritas diferentes mas pronúncias iguais ou parecidas.
- Exemplo: não crie as seguintes variáveis:
 - Nome e nome
 - Idade e idade
 - Num1 e num1
- Você vai acabar confundindo nome com Nome!



Armadilhas...

- Não utilize as palavras reservadas como nome de variável, função ou objeto.
- Palavras reservadas:
 - abstract arguments await
 - boolean break byte
 - case catch char class const continue
 - debugger default delete do double
 - else enum eval export extends
 - false final finally float for function
 - goto
 - if implements import in instanceof int interface
 - let long
 - native new null
 - package private protected public
 - return
 - short static super switch synchronized
 - this throw throws transient true try typeof
 - var volatile void
 - while with
 - yield



Tipos de Dados

- JavaScript permite que uma mesma variável contenha diferentes tipos de dados
- Tipos de dados primitivos:
 - Número: inteiro e ponto flutuante
 - Booleano: true e false
 - String: sequência de caracteres
- Tipos de dados compostos
 - Objeto: uma coleção de dados nomeada
 - Array: uma coleção de dados ordenada numericamente
- Tipos de dados especiais
 - Null: nulo
 - Undefined: valor de variável não inicializada
 - NaN: valor numérico que representa um não número (erro)
 - NaN não é igual a nada, incluindo NaN

Armadilhas...

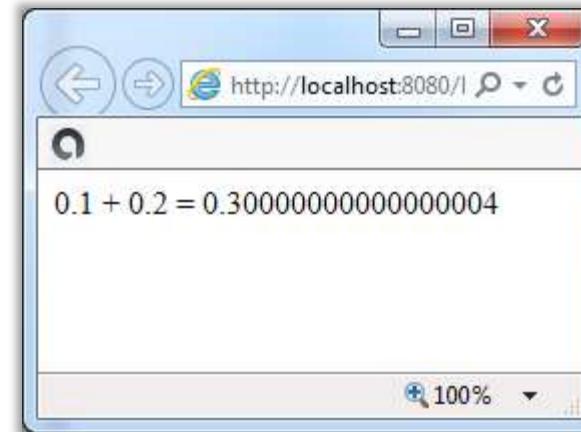
```
<!DOCTYPE html>
<html>
<head>
<title>Precisão da soma</title>
</head>
<body>

<script type="text/javascript">
var a, b, res;

a = 0.1;
b = 0.2;
res = a + b;

document.write(a + " + " + b + " = " + res);
</script>

</body>
</html>
```



Entrada e saída de dados

- `document.write()`
 - Escreve alguma coisa na página
 - Pode escrever na seção `<head>` ou `<body>`
 - Permite código JavaScript produzir texto HTML

```
<html>
<head>
<script type="text/javascript">
document.write("<title>Título da página</title>");
</script>
</head>
<body>
<script type="text/javascript">
document.write("<h1>Conteúdo da página</h1>");
</script>
</body>
</html>
```

Erros que eu não vou fazer...

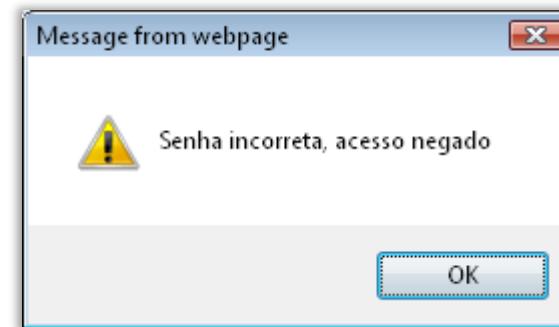
- Depois que a página está totalmente carregada, o uso do comando `document.write()` apaga toda a página.
- O resultado final será apenas os textos gerados após o carregamento total. Ou seja, o que for escrito pelos comandos `document.write()` depois da página carregada.

- Obs.: se você quiser emitir mensagens depois da página carregada, utilize `alert()` ou DHTML.

Entrada e saída de dados

- `window.alert()`
- `alert()`
 - Abre uma janela para exibir um aviso ao usuário

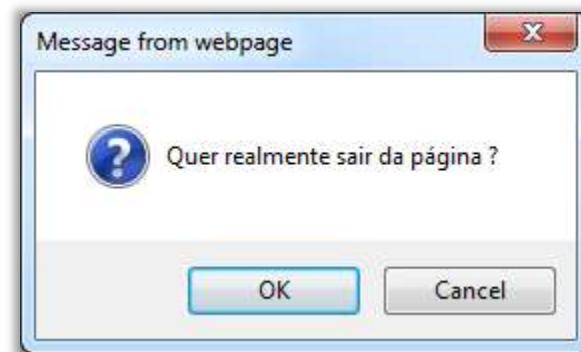
```
<html>
<body>
<script type="text/javascript">
window.alert("Senha incorreta, acesso negado");
alert("Senha incorreta, acesso negado");
</script>
</body>
</html>
```



Entrada e saída de dados

- window.confirm()
- confirm()
 - Abre uma janela para o usuário confirmar ou não

```
<html>
<head>
<title>Exemplo de confirm()</title>
</head>
<body>
<script type="text/javascript">
    if (confirm("Quer realmente sair da página ?"))
        alert("Então adeus!");
</script>
</body>
</html>
```



Entrada e saída de dados

- `window.prompt()`
- `prompt()`
 - Abre uma janela para pedir uma string ao usuário

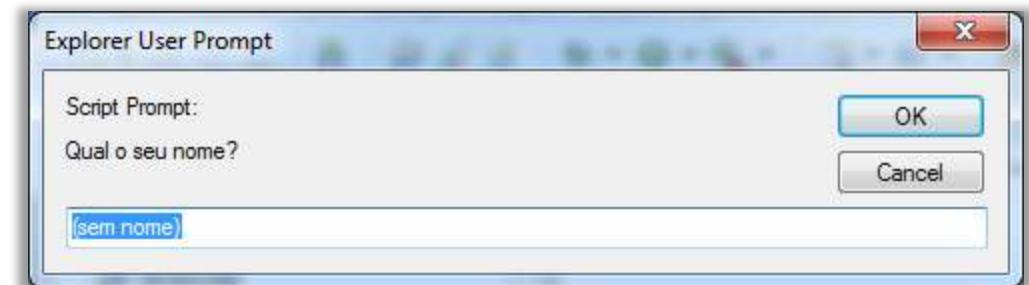
```
<html>
<body>
<script type="text/javascript">
var nome;
nome=prompt("Qual o seu nome?");
</script>
</body>
</html>
```



Entrada e saída de dados

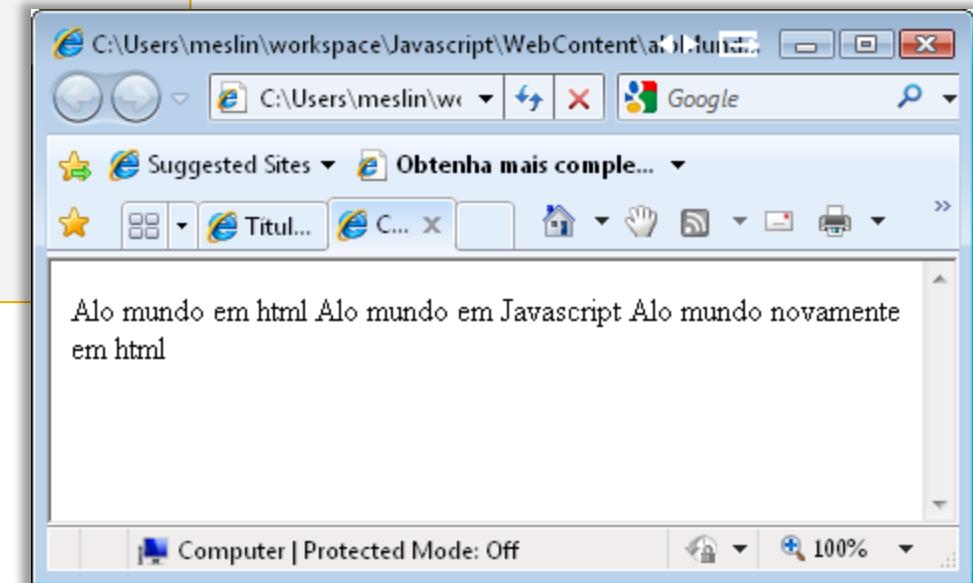
- Um valor padrão (default) pode ser fornecido
- Evita que apareça "undefined" na linha da resposta

```
<html>
<body>
<script type="text/javascript">
var nome;
nome=prompt("Qual o seu nome?", "(sem nome)");
</script>
</body>
</html>
```



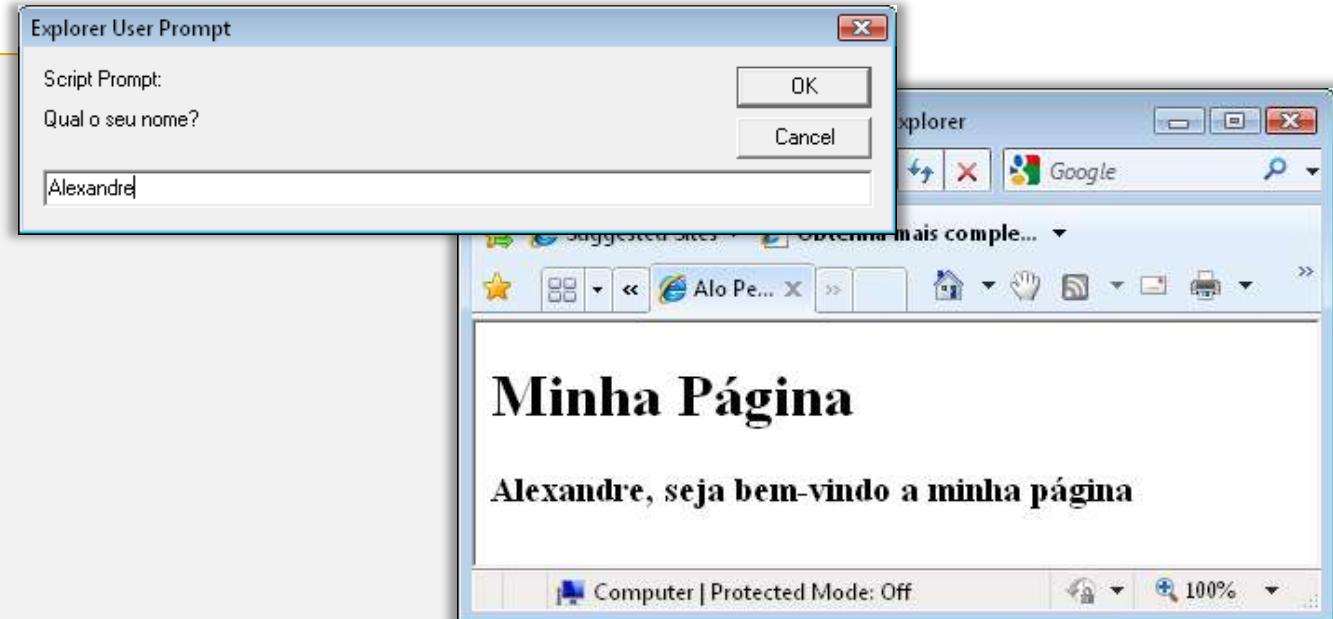
Exemplo:

```
<html>
<body>
Alo mundo em html
<script type="text/javascript">
document.write("Alo mundo em Javascript");
</script>
Alo mundo novamente em html
</body>
</html>
```



Meu Segundo JavaScript

```
<html>
<head>
<title>Alo Personalizado</title>
<script type="text/javascript">
var nome;
nome = prompt("Qual o seu nome?");
</script>
</head>
<body>
<h1>Minha Página</h1>
<script type="text/javascript">
document.write("<h3>" + nome + ", seja bem-vindo a minha página</h3>");
</script>
</body>
</html>
```



Expressões

- Conjunto de constantes, variáveis, operadores, funções que resultam em um único valor
- As expressões podem ser do tipo (tipo do resultado):
 - Aritméticas
 - Logicas
 - Strings
 - Condicionais

Tipos de Operadores

- Aritmético
- Lógico
- Relacional
- String
- Bit-wise
- Atribuição
- Condicional

Operadores

- Aritméticos
 - Soma +
 - $a + b$
 - Subtração –
 - $a - b$
 - Multiplicação *
 - $a * b$
 - Divisão /
 - a / b
 - Resto da divisão de inteiros %
 - $a \% b$
 - Incremento de uma unidade ++
 - $++a$ $a++$
 - Decremento de uma unidade --
 - $--a$ $a--$

Operadores

- Operadores Aritméticos:
 - + (Adição)
 - - (Subtração)
 - * (Multiplicação)
 - / (Divisão)
 - % (Resto da divisão - Módulo)
 - ** (Exponenciação)
- Operadores de Atribuição:
 - = (Atribuição)
 - += (Atribuição de adição)
 - -= (Atribuição de subtração)
 - *= (Atribuição de multiplicação)
 - /= (Atribuição de divisão)
 - %= (Atribuição de resto da divisão)
 - **= (Atribuição de exponenciação)
 - <<= (Atribuição de deslocamento à esquerda)
 - >>= (Atribuição de deslocamento à direita)
 - >>>= (Atribuição de deslocamento à direita sem preenchimento)
- Operadores de Comparação:
 - == (Igual a - com coerção de tipo)
 - === (Igual a - sem coerção de tipo)
 - != (Diferente de - com coerção de tipo)
 - !== (Diferente de - sem coerção de tipo)
 - > (Maior que)
 - < (Menor que)
 - >= (Maior ou igual a)
 - <= (Menor ou igual a)
- Operadores Lógicos:
 - && (E lógico - AND)
 - || (Ou lógico - OR)
 - ! (Negação lógica - NOT)
- Operadores Bit a Bit (Bitwise):
 - & (AND bit a bit)
 - | (OR bit a bit)
 - ^ (XOR bit a bit)
 - ~ (NOT bit a bit)
 - << (Deslocamento à esquerda)
 - >> (Deslocamento à direita com preenchimento de sinal)
 - >>> (Deslocamento à direita sem preenchimento de sinal)
- Operador Ternário:
 - ? : (Operador ternário - condicional)
- Operadores Especiais:
 - , (Vírgula - para avaliar múltiplas expressões e retornar a última)
- Operadores de Tipo:
 - typeof (Retorna uma string com o tipo do operando)
 - instanceof (Verifica se um objeto é uma instância de uma classe ou construtor)
- Operadores de Propriedade:
 - . (Acessar uma propriedade de um objeto)
 - [] (Acessar uma propriedade de um objeto usando notação de colchetes)
- Operadores de Função:
 - () (Chamar uma função)
- Operadores de Desestruturação (ES6):
 - ... (Spread/rest operator)
 - {} (Desestruturação de objetos)
 - [] (Desestruturação de arrays)

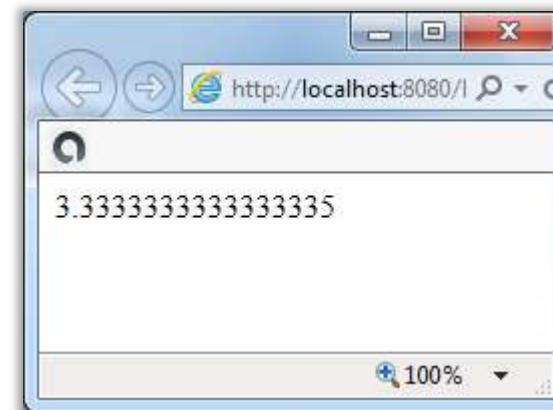
O Operador +

- Adição e concatenação
- Se ambos os operadores são numéricos então
 - Soma os números
- Senão
 - Converte ambos os valores para string
 - Contatena ambos
 - '\$' + 3 + 4 → '\$34'
- Observação
 - +"42" → 42
 - +"3" + (+"4") → 7
 - +"3" + "4" → "34"

O Operador /

- A divisão de 2 inteiros pode resultar em um valor real
 - $10 / 3 = 3.333333333333335$

```
var a, b, res;  
a = 10;  
b = 3;  
res = a / b;  
document.write(res);
```



Operadores

- Atribuição
 - Simples =
 - $a = b + c$
 - Composta $+= \quad -= \quad *= \quad /= \quad \%=$
 - $a += b \quad (a = a + b)$

Operadores

- Relacionais
 - Equivalente ==
 - a == b
 - Estritamente equivalente === (não há conversão de tipo)
 - a === b
 - Diferente !=
 - a != b
 - Estritamente diferente !== (não há conversão de tipo)
 - a !== b
 - Menor <
 - a < b
 - Maior >
 - a > b
 - Menor ou igual <=
 - a <= b
 - Maior ou igual >=
 - a >= b

Erros que eu não vou fazer...

- O que o código JavaScript abaixo apresenta na console?

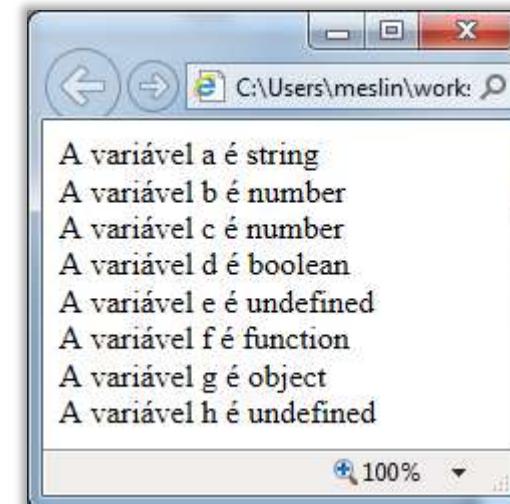
```
var variavel;  
variavel +1;  
console.log(variavel);
```

O operador typeof

- Operador unário
- Retorna o tipo atual da variável:
 - number, string, boolean, object, function, undefined, null

```
<body>
<script language="javascript">
    function f() {}
    var a = "texto", b = 8752, c = 2.1, d = false, e, g = Object();
    document.write("A variável a é " + typeof a + "<br/>");
    document.write("A variável b é " + typeof b + "<br/>");
    document.write("A variável c é " + typeof c + "<br/>");
    document.write("A variável d é " + typeof d + "<br/>");
    document.write("A variável e é " + typeof e + "<br/>");
    document.write("A variável f é " + typeof f + "<br/>");
    document.write("A variável g é " + typeof g + "<br/>");
    document.write("A variável h é " + typeof h + "<br/>");

</script>
</body>
</html>
```



Operadores de Manipulação de Bits

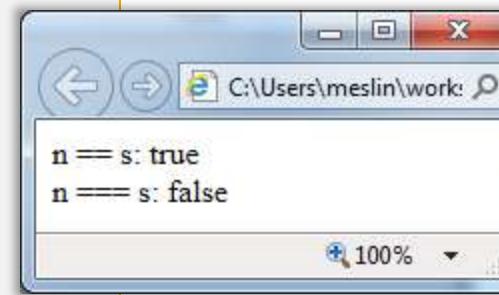
Operador	Nome	Descrição
&	Bitwise AND	AND bit a bit
	Bitwise OR	OU bit a bit
^	Bitwise XOR	OU Exclusivo bit a bit
<<	Bitwise left shift	Deslocamento para a esquerda inserindo zeros
>>	Bitwise signed right shift	Deslocamento para a direita mantendo o sinal
>>>	Bitwise zero-fill right shift	Deslocamento para a direita inserindo zeros

Operadores de comparação

```
<html>
<head>
<title>Exemplo de comparação</title>
</head>
<body>

<script type="text/javascript">
var n = 8752, s = '8752';
document.write("n == s: " + (n == s) + '<br/>');
document.write("n === s: " + (n === s) + '<br/>');
</script>

</body>
</html>
```



Operações

- String
 - Concatenação

```
nome = "Alexandre";  
sobrenome = "Meslin";  
nomeCompleto = nome + " " + sobrenome;
```

- Comparação

```
3 == "3"
```

- Outras operações

```
x = "8752";           // x string  
a = x +10;            // a ← 875210  
b = x -10;            // b ← 8742
```



Conversão explícita de tipos

- `parseInt(str)` ou `parseInt(str, base)`
 - Converte uma string para um número inteiro
 - Exemplo:

```
num = "3A";  
x = parseInt(num);           // x ← 3  
y = parseInt(num, 16);      // y ← 58
```

- `parseFloat(str)`
 - Converte uma string em um número real
 - Exemplo

```
z = parseFloat("3.14");
```

Curiosidades

- Para formatar o texto de saída:
 - toFixed()
 - toPrecision()
 - toExponential()

Ordem de Precedência

Precedência	Operador
1	Parênteses, chamada de função
2	<code>~, -, ++, --, new, void, delete</code>
3	<code>*, /, %</code>
4	<code>+, -</code>
5	<code><<, >>, >>></code>
6	<code><, <=, >, >=</code>
7	<code>==, !=, ===, !==</code>
8	<code>&</code>
9	<code>^</code>
10	<code> </code>
11	<code>&&</code>
12	<code> </code>
13	<code>?:</code>
14	<code>=, +=, -=, *=, ...</code>
15	Operador vírgula (,)

Ordem de Precedência da mais prioritária para a menos prioritária

Operador	Descrição
<code>()</code>	Parenteses (para controlar a ordem de avaliação)
<code>. e []</code>	Acesso a propriedades de objetos e arrays
<code>()</code>	Chamada de função
<code>++ e --</code>	Incremento e decremento pós-fixo
<code>!, ~, - (unário), + (unário), typeof, void, delete, await</code>	Operadores unários
<code>**</code>	Exponenciação
<code>*, /, %</code>	Multiplicação, divisão e resto da divisão
<code>+, -</code>	Adição e subtração
<code><<, >>, >>></code>	Deslocamento de bits
<code><, <=, >, >=, in, instanceof</code>	Operadores de comparação
<code>==, !=, ===, !==</code>	Igualdade e desigualdade
<code>&</code>	AND bit a bit
<code>^</code>	XOR bit a bit
<code> </code>	OR bit a bit
<code>&&</code>	E lógico
<code> </code>	Ou lógico
<code>? :</code>	Operador ternário (condicional)
<code>=, +=, -=, *=, /=, %=, **=, <<=, >>=, >>>=, &=, ^=, =</code>	Operadores de atribuição
<code>,</code>	Vírgula (usada para separar múltiplas expressões)



PERGUNTAR NÃO OFENDE

Erros que eu não vou fazer...

```
<!DOCTYPE html>
<html>
<head>
<title>Atribuição</title>

<script type="text/javascript">
var a, b, y;
// algumas linhas de código
var x parseFloat(y);

a == b;
//outras linhas de código
</script>

</head>
<body>

</body>
</html>
```

E quando tudo der errado...

- No Chome 34:
 - Clique nas 3 linhas horizontais no lado superior direito (barra de ferramentas).
 - Selecione Tools ou Ferramentas
 - Selecione Developer tools
 - Clique em Console

E quando tudo der errado...

- No IE 11:
 - Clique na engrenagem no lado superior direito (barra de ferramentas).
 - Selecione F12 Developer Tools
 - Clique em Console (um quadrado localizado no lado esquerdo com um sinal de maior dentro)

E quando tudo CONTINUAR a dar errado...

- Utilize o comando `console.log(texto)` para enviar um texto para a console do navegador sem interferir na formatação da sua página:

```
console.log(mensagem);
```

Exercícios



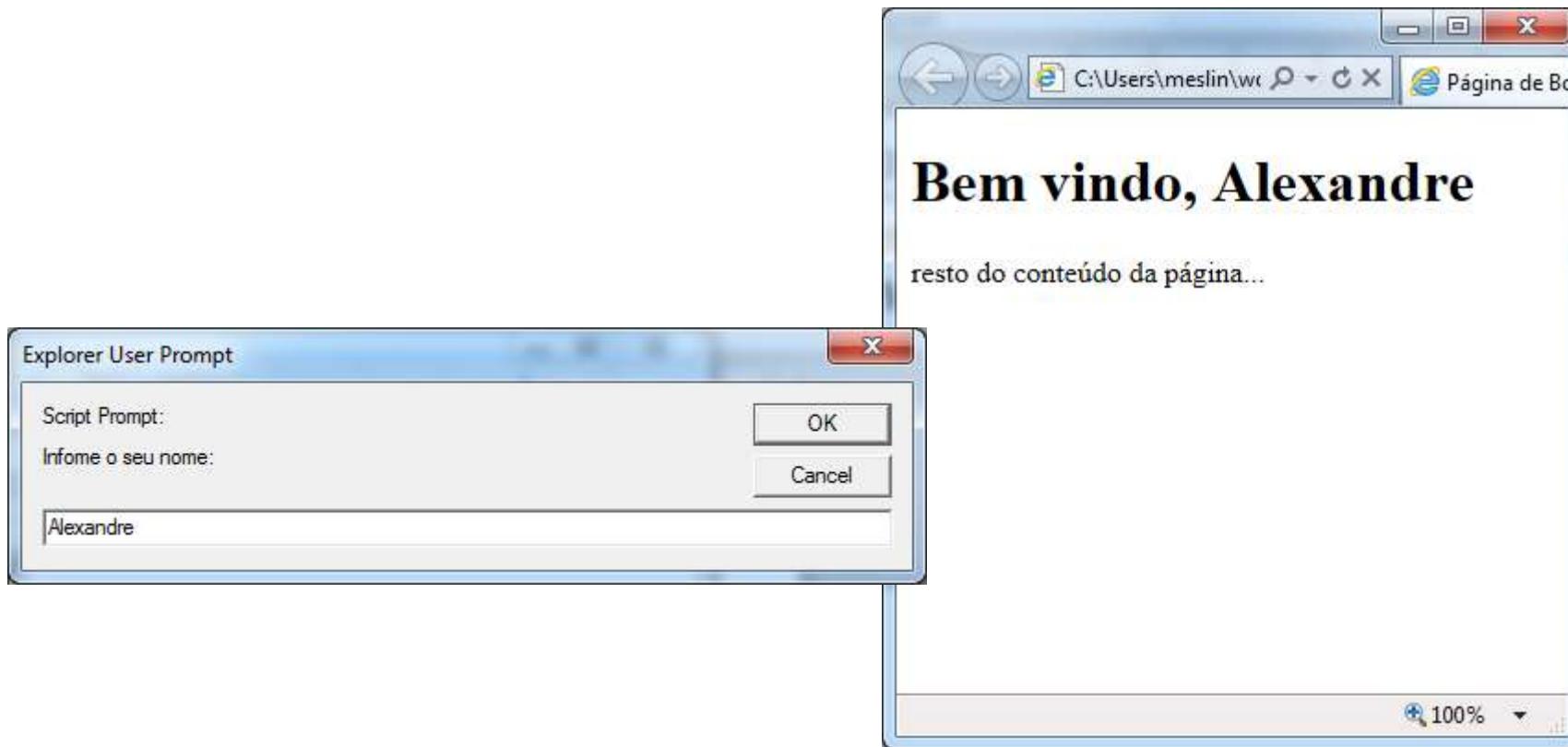
He dreamt of going to
the gym a lot. But didn't.

Exercícios

- Faça uma página que some 25 com 78 e mostre o resultado em:
 - Uma página
 - Uma janela de alerta
- Formato de saída:
 $25 + 78 = 103$

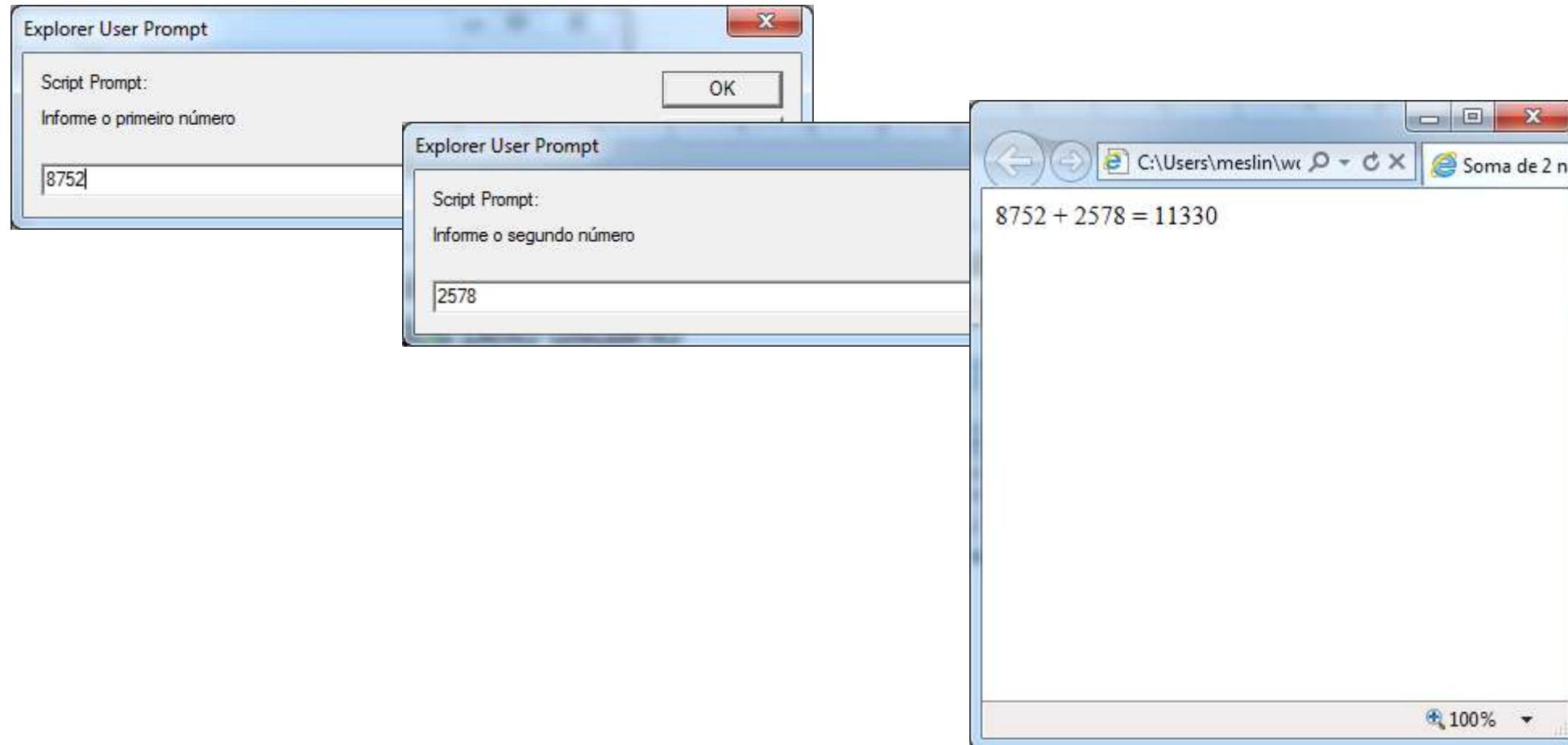
Exercícios

- Faça uma página de boas vindas personalizada



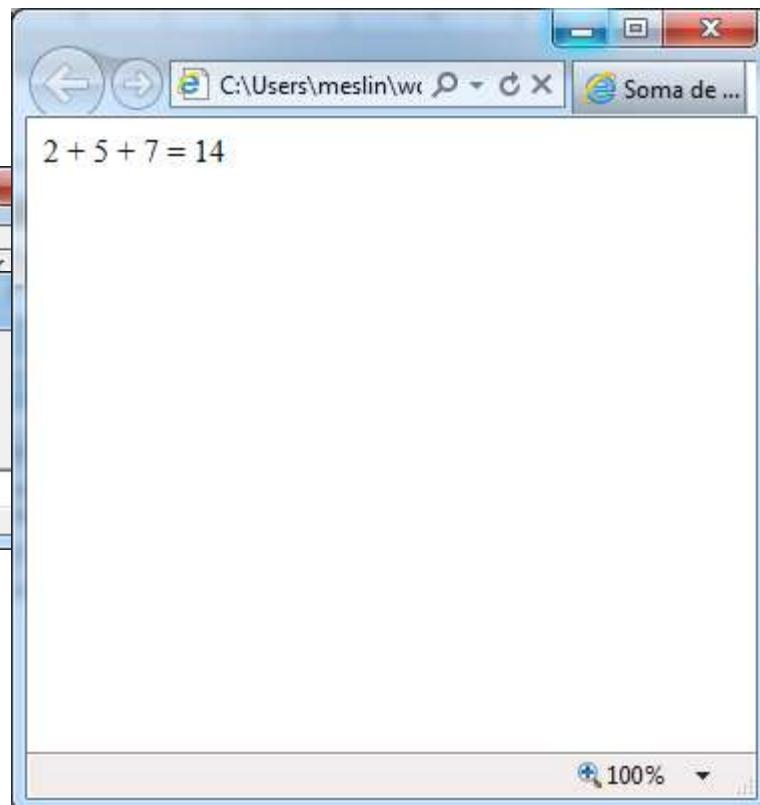
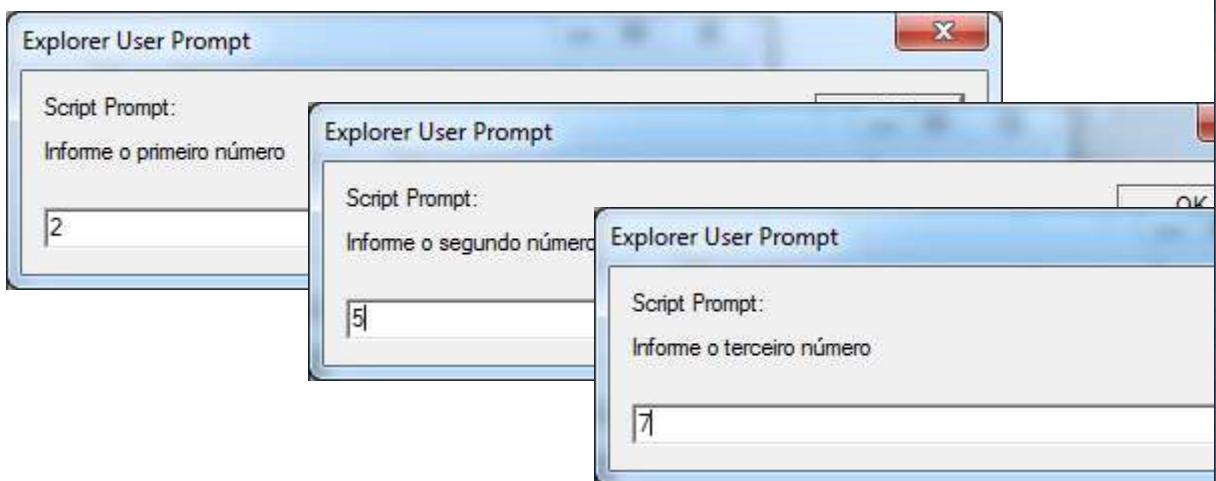
Exercícios

- Faça uma página para somar 2 números informados pelo



Exercícios

- Faça uma página para somar 3 números informados pelo usuário



Exercício

- Faça uma página para calcular o consumo de combustível por quilômetro de um veículo.
- Obtenha do usuário a distância percorrida e a quantidade de combustível consumido.
- Dicas:
 - Tente usar `Math.round()`
 - Faça uma outra tentativa usando `resultado.toFixed(numeroDeCasasDecimais)`

Exercício

- Calcule o índice de massa corporal (IMC) de um usuário.
O usuário deverá informar o seu peso em kg e a sua altura em metros.
 - Obs.: o IMC é calculado dividindo-se o peso pelo quadrado da altura de uma pessoa.

Exercício Extra

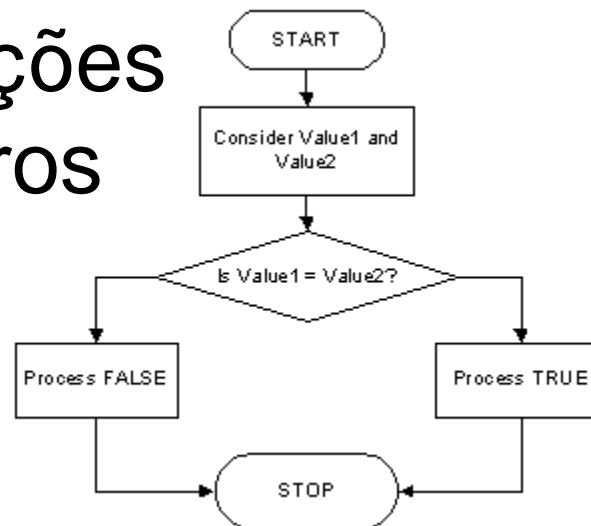
(extraido de <http://www.w3resource.com/javascript-exercises/javascript-basic-exercises.php>)

- 4. Write a JavaScript program to find the area of a triangle where lengths of the three of its sides are 5, 6, 7.

- Modifique o exercício anterior para permitir ao usuário informar os tamanhos dos lados do triângulo.

Mais JavaScript

- Desvio Condicional
- Funções
- Sintaxe mínima de funções
- Funções com parâmetros
- Eventos



Comandos condicionais

- **if()**
- **if()-else**
- **switch()-case**

```
if[condition]
    statement
else
    statement
```

O comando if()

- Utilizado para executar determinado código se a condição for verdadeira
- Sintaxe:
`if(condição) {
 código que será executado
 se a condição for verdadeira
}`
- OU

`if(condição)`
UM comando que será executado se a condição for verdadeira

O comando if()

- Lembre-se que JavaScript é case-sensitive. O comando `if()` deve ser escrito em letras minúsculas



Exemplo

```
var data, hora;  
data = new Date();  
hora = data.getHours();  
if (hora<12)  
    document.write("<b>Bom dia</b>");  
if (hora>12)  
    if (hora<18)  
        document.write("<b>Boa tarde</b>");  
if (hora>18)  
    document.write("<b>Boa noite</b>");
```

O que acontece
às 12h e às 18h?

Comando if()-else

- Utilizado para executar um entre dois códigos.
- O código ligado ao else somente será executado se a condição for falsa
- Obs.: o else não tem condição
- Sintaxe:

```
if(condição) {  
    código que será executado se a condição for verdadeira  
}  
else {  
    código que será executado se a condição for falsa  
}
```

- OU

```
if(condição)  
    UM comando que será executado se a condição for verdadeira  
else  
    UM comando que será executado se a condição for falsa
```

- OU qualquer combinação das sintaxes anteriores

Exemplo:

```
var data, hora;  
data = new Date();  
hora = data.getHours();  
if (hora<12)  
    document.write("<b>Bom dia</b>");  
else  
    if (hora<18)  
        document.write("<b>Boa tarde</b>");  
    else  
        document.write("<b>Boa noite</b>");
```

O mesmo exemplo com outro formato

```
var data, hora;  
data = new Date();  
hora = data.getHours();  
if (hora<12) document.write("<b>Bom dia</b>");  
else if (hora<18) document.write("<b>Boa tarde</b>");  
else document.write("<b>Boa noite</b>");
```

Exemplos de if-else

- Exemplos:

```
if (estado=="RJ")
    cidade = "Rio de Janeiro";

if ( hora < 12 ) {
    manha = true;
    document.write ("bom dia!");
}
else {
    manha = false;
    document.write ("boa tarde!");
}
```

Erros que eu não vou fazer...

```
var x;  
// algumas linhas de código  
if(x>5) {  
    // mais linhas de código  
}  
else (x<5) {  
    // mais outras linhas de código  
}  
// últimas linhas de código
```

Erros que eu não vou fazer...

```
var x, y, z;  
x = parseFloat(prompt("Entre com um número"));  
y = parseFloat(prompt("Entre com outro número"));  
z = parseFloat(prompt("Entre com mais um número"));  
if(x>y>z) prompt("O maior número foi o primeiro");
```

Operadores Lógicos

- Concatenam operações lógicas
 - `&&` – operador E
 - `if((a > b) && (a > c))`
 - Se a maior do que b E a maior do que c
 - `||` – operador OU
 - `if((a > b) || (a > c))`
 - Se a maior do que b OU a maior do que c
 - `!` – operador NÃO
 - `if(!(a > b))`
 - Se NÃO a maior do que b

Exemplo

- Exemplos:

```
if ( hora >= 12 && hora < 18 ) {  
    manha = false;  
    document.write ("boa tarde!");  
}
```

Operador Ternário

- Substitui um if/else simples
- Use com parcimônia

```
<body>
<script type="text/javascript">
if (3 > 2) {
    alert("True");
}
else {
    alert("False");
}

(3 > 2) ? alert("True") : alert("False");
</script>
</body>
```

O Comando switch()-case

- Quando necessitamos escolher uma entre duas alternativas utilizamos o comando if ou if-else.
- Mas isto se torna extremamente trabalhoso quando existem várias alternativas para se escolher. Nestes casos utiliza-se o comando switch

O Comando switch()-case

- Sintaxe:

```
switch (expressão) {  
    case constante1:  
        comando1;  
    case constante2:  
        comando2;  
    default:  
        comandoN;  
}
```

O Comando switch()-case

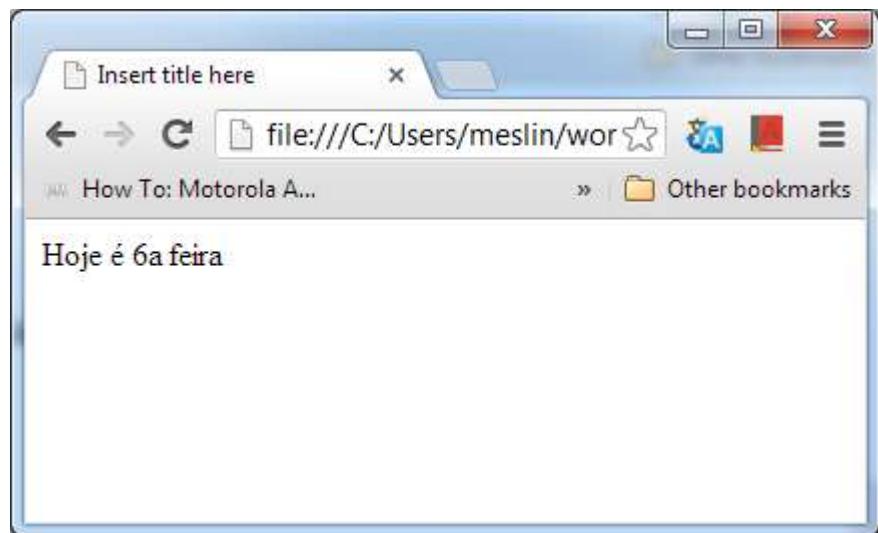
Sintaxe simplificada:

```
switch (expressão) {  
    case constante1:  
        comando1;  
        break;  
    case constante2:  
        comando2  
        break  
    case constante3:  
        comando3;  
        break;  
    default:  
        comandoN;  
}
```

Um pouco mais sofisticada

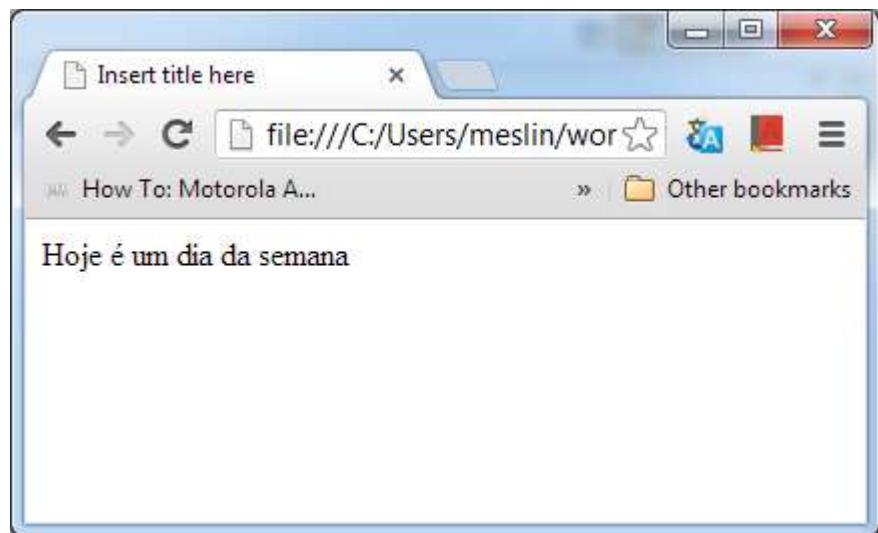
```
switch (expressão) {  
    case constante1:  
        comando1;  
        break;  
    case constante2:  
        comando2  
    case constante3:  
    case constante4:  
        comando3;  
        break;  
    default:  
        comandoN;  
}
```

Exemplo



```
var data;
data = new Date();
semana = data.getDay();
switch(semana) {
    case 1:
        document.write("Hoje é 2a feira");
        break;
    case 2:
        document.write("Hoje é 3a feira");
        break;
    case 3:
        document.write("Hoje é 4a feira");
        break;
    case 4:
        document.write("Hoje é 5a feira");
        break;
    case 5:
        document.write("Hoje é 6a feira");
        break;
    case 6:
        document.write("Hoje é sábado");
        break;
    case 0:
        document.write("Hoje é domingo");
        break;
    default:
        document.write("Hoje não é um dos dias da semana")
}
```

Exemplo



```
var data;
data = new Date();
semana = data.getDay();
switch(semana) {
    case 1:
    case 2:
    case 3:
    case 4:
    case 5:
        document.write("Hoje é um dia da semana");
        break;
    case 6:
        document.write("Hoje é sábado");
        break;
    case 0:
        document.write("Hoje é domingo");
        break;
    default:
        document.write("Hoje não é um dos dias da semana")
}
```

Exercícios



Exercícios

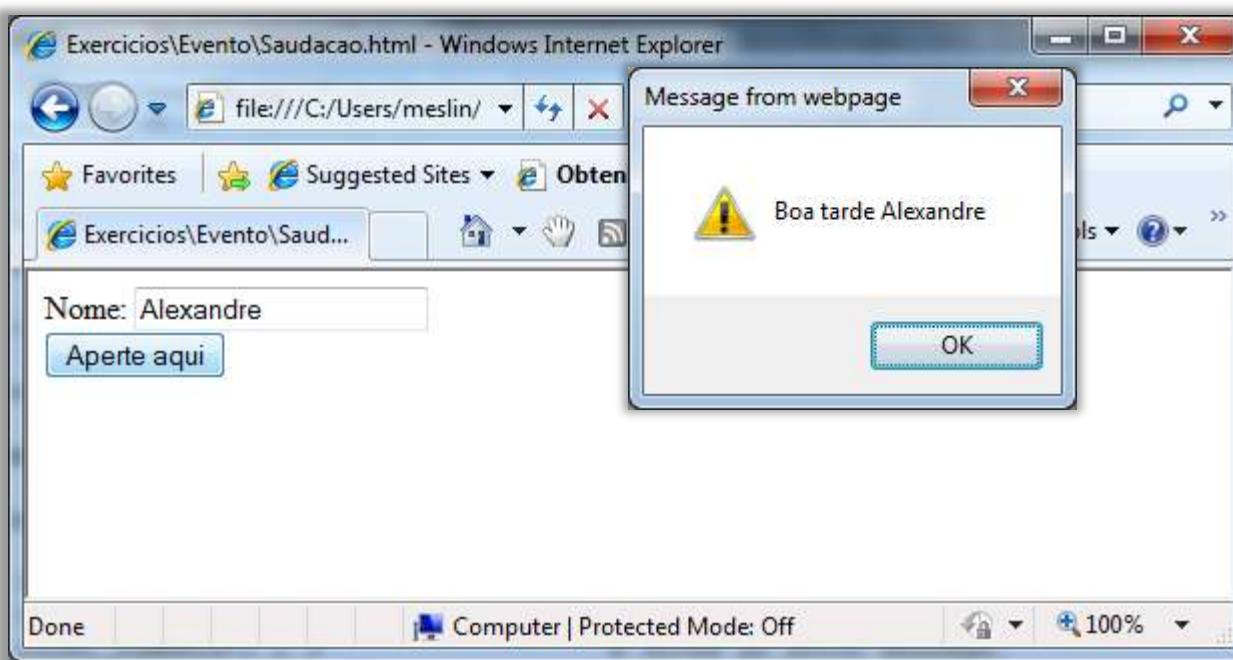
- Faça uma página HTML que leia o valor de 2 números fracionários através de comandos prompt. A página deverá informar o valor do maior número lido.

Exercícios

- Faça uma página HTML que leia o valor de 2 números fracionários através de comandos prompt. A página deverá mostrar os 2 valores em ordem crescente.
- Crie uma nova versão para 3 números
- Desafio: crie uma versão para CINCO números!

Exercícios

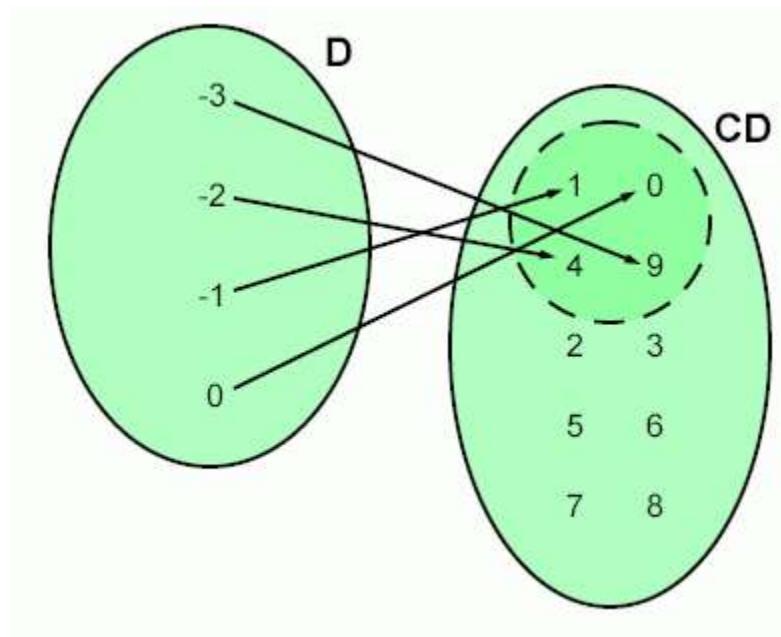
- Faça uma página com um formulário contendo um campo texto onde o usuário irá escrever o seu nome. A página deverá saudar o usuário



Número por Extenso

- Faça uma página HTML com JavaScript que solicite um número entre 0 e 9 para o usuário. A página deverá exibir o número por extenso.
- Modifique o programa para exibir a mensagem "número inválido" caso o usuário digite um número inválido
- Faça uma versão usando if()-else
- Faça uma versão usando switch()-case

Funções



Função

- Uma função é uma porção de código que resolve um problema muito específico, parte de um problema maior (Wikipédia)
- Uma função contém código que será executado por um evento ou uma chamada explícita
- Você pode chamar uma função de qualquer lugar de uma página
- Funções podem ser definidas na seção <head> ou <body>
- Para garantir que a função já foi carregada antes de sua chamada, a função deve ser definida na seção <head>

Definição de Função

- Sintaxe:

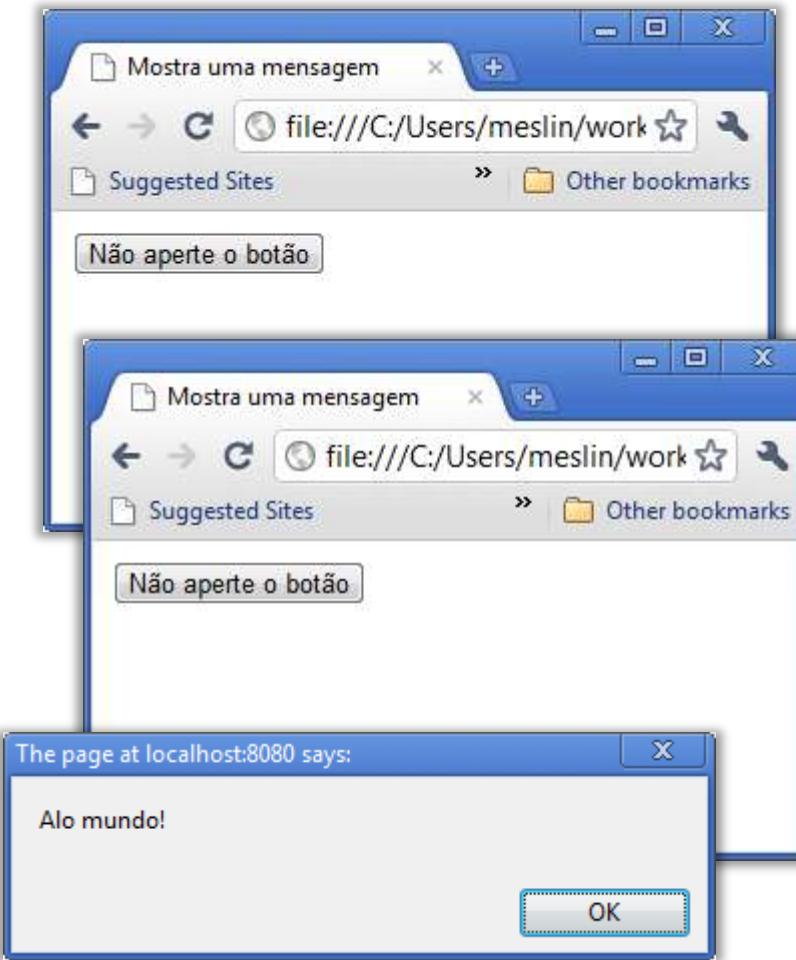
```
function nomeDaFuncao(var1, var2, ..., varN) {  
    código executável  
}
```

Lista de variáveis,
separadas por vírgula

- Os parâmetros var1, var2, etc. são variáveis ou valores passados para a função
- Os { e } definem o início e o fim da função
- Nota: uma função sem parâmetros precisa dos () depois do nome

Exemplo

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function mostraMensagem() {
    alert("Alo mundo!");
}
</script>
<title>Mostra uma mensagem</title>
</head>
<body>
<form>
<input
    type="button"
    value="Não aperte o botão"
    onclick="mostraMensagem()">
</form>
</body>
</html>
```



O comando return

- O comando return é utilizado para especificar um valor que será retornado da função chamada para quem a chamou

- Sintaxe:

```
return; // apenas retorna da função
```

- OU

```
return valor; // retorna um valor da função
```

Exemplo de uso de return

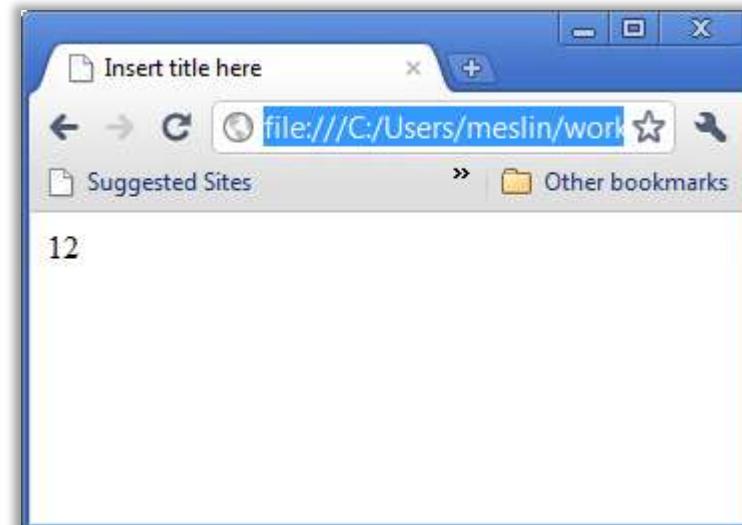
```
function produto(a,b) {  
    var c;  
    c = a * b;  
    return c; // poderia ser return a * b;  
}  
  
result = produto(4, 3);  
document.write(result);
```



Exemplo de uso de return

```
function produto(a,b) {  
    var c;  
    c = a * b;  
    return c; // poderia ser return a * b;  
}
```

```
document.write(produto(4,3));
```



Ciclo de vida de uma variável

- Variável declarada dentro de uma função
 - Variável local
 - Somente pode ser acessada dentro da função
 - Se a função chamar outra função, a variável continua existindo, mas a função chamada não tem acesso a ela
 - Podemos ter variáveis locais com o mesmo nome em funções diferentes
 - A variável é destruída ao término da função
- Variáveis declaradas fora da função
 - Variáveis globais
 - Podem ser acessada em qualquer parte do código, inclusive dentro de outras funções
 - A variável passa a existir depois da sua declaração e continua existindo até a página ser fechada

Atenção!!!

- Variáveis criadas sem o uso de `var` são sempre globais, mesmo quando criadas dentro de uma função!



Exemplo do uso de var

```
<html>
<head>
<script type="text/javascript">
function executaFuncao() {
    var numero = 8752;
    document.write("O valor do número no início da função é " + numero + "<br/>");
    numero += 2;
    document.write("O valor do número no final da função é " + numero + "<br/>");
}
</script>
</head>
<body>
<script type="text/javascript">
var numero = 5;
document.write("O valor inicial do número é " + numero + "<br/>");
numero++;
document.write("O valor do número antes da chamada da função é " + numero +
"<br/>");
executaFuncao();
document.write("O valor do número depois da chamada da função é " + numero +
"<br/>");
</script>
</body>
</html>
```

```
<html>
<head>
<script type="text/javascript">
function executaFuncao() {
    numero = 8752;
    document.write("O valor do número no início da função é " + numero + "<br/>");
    numero += 2;
    document.write("O valor do número no final da função é " + numero + "<br/>");
}
</script>
</head>
<body>
<script type="text/javascript">
var numero = 5;
document.write("O valor inicial do número é " + numero + "<br/>");
numero++;
document.write("O valor do número antes da chamada da função é " + numero +
"<br/>");
executaFuncao();
document.write("O valor do número depois da chamada da função é " + numero +
"<br/>");
</script>
</body>
</html>
```



Funções Predefinidas

■ isNaN (valor)

- Retorna "true" se o valor não for numérico
- Exemplo:

```
x =prompt("Entre um numero:", " ");
if (isNaN(x))
    window.alert("Valor não é numérico !");
```

■ window.confirm (pergunta)

- Abre uma janela para pedir uma string ao usuário
- Exemplo:

```
if (confirm("Quer realmente sair da página ?"))
    alert("Então adeus!");
```

Algumas Funções Interessantes...

- `variável = setTimeout(function,milliseconds,param1,param2,...)`
 - Executa a função depois de x milisegundos
- `window.clearTimeout(variável);`
 - Aborta o timeout de variável
- `setInterval(function,milliseconds,param1,param2,...)`
 - Executa a função a cada x milisegundos
- `variável = window.clearInterval(variável)`
 - Aborta o tempo do intervalo

Erros que eu não vou fazer...

```
<!DOCTYPE html>
<html>
<head>
<title>Função</title>

<script type="text/javascript">
function funcao(variavel) {
// algum código
}
</script>

</head>
<body>

<script type="text/javascript">
// mais algum código
funcao(valor1, valor2);
</script>

</body>
</html>
```

Eventos

HANDLE



Eventos

- Eventos são ações que ocorrem como resultado de atividades do browser ou interações do usuário com a página Web
 - Uso do mouse – clique, duplo clique, movimento, ...
 - Entrada de dados – apertar uma tecla, digitar, soltar uma tecla, ...
 - Carga de uma página ou figura, ...
 - Envio de formulário
 - ...

Manipuladores de Eventos

- Quando um evento ocorre, o segmento de código que é executado em resposta ao evento específico é chamado de manipulador do evento (handler do evento)
- Formato geral:

```
<TagHTML evento="código JavaScript">
```

Alguns Eventos Definidos em JavaScript

Mouse Events

Property	Description	DOM
onclick	The event occurs when the user clicks on an element	2
ondblclick	The event occurs when the user double-clicks on an element	2
onmousedown	The event occurs when a user presses a mouse button over an element	2
onmousemove	The event occurs when the pointer is moving while it is over an element	2
onmouseover	The event occurs when the pointer is moved onto an element	2
onmouseout	The event occurs when a user moves the mouse pointer out of an element	2
onmouseup	The event occurs when a user releases a mouse button over an element	2

Alguns Eventos Definidos em JavaScript

Keyboard Events

Attribute	Description	DOM
onkeydown	The event occurs when the user is pressing a key	2
onkeypress	The event occurs when the user presses a key	2
onkeyup	The event occurs when the user releases a key	2

Alguns Eventos Definidos em JavaScript

Frame/Object Events

Attribute	Description	DOM
onabort	The event occurs when an image is stopped from loading before completely loaded (for <object>)	2
onerror	The event occurs when an image does not load properly (for <object>, <body> and <frameset>)	
onload	The event occurs when a document, frameset, or <object> has been loaded	2
onresize	The event occurs when a document view is resized	2
onscroll	The event occurs when a document view is scrolled	2
onunload	The event occurs once a page has unloaded (for <body> and <frameset>)	2

Alguns Eventos Definidos em JavaScript

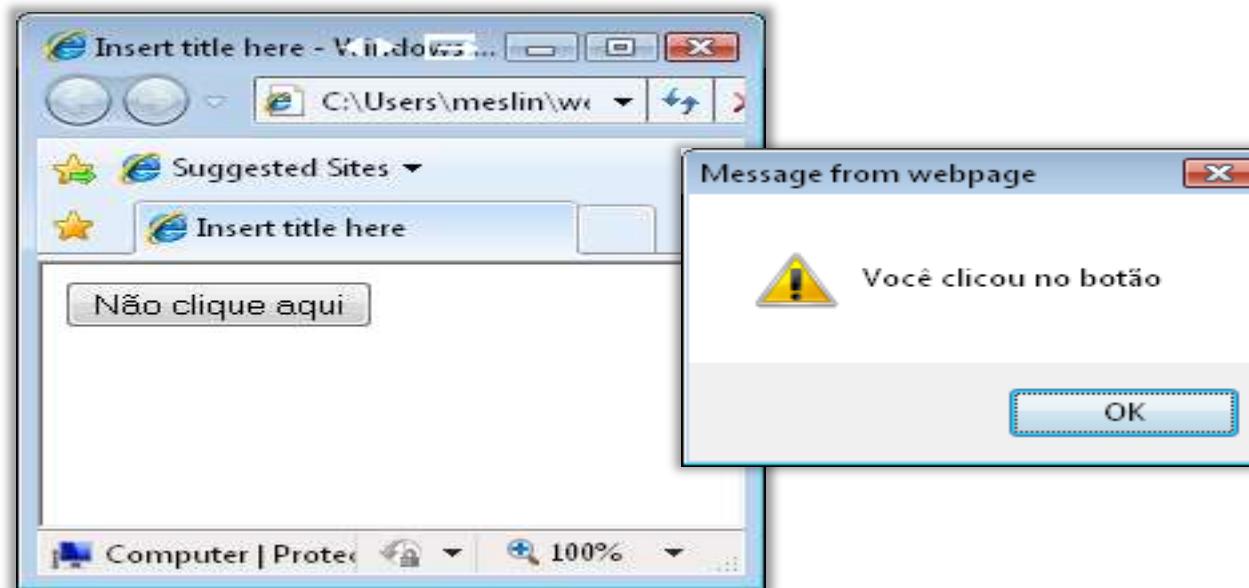
Form Events

Attribute	Description	DOM
onblur	The event occurs when a form element loses focus	2
onchange	The event occurs when the content of a form element, the selection, or the checked state have changed (for <input>, <select>, and <textarea>)	2
onfocus	The event occurs when an element gets focus (for <label>, <input>, <select>, <textarea>, and <button>)	2
onreset	The event occurs when a form is reset	2
onselect	The event occurs when a user selects some text (for <input> and <textarea>)	2
onsubmit	The event occurs when a form is submitted	2

Um pouco sobre eventos

Código JavaScript
(sem as tags script)

```
<!DOCTYPE html>
<html>
<head>
<title>Evento/onClick/click</title>
</head>
<body>
<form>
<input type="button" value="Não clique aqui" onclick='alert("Você clicou no botão")'>
</form>
</body>
</html>
```



$$1+1=2$$



Mais um pouco sobre eventos...

```
<!DOCTYPE html>
<html>
<head>
<title>Exemplo de evento onclick com função</title>

<script type="text/javascript">
function advertencia() {
    return confirm("Aluno de JavaScript do Alexandre?");
}
</script>
</head>
<body>

<a href='http://cursos.meslin.com.br' onclick="return advertencia();">
Acesso somente para alunos de JavaScript
</a>

</body>
</html>
```

Mais um pouco ainda sobre eventos...

```
<!DOCTYPE html>
<html>

<head>
<title>Exemplo de evento onload</title>
</head>

<body onload="alert('Bem vindo, usuário')" onunload="alert('Obrigado por visitar a
página')">

<p>Exemplo de eventos onload e onunload.</p>

</body>
</html>
```

Quando cada código abaixo irá ser executado?

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function f1() {
    var a, b, c
    a = 8;
    b = 7
    c = a + b;
    alert ("Resultado = " + c);
}
function f2() {
    var a, b, c
    a = 8;
    b = 7
    c = a + b;
    alert ("Resultado = " + c);
}
function f3() {
    var a, b, c
    a = 8;
    b = 7
    c = a + b;
    alert ("Resultado = " + c);
}
```

```
<body>
<script type="text/javascript">
var a, b, c
a = 8;
b = 7
c = a + b;
alert ("Resultado = " + c);
</script>

<script type="text/javascript">
f3();
</script>

<button onclick="f1()">Aperte aqui</button>
</body>
</html>
```

$$1+1=2$$



Eventos e Formulários

```
<!DOCTYPE html>
<html>
<head>
<title>Exemplo de evento onclick com formulário</title>

<script type="text/javascript">
function naoClique() {
    alert('Eu falei para não clicar!');
}
</script>

</head>
<body>
<form>
<input type="button" value="Não clique aqui." onclick="naoClique()" />
</form>
</body>
</html>
```

Descobrindo o objeto do evento

```
/*
 * util.js
 * getObjetoAtivado
 * recebe o evento que foi gerado
 * retorna o objeto que recebeu o evento
 */
function getObjetoAtivado(evento) {
    var objeto;
    if(!evento)
        // versões antigas do IE
        objeto = window.event.srcElement;
    else if(evento.srcElement)
        // IE7 ou mais recente
        objeto = evento.srcElement;
    else
        // DOM 2 (quase todos os browsers atuais, incluindo MS)
        objeto = evento.target;
    return objeto;
}
```

O Objeto event

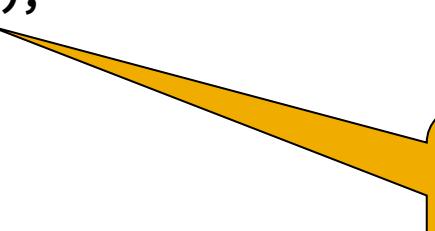
```
isTrusted = true
screenX = 1417
screenY = 124
clientX = 51
clientY = 39
ctrlKey = false
shiftKey = false
altKey = false
metaKey = false
button = 0
buttons = 0
relatedTarget = null
pageX = 51
pageY = 39
x = 51
y = 39
offsetX = 51
offsetY = 39
movementX = 0
movementY = 0
fromElement = null
toElement = [object HTMLButtonElement]
which = 1
layerX = 51
layerY = 39
getModifierState = function getModifierState() {
[native code]}
```

```
initMouseEvent = function initMouseEvent() {
[native code] }
view = [object Window]
detail = 1
initUIEvent = function initUIEvent() { [native code] }
sourceCapabilities =[object
InputDeviceCapabilities]
type = click
target = [object HTMLButtonElement]
currentTarget = [object HTMLButtonElement]
eventPhase = 2
bubbles = true
cancelable = true
defaultPrevented = false
timeStamp = 1381.9450000000002
path = [object HTMLButtonElement],[object
HTMLBodyElement],[object
HTMLHtmlElement],[object
HTMLDocument],[object Window]
srcElement = [object HTMLButtonElement]
returnValue = true
cancelBubble = false
stopPropagation = function stopPropagation() {
[native code] }
```

```
stopImmediatePropagation = function
stopImmediatePropagation() { [native code] }
preventDefault = function preventDefault() { [native
code] }
initEvent = function initEvent() { [native code] }
NONE = 0
CAPTURING_PHASE = 1
AT_TARGET = 2
BUBBLING_PHASE = 3
MOUSEDOWN = 1
MOUSEUP = 2
MOUSEOVER = 4
MOUSEOUT = 8
MOUSEMOVE = 16
MOUSEDRAG = 32
CLICK = 64
DBLCLICK = 128
KEYDOWN = 256
KEYUP = 512
KEYPRESS = 1024
DRAGDROP = 2048
FOCUS = 4096
BLUR = 8192
SELECT = 16384
CHANGE = 32768
```

Um pouco sobre objetos

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function funcao() {
    alert(document.formulario.texto.value);
}
</script>
<title>Insert title here</title>
</head>
<body>
<form name="formulario">
<input type="text" name="texto">
<input type="button" value="Não clique aqui!" onclick="funcao()">
</form>
</body>
</html>
```

- 
- O documento tem um formulário
 - O formulário tem um campo texto
 - O campo texto tem um valor

Explicando...

- Observe a construção HTML:

```
<form name="formulario">  
<input type="text" name="texto">  
</form>
```

- Neste caso, não existe a variável texto, apenas o campo chamado texto do formulário, ou seja, fazer algo do tipo:

alert(texto)

- não é possível, mas podemos escrever:

alert(document.formulario.texto.value)

- Escreve o valor (value) do campo chamado texto que está no <form> chamado formulario dentro do documento (document)

Outros atributos de campo (além de value)

```
<form name="formulario">
<input type="text" name="texto">
<input type="button" value="Não clique aqui!" onclick="funcao()">
</form>
```

■ **document.formulario.texto.value**

Propriedade	Descrição
defaultvalue	O valor inicialmente mostrado no campo
form	Referencia o formulário onde o campo está
maxlength	Número máximo de caracteres permitidos no campo
name	O nome do campo
size	O tamanho do campo em caracteres (aproximadamente)
type	O tipo do campo (button, checkbox, text, hidden, etc.)
value	O valor atual do campo

Continuação

- Para podermos fazer alguma operação com uma variável devemos copiar o campo chamado texto para uma variável.

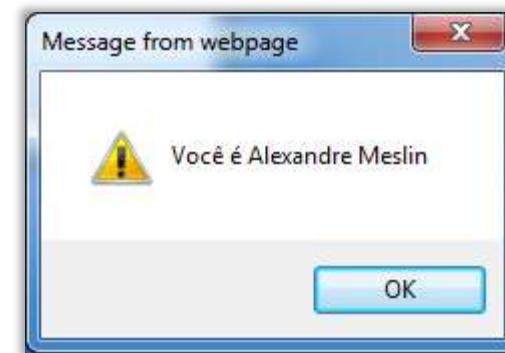
```
<form name="formulario">  
<input type="text" name="texto">  
</form>
```

- Exemplo:

```
variavel = document.formulario.texto.value
```

Exemplo (escreve nome e sobrenome)

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function pessoa() {
    // converte o campo nomeBastimo para variável nome
    var nome = document.familia.nomeBatismo.value;
    // converte o campo nomeFamilia para variável sobrenome
    var sobrenome = document.familia.nomeFamilia.value;
    // escreve uma string constante: "Você é "
    // concatena com uma string variável: nome
    // concatena com uma string constante: " "
    // concatena com uma string variável: sobrenome
    alert("Você é " + nome + " " + sobrenome);
}
</script>
<title>Nome & Sobrenome</title>
</head>
<body>
<form name="familia">
    Nome: <input type="text" name="nomeBatismo"><br/>
    Sobrenome: <input type="text" name="nomeFamilia"><br/>
    <input type="button" value="Quem é você..." onclick="pessoa()">
</form>
</body>
</html>
```



Exemplo – campo text ou password

Propriedade	Descrição
defaultValue	O valor inicialmente mostrado
form	Referência ao formulário
maxlength	Número máximo de caracteres
name	Nome do campo
size	Tamanho do campo em caracteres
type	Tipo do campo <input>
Value	O valor atual do campo
Método	Descrição
blur()	Remove o foco do campo
focus()	Coloca o campo em foco
select()	Seleciona o campo

Exemplo – campo text ou password

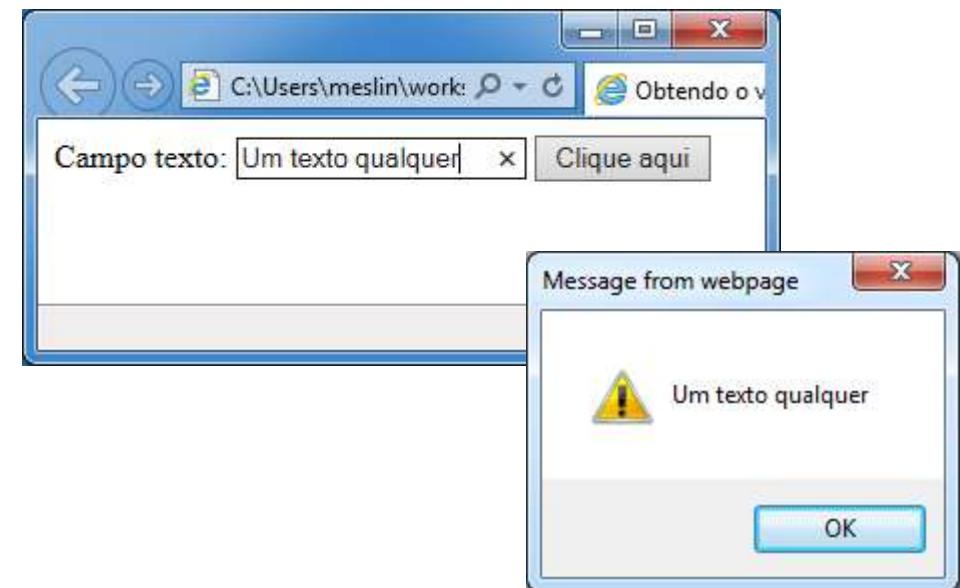
```
<html>
<head>
<title>Obtendo o valor de um campo texto de um formulário</title>

<script type="text/javascript">
function mostraValor() {
    alert(document.formulario.campoTexto.value);
}
</script>

</head>
<body>

<form name="formulario">
Campo texto: <input type="text" name="campoTexto"/>
<input type="button" onclick="mostraValor()" value="Clique aqui"/>
</form>

</body>
</html>
```

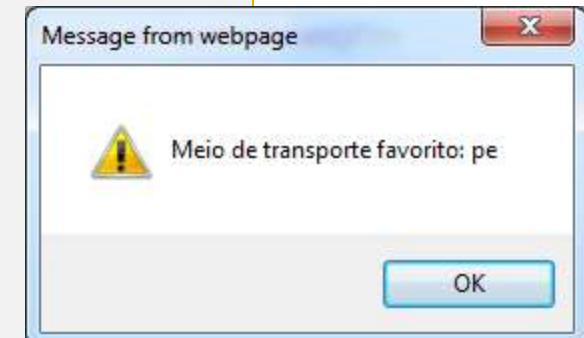
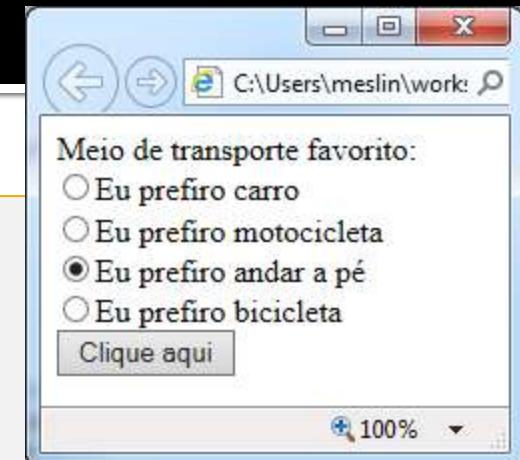


Exemplo de campo checkbox

Propriedade	Descrição
checked	Valor booleano indicando se o campo está atualmente selecionado
defaultChecked	Valor booleano indicando se o campo é selecionado por default
name	Nome do campo
value	Valor do campo

Exemplo de campo radio

```
<html>
<head>
<title>Obtendo o valor de um campo radio de um formulário</title>
<script type="text/javascript">
function mostraValor() {
    for(var i=0; i<document.formularioTransporte.campoMeio.length; i++)
        if(document.formularioTransporte.campoMeio[i].checked)
            alert("Meio de transporte favorito: " + document.formularioTransporte.campoMeio[i].value);
}
</script>
</head>
<body>
<form name="formularioTransporte">
Meio de transporte favorito:<br />
<input type="radio" value="carro" name="campoMeio" />Eu prefiro carro<br />
<input type="radio" value="moto" name="campoMeio" />Eu prefiro motocicleta<br />
<input type="radio" value="pe" name="campoMeio" />Eu prefiro andar a pé<br />
<input type="radio" value="bicicleta" name="campoMeio" />Eu prefiro bicicleta<br />
<input type="button" value="Clique aqui" onclick="mostraValor()" />
</form>
</body>
</html>
```



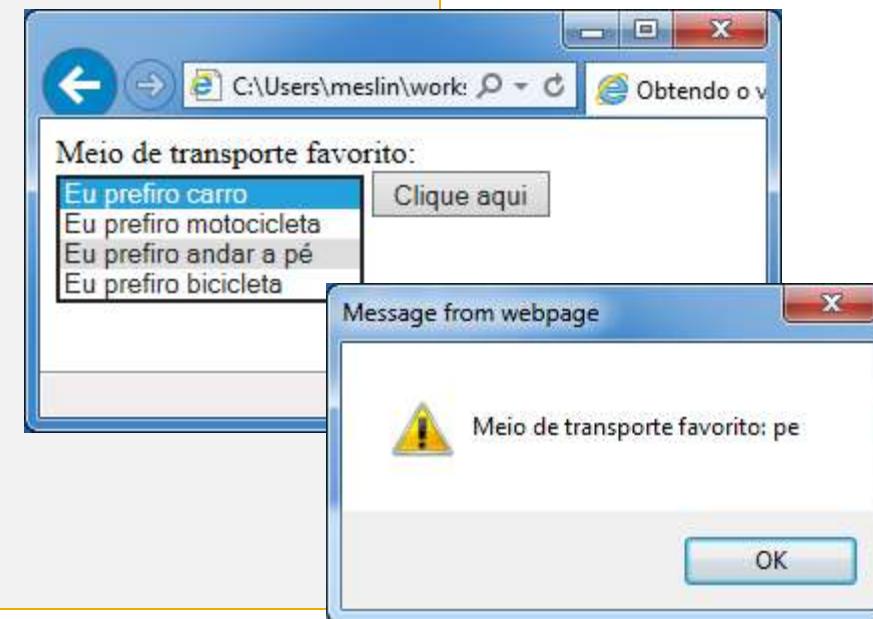
Exemplo – campo select (simples ou múltiplo)

	Propriedade	Descrição
<select>	length	Número de opções na lista
	name	Nome da lista
	options	Lista de opções
	selectedIndex	Índice do item selecionado da lista
<option>	defaultSelected	Valor booleano indicando se o item é selecionado por default
	index	O valor do índice da opção
	selected	Valor booleano indicando se o item está selecionado
	text	O texto associado à opção
	value	O valor associado à opção

Exemplo – campo select (simples ou múltiplo)

```
<html>
<head>
<title>Obtendo o valor de um campo select de um formulário</title>
<script type="text/javascript">
function mostraValor()
{
    var i;
    for(i=0; i<document.formularioTransporte.campoMeio.length; i++)
        if(document.formularioTransporte.campoMeio.options[i].selected)
            alert("Meio de transporte favorito: " + document.formularioTransporte.campoMeio.options[i].value);
}
</script>
</head>
<body>

<form name="formularioTransporte">
Meio de transporte favorito:<br />
<select name='campoMeio'>
    <option value="carro">Eu prefiro carro</option>
    <option value="moto">Eu prefiro motocicleta</option>
    <option value="pe">Eu prefiro andar a pé</option>
    <option value="bicicleta">Eu prefiro bicicleta</option>
</select>
<input type="button" value="Clique aqui" onclick="mostraValor()" />
</form>
</body>
</html>
```



Exemplo de Drag-and-Drop

```
<html>
<head>

<style type="text/css">
.retangulo {
    border: 1px solid #AAAAAA;
    width: 400px;
    height: 100px;
}
</style>

<script type="text/javascript">
function eventoDrop(evento) {
    var idBloco;
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragOver(evento) {
    evento.preventDefault();
}

function eventoDragStart(evento) {
    evento.dataTransfer.setData("bloco", evento.target.id);
}
</script>
</head>

<body>
<p>
    Mova a imagem e o texto de um retângulo para o outro
</p>

<div class="retangulo"
    ondrop="eventoDrop(event)"
    ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        draggable="true"
        ondragstart="eventoDragStart(event)">
            <br/>
            Eclipse
        </div>
    </div>

    <div class="retangulo">
        Aqui não, no outro!
    </div>

    <div class="retangulo"
        ondrop="eventoDrop(event)"
        ondragover="eventoDragOver(event)">
    </div>
</body>
</html>
```

Exemplo de Drag-and-Drop

`draggable="true"`

Permite que o objeto sofra a ação de drag

```
<style type="text/css">
.retangulo {
  border: 1px solid #AAAAAA;
  width: 400px;
  height: 100px;
}
</style>

<script type="text/javascript">
function eventoDrop(evento) {
  var idBloco;
  evento.preventDefault();
  idBloco = evento.dataTransfer.getData("bloco");
  evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragOver(evento) {
  evento.preventDefault();
}

function eventoDragStart(evento) {
  evento.dataTransfer.setData("bloco", evento.target.id);
}
</script>
</head>
```

```
<body>
<div id='idBloco' style="text-align: center;" 
      draggable="true"
      ondragstart="eventoDragStart(event)">
  <br/>
  Eclipse
</div>
<div id='idBloco' style="text-align: center;" 
      draggable="true"
      ondragstart="eventoDragStart(event)">
  <br/>
  Eclipse
</div>
<div class="retangulo">
  Aqui não, no outro!
</div>
<div class="retangulo"
      ondrop="eventoDrop(event)"
      ondragover="eventoDragOver(event)">
</div>
</body>
</html>
```

Exemplo de Drag-and-Drop

```
<html>
<head>
<style>
    .retangulo {
        width: 150px;
        height: 100px;
        background-color: #ccc;
        border: 1px solid black;
        padding: 10px;
    }
</style>

<script type="text/javascript">
function eventoDrop(evento) {
    var idBloco;
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragStart(evento) {
    evento.preventDefault();
    evento.dataTransfer.setData("bloco", evento.target.id);
}
</script>
</head>
```

Salva o id do objeto que irá sofrer a ação de drag

```
<body>
<div id='idBloco' style="text-align: center;" draggable="true"
      ondragstart="eventoDragStart(event)">
    <br/>
    Eclipse
</div>
<div id='idBloco' style="text-align: center;" draggable="true"
      ondragstart="eventoDragStart(event)">
    <br/>
    Eclipse
</div>
<div class="retangulo">
    Aqui não, no outro!
</div>
<div class="retangulo"
      ondrop="eventoDrop(event)"
      ondragover="eventoDragOver(event)">
</div>
</body>
</html>
```

Exemplo de Drag-and-Drop

```
<html>
<head>

<style type="text/css">
.retangulo {
    border: 1px solid #AAAAAA;
    width: 400px;
    height: 100px;
}
</style>

<script type="text/javascript">
function eventoDrop(evento) {
    var idBloco;
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
    evento.preventDefault();
}

function eventoDragOver(evento) {
    evento.preventDefault();
}

function eventoDragStart(evento) {
    evento.dataTransfer.setData("bloco", evento.target.id);
}
</script>
</head>
```

Normalmente somente um elemento editável pode ser destino de um drag&drop.

Cancelar o comportamento permite que objetos containers sejam destino também.

```
<body>
<p>
    Mova a imagem e o texto de um retângulo para o outro
</p>

<div class="retangulo"
    ondrop="eventoDrop(event)"
    ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        draggable="true"
        ondragstart="eventoDragStart(event)">
            <br/>
<div class="retangulo"
    ondrop="eventoDrop(event)"
    ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        Eclipse
    </div>
</div>
</div>
```

```
<div class="retangulo"
    ondrop="eventoDrop(event)"
    ondragover="eventoDragOver(event)">
</div>
</body>
</html>
```

Exemplo de Drag-and-Drop

Evita o comportamento padrão do drop que é abrir como link

Coloca o novo objeto como filho do container onde o drop foi realizado

```
<html>
<head>

<style type="text/css">
.retangulo {
    border: 1px solid #AAAAAA;
    width: 100px;
    height: 100px;
}

</style>

<script>
function eventoDrop(evento)
{
    var idBloco;
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragOver(evento) {
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragStart(evento) {
    evento.dataTransfer.setData("bloco", evento.target.id);
}

</script>
</head>
```

```
<body>
<p>
    Mova a imagem e o texto de um retângulo para o outro
</p>

<div class="retangulo"
      ondrop="eventoDrop(event)"
      ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        draggable="true"
        ondragstart="eventoDragStart(event)">
        <br/>
    </div>
</div>
<div class="retangulo"
      ondrop="eventoDrop(event)"
      ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        <br/>
    </div>
</div>

</body>
</html>
```

Exemplo de Drag-and-Drop

```
<html>
<head>

<style type="text/css">
.retangulo {
    border: 1px solid #AAAAAA;
    width: 400px;
    height: 100px;
}
</style>

<script type="text/javascript">
function eventoDrop(evento) {
    var idBloco;
    evento.preventDefault();
    idBloco = evento.dataTransfer.getData("bloco");
    evento.target.appendChild(document.getElementById(idBloco));
}

function eventoDragOver(evento) {
    evento.preventDefault();
}

function eventoDragStart(evento) {
    evento.dataTransfer.setData("bloco", evento.target.id);
}
</script>
</head>

<body>
<p>
    Mova a imagem e o texto de um retângulo para o outro
</p>

<div class="retangulo"
    ondrop="eventoDrop(event)"
    ondragover="eventoDragOver(event)">
    <div id='idBloco' style="text-align: center;">
        draggable="true"
        ondragstart="eventoDragStart(event)">
            <br/>
            Eclipse
        </div>
    </div>

    <div class="retangulo">
        Aqui não, no outro!
    </div>

    <div class="retangulo"
        ondrop="eventoDrop(event)"
        ondragover="eventoDragOver(event)">
    </div>
</body>
</html>
```

Perguntas?



Exercícios



Pura matemática (mas feito em JavaScript)

- Considere a função $f(x) = 2x + 5$
- Calcule manualmente:
 - $f(1)$
 - $f(8)$
 - $f(7)$
 - $f(5)$
 - $f(2)$

Pura matemática (mas feito em JavaScript)

- Considere a função $f(x, y) = 2x + 5y + 7$
- Calcule manualmente:
 - $f(1, 8)$
 - $f(8, 7)$
 - $f(7, 5)$
 - $f(5, 2)$
 - $f(2, 0)$

Exercício

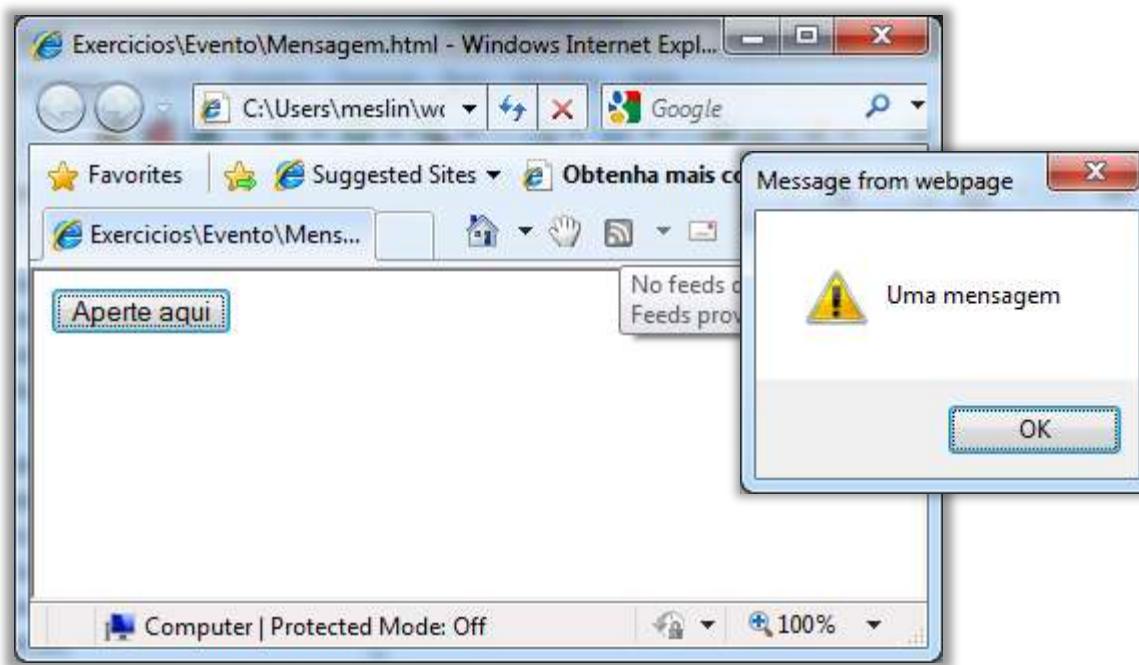
- Faça uma página HTML onde o usuário poderá entrar com o valor de x . Mostre o resultado de $f(x) = 2x + 5$ (use uma função feita por você)

Exercício

- Faça uma página HTML onde o usuário poderá entrar com os valores de x e y. Mostre o resultado de $f(x, y) = 2x + 5y + 7$ (use uma função feita por você).

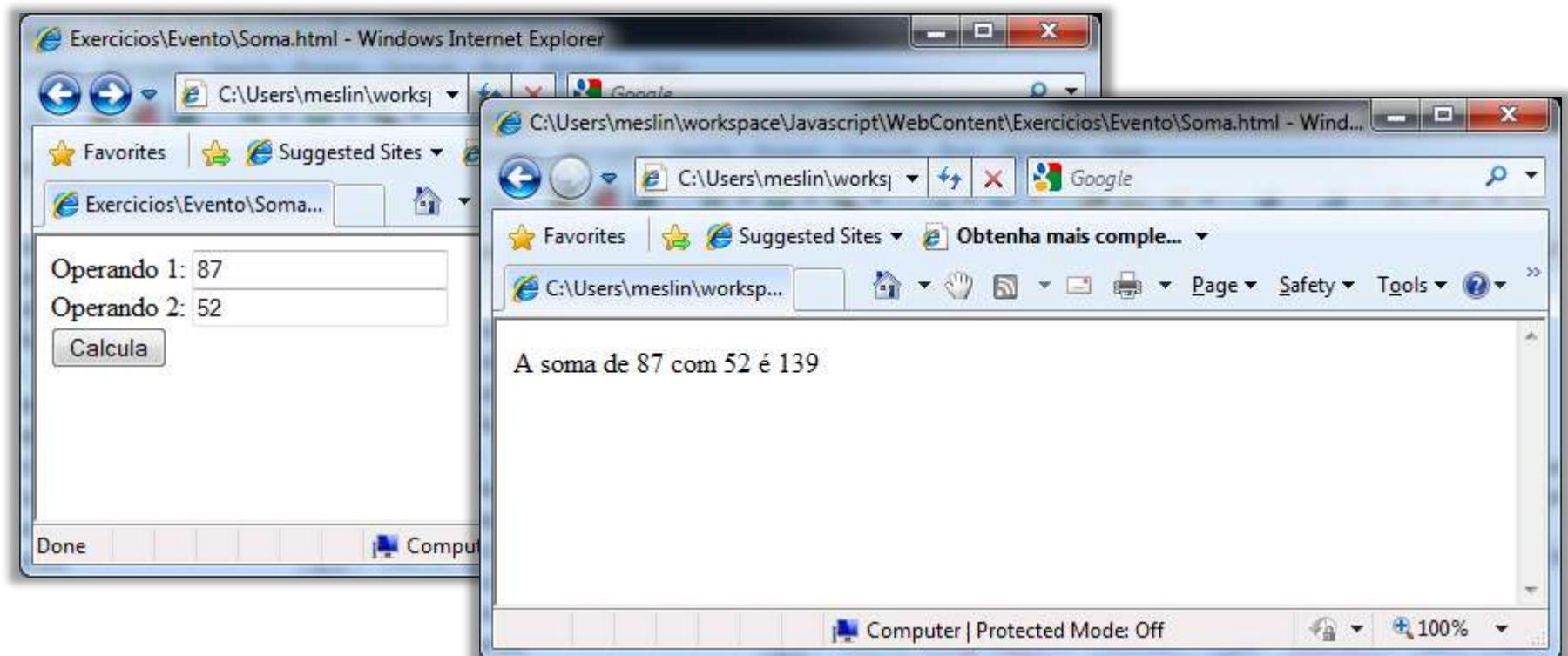
Exercícios

- Faça uma página HTML contendo um formulário com um botão. A página deverá exibir uma mensagem quando o usuário clicar no botão.



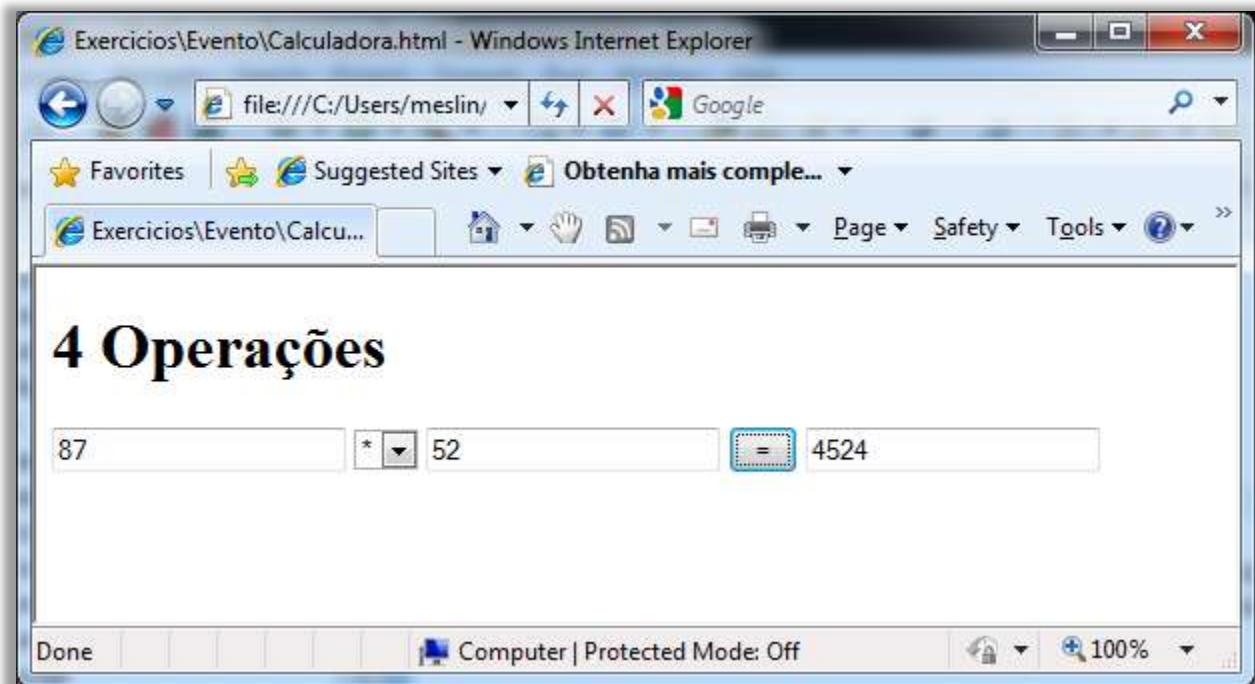
Exercícios

- Faça uma página com um formulário contendo dois campos texto. A página deverá exibir a soma dos valores digitados nos campos



Desafio

- Fazer uma página HTML contendo um formulário com quatro campos: operando1, operador (+-* /), operando2 e resultado e um botão de igual.



Exercícios

- Faça uma página HTML que leia o valor de 2 números fracionários através de comandos prompt. A página deverá informar o valor do maior número lido.
- Faça 4 versões diferentes desta página:
 - 1) Utilize uma função que leia os 2 números e escreva o maior.
 - 2) Utilize uma função que leia os 2 números e retorne o maior (esta função não escreva nada).
 - 3) Utilize uma função que receba os 2 números e escreva o maior (esta função não lê nada).
 - 4) Utilize uma função que receba os 2 números e retorne o maior (esta função não lê e não escreve nada).

Exercícios

- Complete a página HTML a seguir:

```
<!DOCTYPE html>
<html>
<head>
<title>Quadrado</title>
</head>
<body>
<script type="text/javascript">
var i, q;
i = parseFloat(prompt("Informe um número"));
q = quad (i);
alert(i + " ao quadrado = " + q);
</script>
</body>
</html>
```

- Modifique a sua página para não utilizar a variável q

Exercícios

- Complete a página HTML a seguir:

```
<!DOCTYPE html>
<html>
<head>
<title>Exercício de Função</title>
</head>
<body>
<script type="text/javascript">
var n1, n2, n3, aux, npares, tot;
n1 = parseFloat(prompt("Entre com o primeiro número"));
n2 = parseFloat(prompt("Entre com o o segundo número"));
n3 = parseFloat(prompt("Entre com o terceiro número"));
aux = maior (n1, n2);
alert("O maior numero lido = " + maior(aux, n3));
alert("Media dos numeros lidos = " + media (n1, n2, n3));
npares = par(n1) + par(n2) + par(n3);
alert("Total de numeros pares lidos = " + npares);
tot = media(n1, n2, n3);
alert("Total de valores acima da media = " + total(n1, n2, n3, tot));
</script>
</body>
</html>
```

Exercícios

- Defina uma função chamada maximo() que recebe 2 argumentos como parâmetro e retorna o maior deles. Faça uma página para testar a sua função.

Exercícios

- Defina uma função chamada maiorDe3() que recebe 3 números como parâmetros e retorna o maior deles.
- Faça uma página para testar a sua função

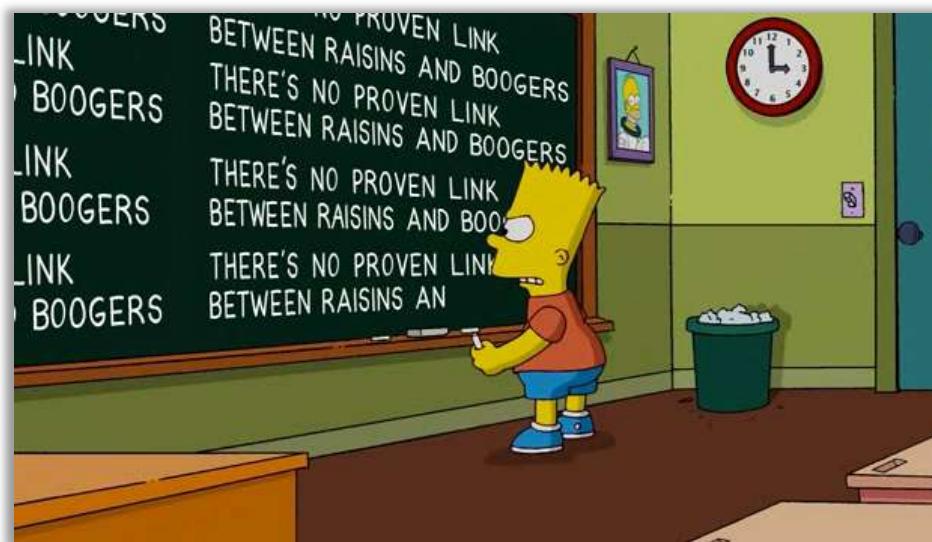
Aula



- Laço while
- Laço for
- Comando break
- Comando continue

Repetição

- Permite a execução repetida de um ou mais comandos
 - Repetição determinada
 - Repetição indeterminada

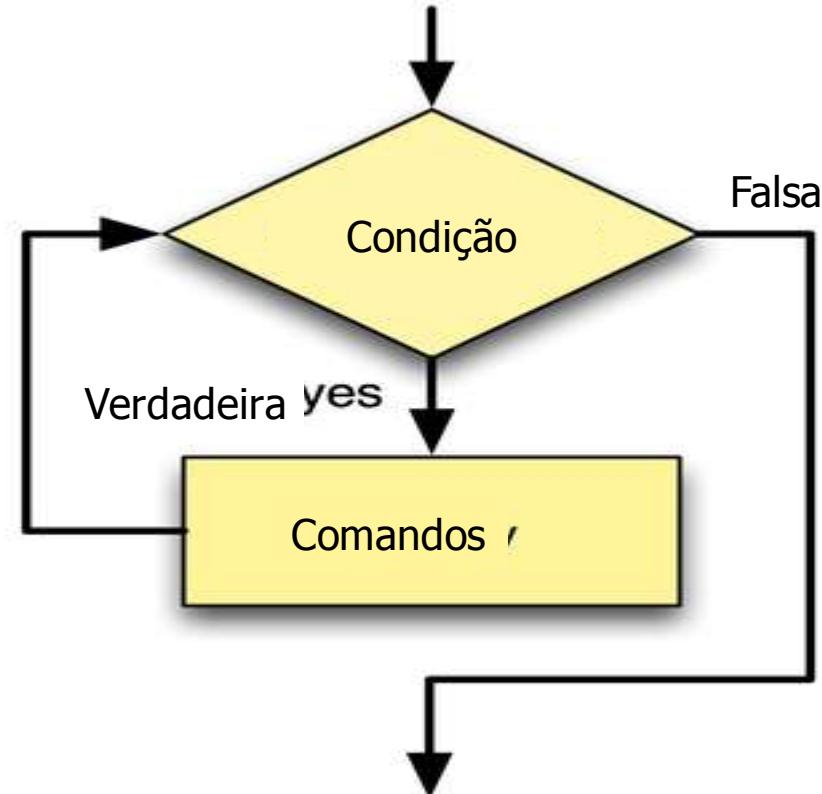


Laços while

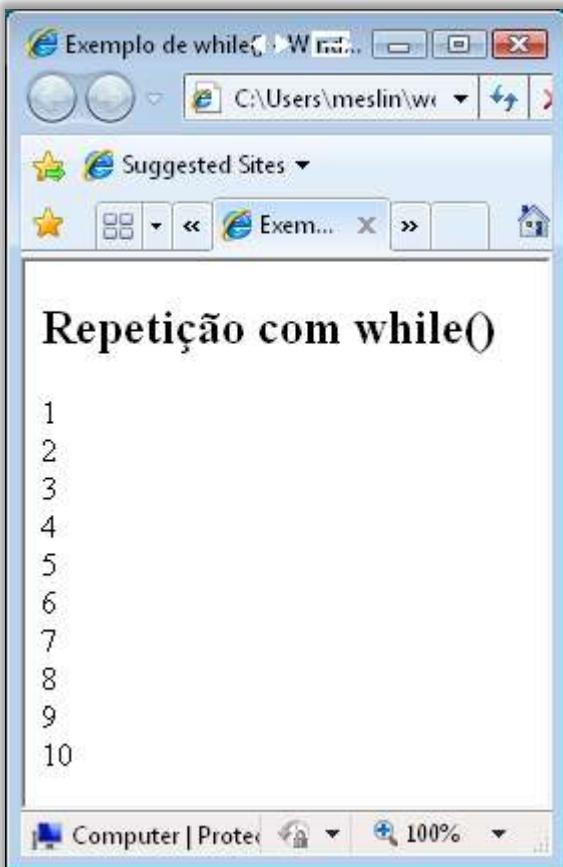
- Permite repetir um comando ou bloco enquanto uma condição for verdadeira

- Forma geral:

```
while(condição) {  
    comandos;  
}
```



Exemplo

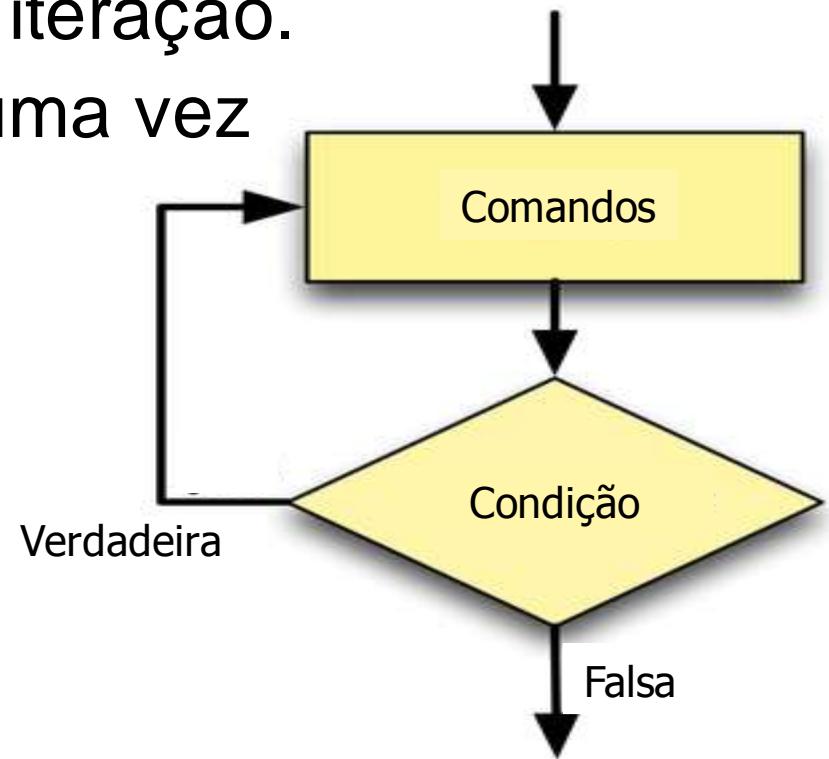


```
var i;  
i = 1;  
while(i <= 10) {  
    document.write(i + "<br>");  
    i++;  
}
```

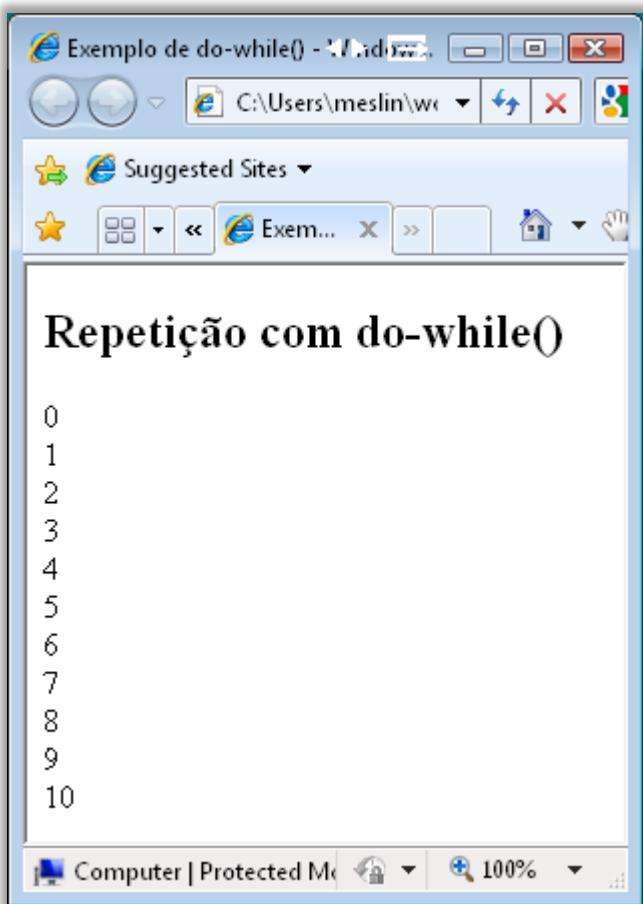
Laços do – while

- Repete um bloco de comandos enquanto uma condição for verdadeira.
- Teste da condição é realizado no final da iteração.
- Comandos são executados pelo menos uma vez
- Sintaxe:

```
do {  
    comandos;  
} while (condição);
```



Exemplo:



```
var i;  
i = 0;  
do {  
    document.write(i + "<br>");  
    i++;  
} while(i <= 10);
```

Comparação entre while() e do-while()

Usando while

```
var i;  
i = 8752;  
while ( i <= 10 ) {  
    document.write(i + "<br>");  
    i++;  
}
```

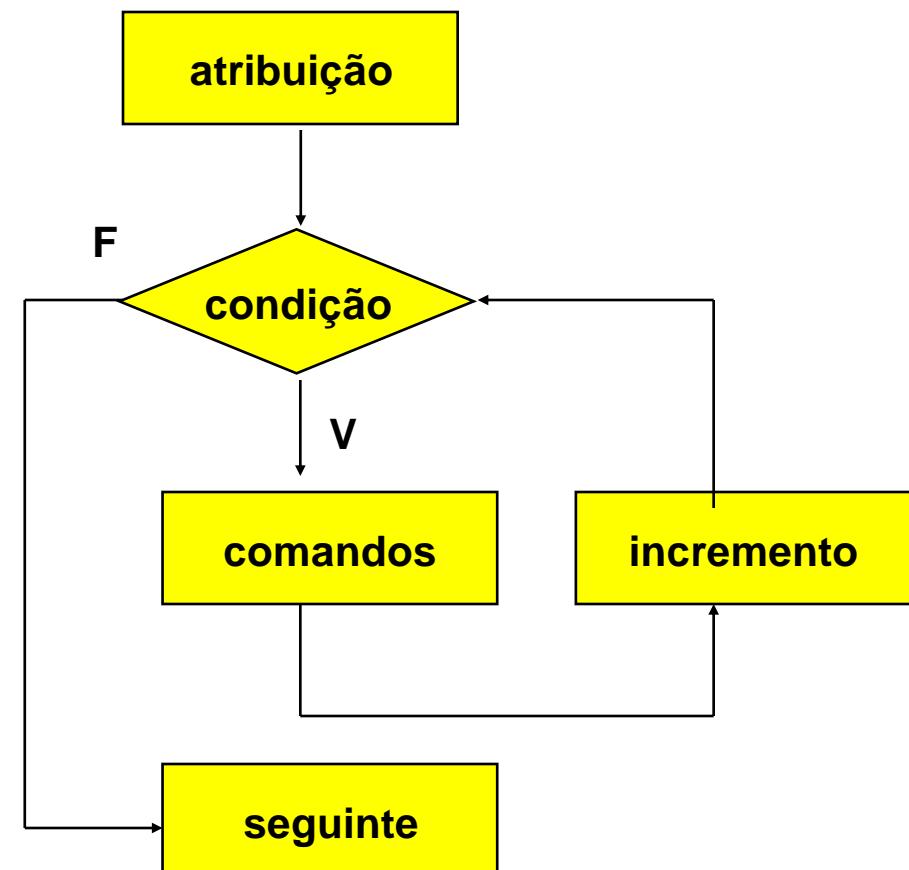
Usando do-while

```
var i;  
i = 8752;  
do {  
    document.write (i + "<br>");  
    i++;  
} while ( i <= 10 );
```

Laços for()

- Repete um comando ou bloco controlado por uma variável.
- Forma geral:

```
for (atribuição; condição; incremento) {  
    comandos;  
}  
seguinte;
```



Exemplo:



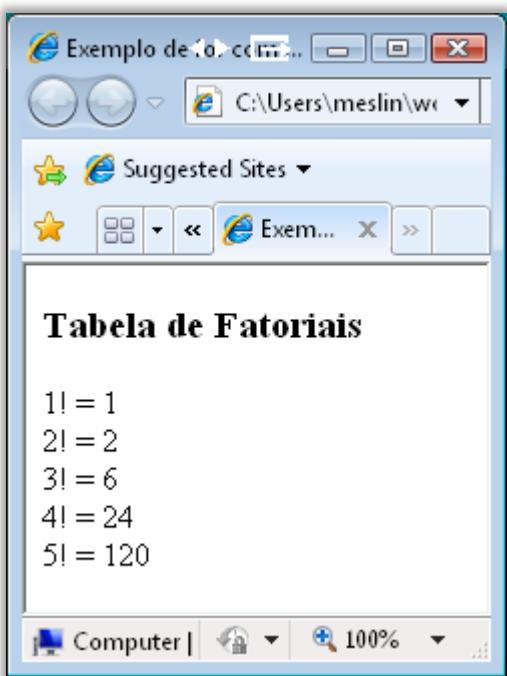
```
var i;  
for(i = 0; i < 5; i++)  
    document.write (i + "<br>");
```

Separador de comandos no for()

- A vírgula funciona como separador de comandos na atribuição e no incremento
- Qualquer uma das três partes do for é opcional (o ; não!)
- Um comando for sem condição é um loop eterno
- Sintaxe:

```
for (inic1, inic2, inic3, ...; condição; atual1, atual2, atual3, ...) {  
    comandos;  
}
```

Exemplo:

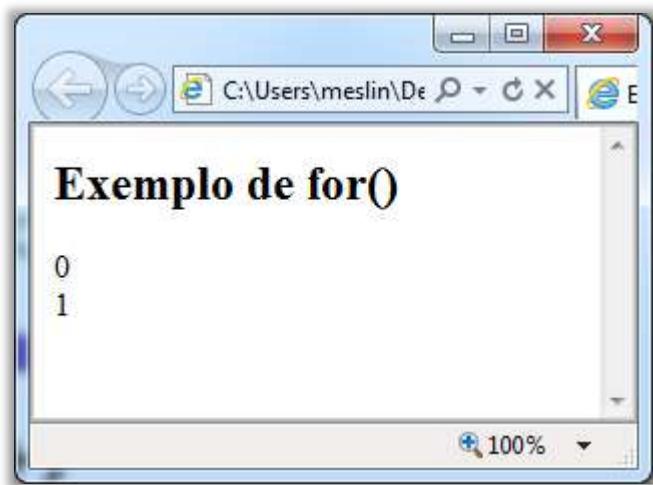


```
document.write ("<h3>Tabela de Fatoriais</h3>");  
for(var i = 1, fat = 1; i < 6; i++, fat *= i)  
    document.write(i, "!" , fat, "<br>");
```

Comandos break e continue

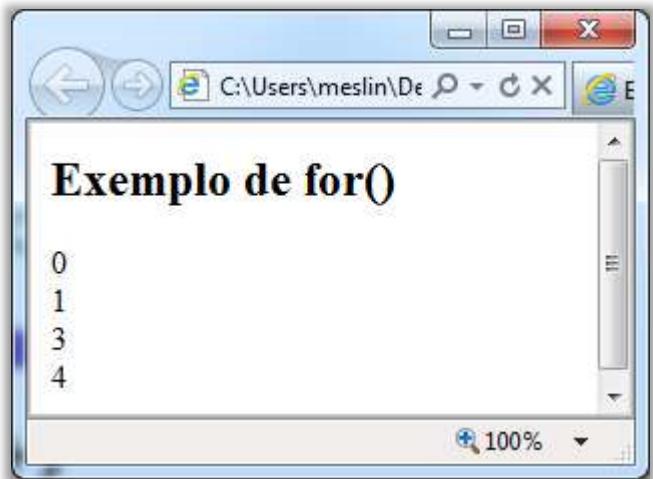
- Permitem um controle adicional sobre os laços de repetição
- break;
 - Pare a repetição já!
- continue;
 - Passe para a próxima iteração do laço!

Exemplo de uso de break:



```
for(i = 0; i < 5; i++) {  
    if(i == 2) break;  
    document.write (i + "<br>");  
}
```

Exemplo de uso de continue:



```
for(i = 0; i < 5; i++) {  
    if(i == 2) continue;  
    document.write (i + "<br>");  
}
```

Exemplo de for-in

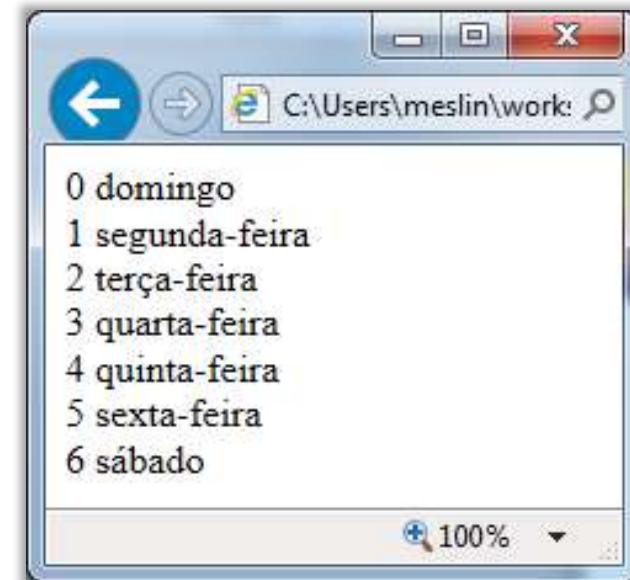
- Cria um índice para percorrer um array
- Sintaxe:

```
for (contador in objeto) {  
    comandos;  
}
```

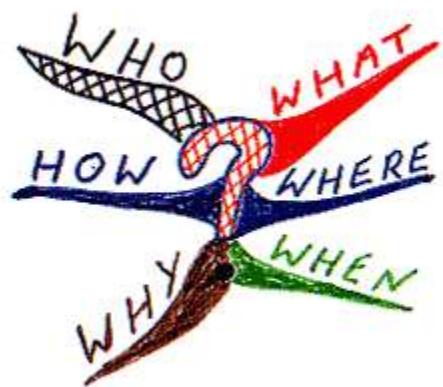
Exemplo de for-in

```
var diaSemana= new Array();
diaSemana[0] = "domingo";
diaSemana[1] = "segunda-feira";
diaSemana[2] = "terça-feira";
diaSemana[3] = "quarta-feira";
diaSemana[4] = "quinta-feira";
diaSemana[5] = "sexta-feira";
diaSemana[6] = "sábado";

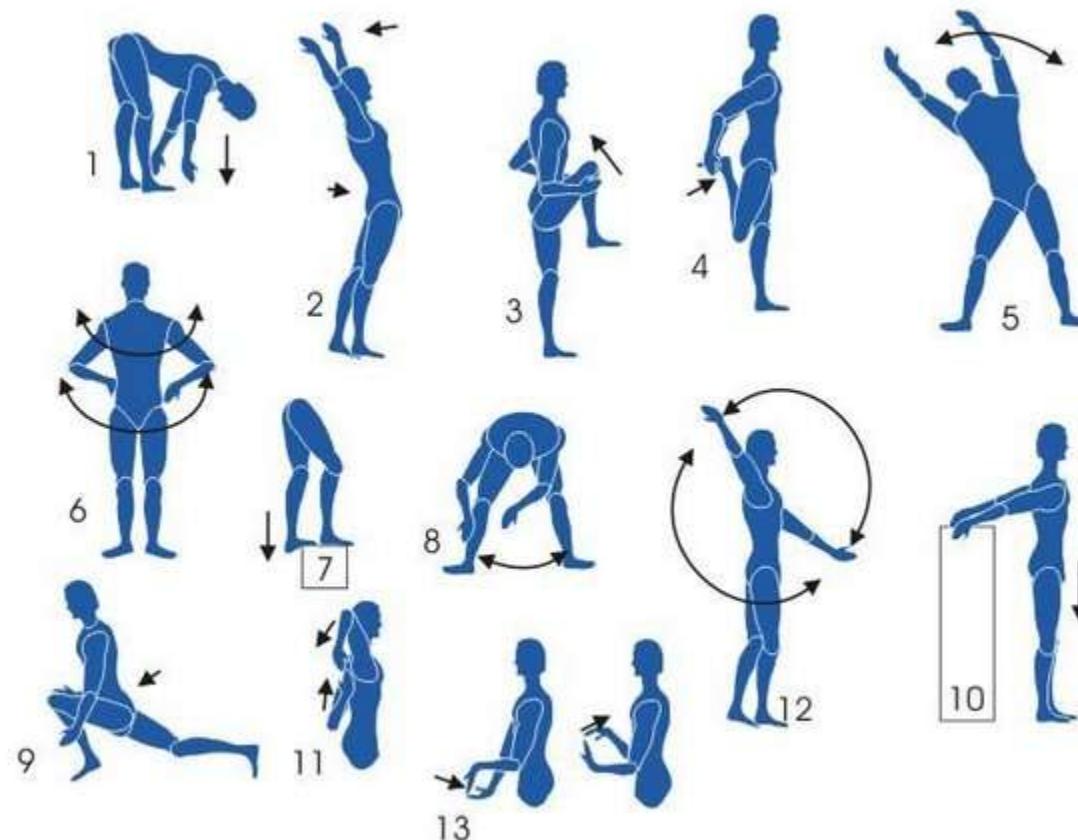
for (var i in diaSemana)
    document.write(i + " " + diaSemana[i] + "<br />");
```



Perguntas?



Exercícios



Exercícios

- Faça um programa que imprima todos os números de 0 (inclusive) até 10 (exclusive)

Exercícios

- Faça um programa que imprima todos os números pares de 0 até 25 ॥

Exercícios

- Faça um programa que imprima todos os números pares de um intervalo
 - Faça uma versão lendo os dados de um formulário HTML
 - Faça uma versão lendo os dados através de um prompt()

Exercícios

- Faça um programa que imprima a tabuada de um número lido
 - Faça uma versão lendo de um formulário HTML
 - Faça uma versão lendo de um prompt()

Exercício

- Faça uma página que mostre os anos das próximas 200 olimpíadas

Calculadora com Repetição

- Faça uma página HTML que receba do usuário 2 números reais através de um prompt() e mostre a sua soma.
- Após mostrar a soma, a página deverá perguntar se o usuário deseja somar novamente através do método confirm()

Exercícios

- Faça um programa que imprima o fatorial de um número lido
 - Faça uma versão lendo de um formulário HTML
 - Faça uma versão lendo de um prompt()

Desafios

- Faça um programa que imprima TODA a tabuada

Tabuada										
X	1	2	3	4	5	6	7	8	9	
1	1	2	3	4	5	6	7	8	9	
2	2	4	6	8	10	12	14	16	18	
3	3	6	9	12	15	18	21	24	27	
4	4	8	12	16	20	24	28	32	36	
5	5	10	15	20	25	30	35	40	45	
6	6	12	18	24	30	36	42	48	54	
7	7	14	21	28	35	42	49	56	63	
8	8	16	24	32	40	48	56	64	72	
9	9	18	27	36	45	54	63	72	81	

Desafios

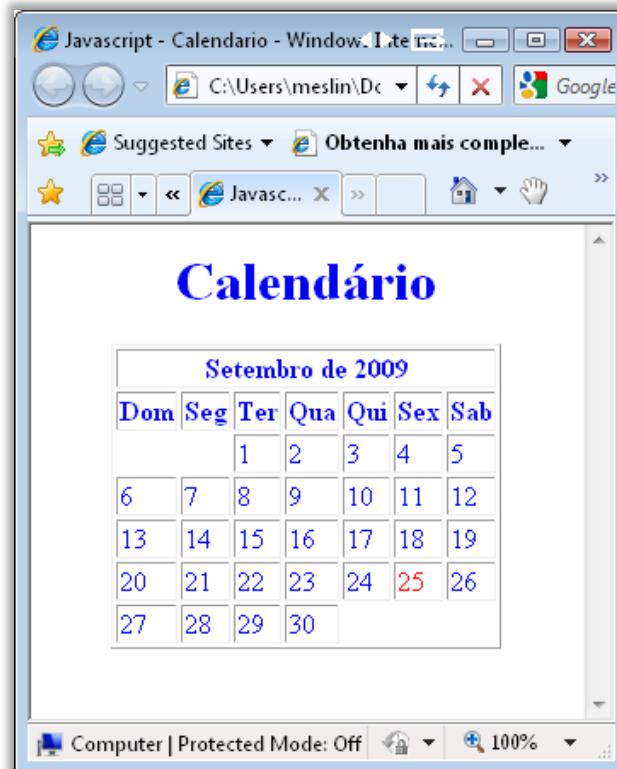
- Faça um programa que imprima um tabuleiro de xadrez

Pura matemática (mas feito em JavaScript)

- Considere a função $f(x) = 2x + 5$
- Calcule manualmente $f(x)$ para x variando de 1 até 5

Grande desafio

- Obtenha a data de hoje (não é para ler) e monte uma página com o calendário, destacando o dia de hoje (em vermelho, por exemplo).



Enorme Desafio: Jogo da Senha

- Faça o Jogo da Senha
- Funcionamento:
 - 2 jogadores
 - Jogador 1:
 - O jogador 1 informa um número entre 0 e 9, inclusive os extremos. A leitura é realizada através de um campo do tipo senha.
 - O programa não aceita números inválidos. Neste caso, pede para entrar com o número novamente
 - O jogador 1 informa o número máximo de tentativas do jogador 2
 - Jogador 2:
 - Chuta um número
 - Se for errado ou inválido, perde a chance e chuta novamente
 - Se o número for válido mas errado, é informado se o chute foi muito grande ou pequeno.
 - Se o chute estiver 1 unidade de distância da senha, o usuário é informado que o chute foi quente (o jogador 2 não é informado se o número é maior ou menor que a senha)

Outro desafio:

- Faça uma função que receba um número inteiro como parâmetro e retorne verdadeiro se o número for primo, caso contrário, retorne falso
- Faça um script que informe os N primeiros números primos.

TypeScript

TypeScript

- Navegadores não entendem TypeScript
- Precisamos compilar (transladar) para JavaScript
- Desenvolvida e mantida pela Microsoft
 - Site oficial: <https://www.typescriptlang.org>
 - Código fonte: <https://github.com/Microsoft/TypeScript>

TypeScript

- Superset do JavaScript
- Possibilita o uso de tipos estritos
- Características mais modernas
 - Operador seta
 - let
 - const
 - generic
 - interface
 - tupla

Instalação de TypeScript (Linux – Ubuntu)

- No terminal, digite o seguinte comando:

```
$ sudo apt install -y node-typescript
```

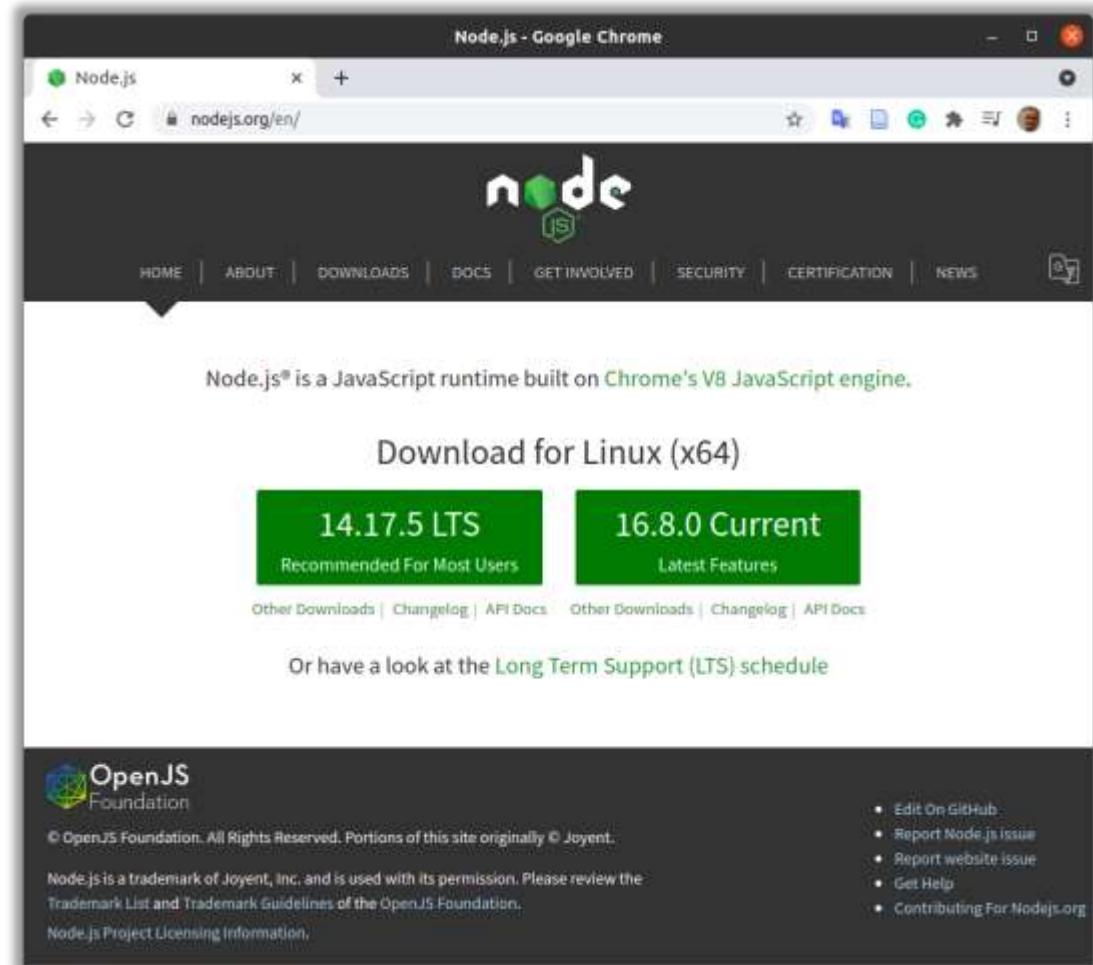
- Ou então:

```
$ sudo apt install npm
```

Instalação de TypeScript (Windows)

- Visite <https://nodejs.org/>
- Faça download da versão LTS (preferencialmente) e instale
- Em uma janela de linha de comando, digite o seguinte comando:

```
C:/> npm install -g typescript
```



Para compilar TypeScript

- No mesmo diretório onde os arquivos TypeScript e JavaScript estão:
\$ tsc <arquivo.ts> <arquivo.js>
- Ou melhor...
\$ tsc <arquivo.ts>
- Ou melhor ainda... (usar a opção w: watch)
\$ tsc <arquivo.ts> -w
- Ou melhor que o melhor ainda... Ver o próximo slide!

Opções de Organização e Compilação

- Para inicializar a configuração (no diretório typescript)

```
$ tsc --init
```

message TS6071: Successfully created a tsconfig.json file.

- Examinar no arquivo tsconfig.json as opções:

- target
 - rootDir
 - outDir

- Fora de compilerOptions, incluir lista de diretórios-fonte (com as aspas):
 "**include**": ["**src**", "**src/diretorio**"]

- Compilar com a opção -w (--watch):

```
$ tsc -w
```

Opções de Organização e Compilação

tsconfig.json

```
{
  "compilerOptions": [
    /* Basic Options */
    // "incremental": true, /* Enable incremental compilation */
    // "target": "es5", /* Specify ECMAScript target version: 'ES3' (default), 'ES5', 'ES2015', 'ES2016', 'ES2017', 'ES2018', 'ES2019', 'ES2020', or 'ESNEXT'. */
    // "module": "commonjs", /* Specify module code generation: 'none', 'commonjs', 'amd', 'system', 'umd', 'es2015', 'es2020', or 'ESNext'. */
    // "lib": [], /* Specify library files to be included in the compilation. */
    // "allowJs": true /* Allow javascript files to be compiled. */
    // "checkJs": true /* Report errors in .js files. */
    // "jsx": "preserve", /* Specify JSX code generation: 'preserve', 'react-native', or 'react'. */
    // "declaration": true, /* Generates corresponding '.d.ts' file. */
    // "declarationMap": true, /* Generates a sourcemap for each corresponding '.d.ts' file. */
    // "outDir": "./dist", /* Redirect output structure to the directory. */
    // "rootDir": "./src", /* Specify the root directory of input files. Use to control the output directory structure with --outDir. */
    // "composite": true, /* Enable project compilation */
    // "tsbuildInfoFile": "./", /* Specify file to store incremental compilation information */
    // "removeComments": true, /* Do not emit comments to output. */
    // "noEmit": true, /* Do not emit outputs. */
    // "importHelpers": true, /* Import emit helpers from 'tslib'. */
    // "downlevelIteration": true, /* Provide full support for iterables in 'for-of', spread, and destructuring when targeting 'ES5' or 'ES3'. */
    // "isolateModules": true, /* Transpile each file as a separate module (similar to 'ts.transpileModule'). */

    /* Strict Type-Checking Options */
    "strict": true, /* Enable all strict type-checking options. */
    // "noImplicitAny": true, /* Raise error on expressions and declarations with an implied 'any' type. */
    // "strictNullChecks": true, /* Enable strict null checks. */
    // "strictFunctionTypes": true, /* Enable strict checking of function types. */
    // "strictBindCallApply": true, /* Enable strict 'bind', 'call', and 'apply' methods on functions. */
    // "strictPropertyInitialization": true, /* Enable strict checking of property initialization in classes. */
    // "noImplicitThis": true, /* Raise error on 'this' expressions with an implied 'any' type. */
    // "alwaysStrict": true, /* Parse in strict mode and emit "use strict" for each source file. */

    /* Additional Checks */
    // "noUnusedLocals": true, /* Report errors on unused locals. */
    // "noUnusedParameters": true, /* Report errors on unused parameters. */
    // "noImplicitReturns": true, /* Report error when not all code paths in function return a value. */
    // "noFallthroughCasesInSwitch": true, /* Report errors for fallthrough cases in switch statement. */

    /* Module Resolution Options */
    // "moduleResolution": "node", /* Specify module resolution strategy: 'node' (Node.js) or 'classic' (TypeScript pre-1.6). */
    // "baseUrl": "./", /* Base directory to resolve non-absolute module names. */
    // "paths": {}, /* A series of entries which re-map imports to lookup locations relative to the 'baseUrl'. */
    // "rootDirs": [], /* List of root folders whose combined content represents the structure of the project at runtime. */
    // "modules": "commonjs", /* The module code generation style for imports. */
    // "types": [], /* Type declaration file to be included in compilation. */
    // "allowSyntheticDefaultImports": true, /* Allow default imports from modules with no default export. This does not affect code emit, just typechecking. */
    // "esModuleInterop": true, /* Enables emit interoperability between CommonJS and ES Modules via creation of namespace objects for all imports. Implies 'allowSyntheticDefaultImports'. */
    // "preserveSymlinks": true, /* Do not resolve the real path of symlinks. */
    // "allowUndeclaredGlobalAccess": true, /* Allow accessing UMD globals from modules. */

    /* Source Map Options */
    // "sourceRoot": "", /* Specify the location where debugger should locate TypeScript files instead of source locations. */
    // "mapRoot": "", /* Specify the location where debugger should locate map files instead of generated locations. */
    // "inlineSourceMap": true, /* Emit a single file with source maps instead of having a separate file. */
    // "inlineSources": true, /* Emit the source alongside the sourcemaps within a single file; requires '--inlineSourceMap' or '--sourceMap' to be set. */

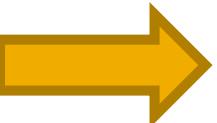
    /* Experimental Options */
    // "experimentalDecorators": true, /* Enables experimental support for ES7 decorators. */
    // "emitDecoratorMetadata": true, /* Enable experimental support for emitting type metadata for decorators. */

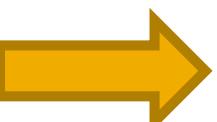
    /* Advanced Options */
    // "forceConsistentCasingInFileNames": true /* Disallow inconsistently-cased references to the same file. */
  ],
  "include": ["src", "src/drl1", "src/exercicios"]
}
```

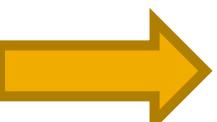
Opções de Organização e Compilação

- Algumas configurações interessantes fora do padrão (default):

`/* Additional Checks */`

 `"noUnusedLocals": true, /* Report errors on unused locals. */`

 `"noUnusedParameters": true, /* Report errors on unused parameters. */`

 `"noImplicitReturns": true, /* Report error when not all code paths in function return a value. */`

`"noFallthroughCasesInSwitch": true, /* Report errors for fallthrough cases in switch statement. */`

`"target": "es2018", /* Set the JavaScript language version for emitted JavaScript and include compatible library declarations. */`

Servidor Web

- Python (porta default: 8000):

```
$ python -m http.server 8080
```

- Usando Node.js (porta default: 8080):

- Instalar o servidor Web usando NPM

```
$ npm install -g http-server
```

```
$ http-server -p 8080
```

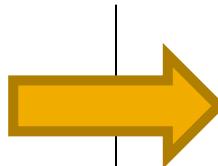
Exemplo

TypeScript

```
onload = function() {
    var realizaSoma =
        document.getElementById("realizaSoma") as
        HTMLButtonElement;
    var numero1 = document.getElementById("numero1")! as
        HTMLInputElement
    var numero2 = document.getElementById("numero2")! as
        HTMLInputElement;

    realizaSoma?.addEventListener("click", function(){
        console.log(soma(parseFloat(numero1.value),
        parseFloat(numero2.value)));
    });
}

function soma(numero1: number, numero2: number) {
    return numero1 + numero2;
}
```



JavaScript

```
"use strict";
onload = function () {
    var realizaSoma =
        document.getElementById("realizaSoma");
    var numero1 = this.document.getElementById("numero1");
    var numero2 = this.document.getElementById("numero2");
    realizaSoma === null || realizaSoma === void 0 ?
        void 0 :
        realizaSoma.addEventListener("click", function () {
            console.log(soma(parseFloat(numero1.value),
            parseFloat(numero2.value)));
        });
    function soma(numero1, numero2) {
        return numero1 + numero2;
    }
}
```

TypeScript
agora vai começar mesmo!

Tipos de Dados

- TypeScript usa estritos tipos de variáveis
 - Uma variável é SEMPRE do tipo que ela foi criada
 - Verificado em tempo de compilação
- Tipos básicos:

Template: Permite
interpolação usando \${}

Tipo	Descrição	Exemplo
string	Um texto entre aspas	'texto', "texto", `texto`
number	Qualquer número inteiro ou real	8, 7.5, 2
boolean	Valor lógico	true, false
any	Um valor de qualquer tipo	true, false, 'texto', "texto", 8, 7.5, 2
void	Nada (geralmente tipo de retorno de função)	
undefined	Valor não definido	Undefined
null	Valor nulo	Null
unknown	Tipo desconhecido	true, false, 'texto', "texto", 8, 7.5, 2
never	Tipo de função que nunca retorna	
bignint	Inteiros maiores do que (2**53)-1	8752n

Tipos de Dados

- TypeScript usa estritos tipos de variáveis
 - Uma variável é SEMPRE do tipo que ela foi criada
 - Verificado em tempo de compilação
- Tipos não básicos:

Tipo	Descrição	Exemplo
object	Um objeto	{nome='Sílvia', idade=40}
Function	Uma função	
[] (array) Array<tipo>	Vetor de qualquer tipo, com diversos tipos	[8,7,5], [1, "texto", true]
[tipos] (tupla)	Lista de tamanho fixo de valores	[2, 'admin']
enum	Constantes numeradas automaticamente	{VERDE, VERMELHO, AMARELO}
union		string number
intersection		
class	Criar classes	
interface	Criar interfaces (contrato entre objetos)	

Declaração de Variáveis

- Todas as variáveis precisam ser **explicitamente** declaradas
 - `var` → variável existe dentro da função
 - `let` → variável existe dentro do bloco
 - `const` → cria uma constante dentro do bloco
- Uso da diretiva `let`
 - Sintaxe:

```
let <variável>: <tipo>;
```

```
let <variável>: <tipo | tipo | tipo>;
```

Declaração de Variáveis

Anotação

```
let nome: string;
let idade: number;
let matriculado: boolean;
let uuid: string|number;
let escreve: Function;
```

Inferência

```
let nome = "Claudia";
let idade = 57;
let matriculado = false;
let uuid = 8752;
let escreve = () => console.log("Uma função");
```

Tipos de Dados – Escalares e Constantes

TiposDeVariaveis.ts

```
// Constantes
const TEXTO: string = "Um texto";
const OUTRO_TEXTO = "Um outro texto";
//TEXTO = "Um novo texto"; <== ERRADO!

// Escalares
let nome: string;
let idade: number;
let matriculado: boolean;
let uuid: string | number;
let escreve: Function;
uuid = "8752";
uuid = 8752;
//uuid = false; <== ERRADO!
```

TEXTO é constante,
não pode mudar

uuid é string ou
número, não pode
ser booleano

Tipos de Dados – Vetores

TiposDeVariaveis.ts

```
// Vetor de um tipo único
let frutas = ['banana', 'laranja', 'abacaxi'];
frutas.push('pera');
frutas[2] = 'kiwi';
//frutas.push(8752); <== ERRADO!
//frutas[1] = 8752; <== ERRADO!

// Vetor de vários Tipos
let varios = ['banana', 8752]
varios.push(2578)
varios.push('laranja');
//varios.push(true); <== ERRADO!
varios[1] = 'uva';
varios[0] = 5;
//varios[2] = false; <== ERRADO!

let vetor = [];
vetor.push(1);
vetor.push("banana");
```

vários é composto por strings ou números, não inclui booleanos

Tipos de Dados – Explicitando Vetores

TiposDeVariaveis.ts

```
// Vetores
let nomes: string[] = [];           // vetor de strings
let mistura: (string|number|boolean)[] = [] // vetor de strings ou números ou booleanos
let numeros: Array<number>;          // vetor de números usando generics
```

Tipos de Dados – Objetos

TiposDeVariaveis.ts

```
let aluno = {  
    nome: "Ana",  
    idade: 20,  
    matriculado: true,  
};  
  
aluno.nome = 'Katia';  
//aluno.nome = 30;  <== ERRADO!  
//aluno.disciplinas = [];  <== ERRADO!  
  
aluno = {  
    nome: "Márcia",  
    matriculado: false,  
    idade: 40,  
    // disciplinas = [],  <== ERRADO!  
};
```

aluno não tem o
campo disciplinas

Tipos de Dados – Explicitando Objetos

TiposDeVariaveis.ts

```
let visitante: object; // visitante é um objeto qualquer
visitante = {           // logo, visitante pode ser uma estrutura
  nome: 'Renata',
  idade: 35,
};
visitante = [];         // ou um vetor
```

Tipos de Dados – Explicitando Objetos

TiposDeVariaveis.ts

```
let professor: {  
    nome: string,  
    idade: number,  
    ministrando: boolean,  
};  
  
professor = {  
    nome: 'Carolina',  
    idade: 50,  
    ministrando: true,  
    // disciplinas: [], <== ERRADO!  
};
```

O objeto professor
não tem o campo
disciplinas

Tipos de Dados – Explicitando Objetos

Objetos/objeto.ts

```
let pessoa1: {           // variável com campos mas sem valores
  nome: string;        // note o uso de ; ao especificar tipo
  idade: number;
}

let estudante1 = {       // variável com campos com valores
  nome: 'Sandra',      // note o uso de , ao especificar valor
  idade: 15,
}

const estudante2 = {     // constante com campos e valores
  nome: 'Paula',
  idade: 18
}
```

- : para tipo
- = para valores

Tipos de Dados – Explicitando Objetos

Objetos/objeto.ts

```
let estudante3: { // não precisa ser tão detalhista
    nome: string;
    idade: number;
} = {
    nome: 'Fernanda',
    idade: 12,
}
```

Os tipos podem ser
inferidos

```
const pessoa2: { // não precisa ser tão detalhista
    nome: string;
    idade: number;
} = {
    nome: 'Fernanda',
    idade: 12,
}
```

Tipos de Dados – Tuplas

TiposDeVariaveis.ts

```
/*
 * Tupla
 */
{
  let papel = [2, 'admin'];
  papel[1] = 1;
}
{
  let papel = [2, 'admin'] as [number, string];
  //papel[1] = 1; ==> ERRO!
}
```

Tipos de Dados – Qualquer Tipo (cuidado!)

TiposDeVariaveis.ts

```
// Tipo any (CUIDADO)
let valor: any = 25;
valor = 'qualquer';
valor = true;
valor = { nome: "Regina", idade: 50, }
```



Tipos de Dados – Enumeração

- Cria constantes começando de 0 (zero)
- Valor default pode ser modificado
 - ADMIN = 8
- Se for inicializada com valores não numéricos, todos os valores devem ser inicializados

TiposDeVariaveis.ts

```
/*
 * Enumeração
 */
enum Papel {ADMIN, USER, ROOT, VISITOR, STUDENT};
let estado: Papel;
estado = Papel.USER;
if(estado === Papel.USER) {
  console.log("Papel de usuário");
}
```

Aliases

Aliases.ts

```
type NumberOrString = number | string;
type Timestamp = {nDay: number, nMonth: number, nYear: number}
type Bus = {line: NumberOrString, uuid: string, serial: string, timestamp: Timestamp};

const _getWeekDay = (timestamp: {nDay: number, nMonth: number, nYear: number}): void => {
    console.log(` ${timestamp.nDay}/${timestamp.nMonth}/${timestamp.nYear}`)
}

const getWeekDay = (timestamp: Timestamp): void => {
    console.log(` ${timestamp.nDay}/${timestamp.nMonth}/${timestamp.nYear}`)
}

const _getBus = (bus: {line: number | string, uuid: string, serial: string, timestamp: {nDay: number,
    console.log(`${bus.line} @ ${bus.uuid}`)

const getBus = (bus: Bus): void => {
    console.log(`${bus.line} @ ${bus.uuid}`)
```

Funções em TypeScript

Declaração de Função

Uma função pode ser declarada como fazemos em Javascript, apenas acrescentando os tipos:

```
function nomeDaFuncao(parametro1: Tipo, parametro2: Tipo): TipoDeRetorno {  
    // Corpo da função  
    // Pode conter lógica para processar os parâmetros e retornar um valor  
}
```

Exemplo:

```
function somar(a: number, b: number): number {  
    return a + b;  
}  
  
const resultado = somar(3, 5); // Chama a função e atribui o resultado a 'resultado'  
console.log(resultado); // Saída: 8
```

Declaração de Função – Arrow Function

A função também pode ser declarada usando o operador seta (arrow) =>

```
const nomeDaFuncao = (parametro1: Tipo, parametro2: Tipo): TipoDeRetorno => {
    // Corpo da função
    // Pode conter lógica para processar os parâmetros e retornar um valor
}
```

Exemplo:

```
const somar = (a: number, b: number): number => {
    return a + b;
}

const resultado = somar(3, 5); // Chama a função e atribui o resultado a 'resultado'
console.log(resultado);    // Saída: 8
```

Tipos de Dados – Funções

TiposDeVariaveis.ts

```
const circunferencia = (raio: number) => {
    return 2 * Math.PI * raio;
}

console.log(circunferencia(5.7));
//console.log(circunferencia(true)); <== ERRADO!
```

O parâmetro somente
pode ser numérico

Tipos de Dados – Funções

- Parâmetros a e b são obrigatórios
- Parâmetro c é opcional, sem valor default
 - ? Indica que é opcional
- Parâmetro d é opcional, com valor default
 - Atribuição indica que é default
 - Não usar junto com ?
- Parâmetros obrigatórios devem aparecer antes dos parâmetros opcionais

TiposDeVariaveis.ts

```
const funcao = (a: number, b: number,  
               c?: number | string,  
               d: number = 32): void => {  
    console.log(a+b+d);  
    console.log(c)  
}  
  
//funcao(1); <== ERRADO!  
funcao(1,2);  
funcao(1,2,4);  
funcao(1,2,'texto');  
funcao(1,2,4,16);  
//funcao(1,2,4,'texto'); <== ERRADO!
```

Tipos de Dados – Funções

TiposDeVariaveis.ts

```
// função do tipo void
const funcao = (a: number, b: number, c?: number | string, d: number = 32): void => {
    console.log(a+b+d);
    console.log(c)
}

// função do tipo numérico
const soma = (a: number, b: number): number => {
    return a+b;
}
```

Tipos de Dados – Funções (void x undefined)

Functions.ts

```
function funcao1(numero: number) : void {  
    console.log("O valor é " + numero);  
    return; // opcional  
}
```

Functions.ts

```
function funcao2(numero: number) : undefined {  
    console.log("O valor é " + numero);  
    return; // obrigatório  
}
```

Assinatura de função

Functions.ts

```
// Assinatura da função calculadora
let calculadora: (n1: number, n2: number, op: string) => number;

calculadora = (num1: number, num2: number, operacao: string): number => {
    switch (operacao) {
        case '+':
            return num1 + num2;
        case '-':
            return num1 - num2;
        case '*':
            return num1 * num2;
        case '/':
            return num1 / num2;
        default:
            return NaN;
    }
}
```

Assinatura de função

Functions.ts

```
let matriculaAluno: (aluno: {nome: string, idade: number}) => boolean;
type Aluno = {nome:string, idade: number};

matriculaAluno = (aluno: Aluno): boolean => {
    return true;
}
//matriculaAluno = (aluno: {nome:string, idade: number}): boolean => {
//    return true;
//}
    ➔ Correto!

// O tipo Aluno é do mesmo tipo do parâmetro da função e é aceito em TypeScript
let aluno: Aluno;
aluno = {nome: 'Sandra', idade: 10};
matriculaAluno(aluno);
```

Assinatura de função

Functions.ts

```
// Declara variável que recebe qualquer função que receba 2 números
let variavelFuncao1: (n1: number, n2: number) => number;

// Declara constante que tem que ser inicializada com uma função
const variavelFuncao2:Function = (n1: number, n2: number): number => {
    return n1 + n2;
};
```

TypeScript – Classes Predefinidas

Introdução

- TypeScript (Javascript) é uma linguagem de programação orientada a objetos (OO)
- Uma linguagem de programação OO permite a definição de novos objetos e novos tipos de dados

Programação OO

- Podemos utilizar os objetos já definidos na linguagem ou criar novos objetos
- Começaremos utilizando objetos pré-definidos na linguagem.
- Um objeto é apenas um tipo especial de dado. Um objeto possui propriedades e métodos.
- Um objeto é uma coisa, qualquer coisa, como uma coisa no mundo real
 - Ex.: carros, pássaros, dinheiro, livros, etc.
- No navegador, objetos são o próprio navegador, formulários, botões, caixas de texto, etc.

Propriedades

- São os valores associados a um objeto
- No exemplo abaixo usamos a propriedade `length` do objeto `string` para retornar a quantidade de caracteres de um texto
- Exemplo:

```
var txt;  
txt = "Olá Turma!";  
console.log(txt.length);
```

- O código acima terá como saída o valor 9

Métodos

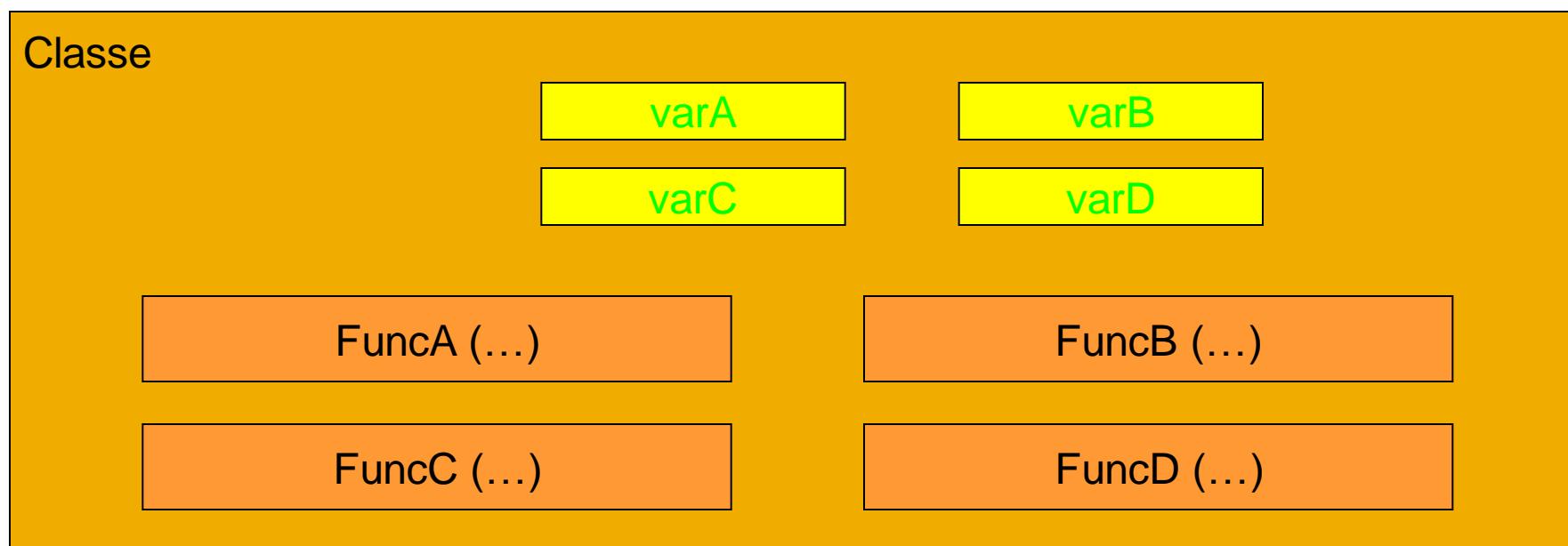
- São as ações que podem ser executadas por um objeto
- No exemplo abaixo estamos usando o método `toUpperCase()` do objeto `string` para exibir o texto em letras maiúsculas
- Exemplo:

```
var txt;  
txt = "Oi Turma!";  
console.log(txt.toUpperCase());
```

- O código acima terá como saída o texto Oi TURMA!

Objeto

- Em resumo, um objeto é um conjunto de:
 - Propriedades
 - Métodos
- Objetos representam "coisas" no programa, como documentos ou elementos HTML



Criando uma Instância de um Objeto

- Você pode usar o operador ***new*** para criar instâncias de objetos de uma classe em particular

```
variavel = new tipoDoObjeto(parametros);
```

- **tipoDoObjeto()** é chamado construtor

- Ex.: Date é um objeto pré-definido

- Criando um objeto

```
var objA = new Date();
```

- Criando um objeto com uma data pré-definida

```
var objB = new Date(2013, 8, 17);
```

Objeto Math

- O objeto Math contém:

Método	Finalidade
Math.abs(x)	Valor absoluto
Math.floor(x)	Inteiro
Math.log(x)	Logaritmo
Math.max(lista de valores)	Máximo
Math.pow(x, y)	Eleva a determinada potência
Math.random()	Número aleatório
Math.round(x)	Inteiro mais próximo
Math.sin(x)	Seno
Math.sqrt(x)	Raiz quadra

Objeto String

- Permite a manipulação de strings (textos) ou de partes de um texto
- Esta manipulação pode ser feita através da chamada de métodos
- Existem vários métodos pré-definidos para o objeto String

Objeto String: método indexOf()

- Exemplo usando o método indexOf() para retornar a posição da primeira ocorrência de um texto específico em um texto

```
var str: string;  
str = "Olá Turma!";  
console.log(str.indexOf("Olá") + "<br />");  
console.log(str.indexOf("Turma") + "<br />");  
console.log(str.indexOf("turma"));
```

- O código acima terá como saída os valores:

0

3

-1

Objeto String: método charAt()

- O método `charAt(posição)` retorna o caracter na posição informada.

```
var texto: string, posicao: number, caracter: string;  
texto = "Este é um texto";  
posicao = 8;  
  
caracter = texto.charAt(posicao);  
console.log("O caracter que está na posição " + posicao + " é " + caracter);  
  
console.log("Ou seja, " + texto[posicao]);
```

```
O caracter que está na posição 8 é m  
Ou seja, m
```

Objeto String: método match()

- Exemplo usando o método match() para procurar um texto específico dentro de um texto e caso encontre retorna o valor do texto:

```
var str: string;  
str="Oi Turma!";  
console.log(str.match("turma"));  
console.log(str.match("Turma"));  
console.log(str.match("turna"));  
console.log(str.match("Turma!"));
```

- O exemplo acima terá como saída:

```
null  
Turma  
null  
Turma!
```

Objeto String: método replace()

- Exemplo usando o método replace() para substituir alguns caracteres por outros em um texto:

```
var str: string;  
str="Oi Turma!";  
console.log(str.replace(/Turma/ , "Pessoal"));  
console.log(str);
```

- O resultado da página será:

```
Oi Pessoal  
Oi Turma!
```

Objeto String - Propriedades

Property	Description
<u>constructor</u>	Returns the function that created the String object's prototype
<u>length</u>	Returns the length of a string
<u>prototype</u>	Allows you to add properties and methods to an object

Objeto String - Métodos

Method	Description
charAt()	Returns the character at the specified index
charCodeAt()	Returns the Unicode of the character at the specified index
concat()	Joins two or more strings, and returns a copy of the joined strings
fromCharCode()	Converts Unicode values to characters
indexOf()	Returns the position of the first found occurrence of a specified value in a string
lastIndexOf()	Returns the position of the last found occurrence of a specified value in a string
match()	Searches for a match between a regular expression and a string, and returns the matches
replace()	Searches for a match between a substring (or regular expression) and a string, and replaces the matched substring with a new substring
search()	Searches for a match between a regular expression and a string, and returns the position of the match
slice()	Extracts a part of a string and returns a new string
split()	Splits a string into an array of substrings
substr()	Extracts the characters from a string, beginning at a specified start position, and through the specified number of character
substring()	Extracts the characters from a string, between two specified indices
toLowerCase()	Converts a string to lowercase letters
toUpperCase()	Converts a string to uppercase letters
valueOf()	Returns the primitive value of a String object

Objeto Array

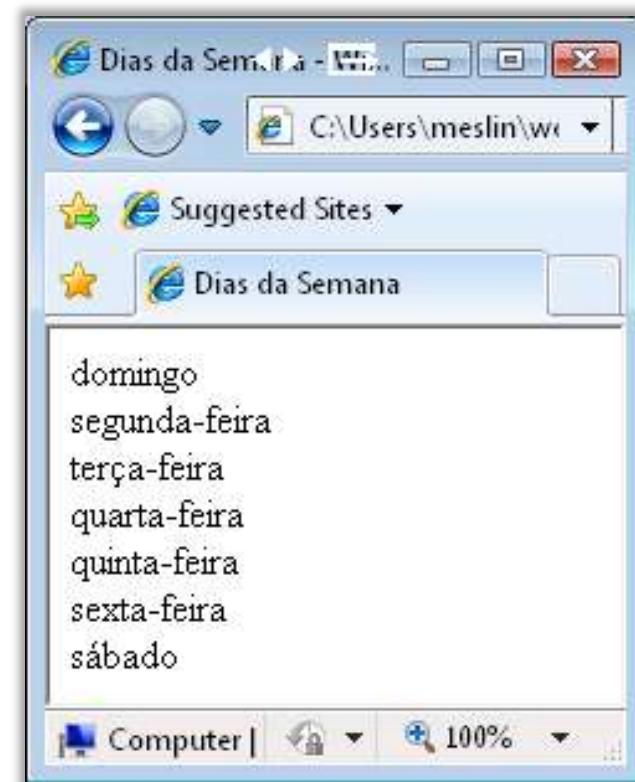
- Um objeto Array é utilizado para armazenar vários valores em uma única variável
- Todos os valores armazenados em um array são de um mesmo tipo
- Necessita de um índice para podermos nos referenciarmos a um dos dados armazenados no array
- O índice do array é um número inteiro. O primeiro índice de um array é sempre o 0 (zero).

Objeto Array: criação

- Exemplo criando um objeto Array

```
var diaSemana: string[] = new Array();
diaSemana[0] = "domingo";
diaSemana[1] = "segunda-feira";
diaSemana[2] = "terça-feira";
diaSemana[3] = "quarta-feira";
diaSemana[4] = "quinta-feira";
diaSemana[5] = "sexta-feira";
diaSemana[6] = "sábado";

for (var i=0; i<diaSemana.length; i++)
    document.write(diaSemana[i] + '<br>');
```



Objeto Array: método sort()

- Exemplo utilizando o método sort() que ordena o array em ordem crescente

```
var estado: string[] = new Array(5);
estado[0] = "Rio de Janeiro";
estado[1] = "Minas Gerais";
estado[2] = "Paraná";
estado[3] = "Bahia";
estado[4] = "São Paulo";

console.log(estado);
console.log(estado.sort());
```

```
▶ (5) ['Rio de Janeiro', 'Minas Gerais', 'Paraná', 'Bahia', 'São Paulo']
▶ (5) ['Bahia', 'Minas Gerais', 'Paraná', 'Rio de Janeiro', 'São Paulo']
```

Outro exemplo de criação de Array

```
var estado: string[];  
estado = ["Rio de Janeiro", "Minas Gerais", "Paraná", "Bahia", "São Paulo"];  
  
document.write(estado + "<br />");  
document.write(estado.sort() + "<br/>");  
document.write(estado.reverse() + "<br />");
```

Referência completa do objeto array

Method	Description
<code>concat()</code>	Joins two or more arrays and returns the result
<code>join()</code>	Puts all the elements of an array into a string. The elements are separated by a specified delimiter
<code>pop()</code>	Removes and returns the last element of an array
<code>push()</code>	Adds one or more elements to the end of an array and returns the new length
<code>reverse()</code>	Reverses the order of the elements in an array
<code>shift()</code>	Removes and returns the first element of an array
<code>slice()</code>	Returns selected elements from an existing array
<code>sort()</code>	Sorts the elements of an array
<code>splice()</code>	Removes and adds new elements to an array
<code>toSource()</code>	Represents the source code of an object
<code>toString()</code>	Converts an array to a string and returns the result
<code>unshift()</code>	Adds one or more elements to the beginning of an array and returns the new length
<code>valueOf()</code>	Returns the primitive value of an Array object

Objeto Date

- Permite a utilização de datas e horas
- Criando um objeto data:

```
var minhaData;  
minhaData = new Date();
```

- Obs.: o objeto data possuirá automaticamente a data e a hora atual como valor inicial

Objeto Date: método Date()

- Exemplo usando o método Date() para obtenção da data e hora corrente

```
document.write(Date());
```



Objeto Date: método Date()

- Exemplo usando o método Date() e um array para exibir o dia da semana

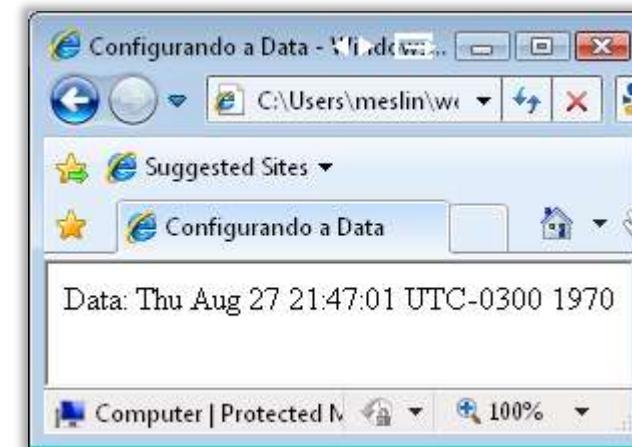
```
var data: Date, diaSemana: string[];
data = new Date();
diaSemana = new Array(7);
diaSemana[0] = "domingo";
diaSemana[1] = "segunda-feira";
diaSemana[2] = "terça-feira";
diaSemana[3] = "quarta-feira";
diaSemana[4] = "quinta-feira";
diaSemana[5] = "sexta-feira";
diaSemana[6] = "sábado";
document.write("Hoje é " + diaSemana[data.getDay()]);
```



Configurando datas

- Criar um objeto data representado o dia 27 de agosto de 1970

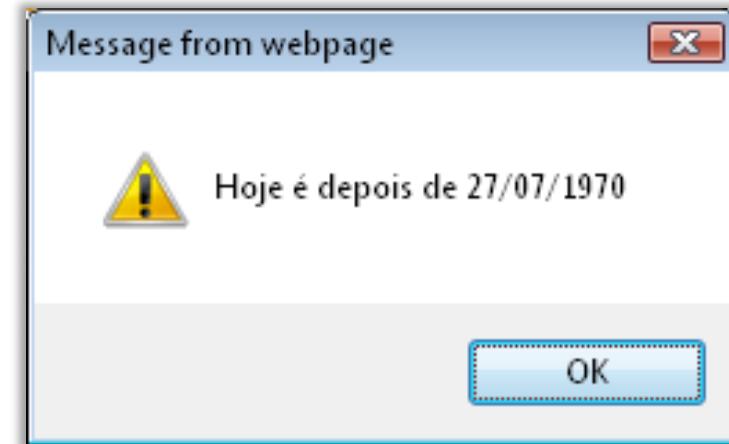
```
var data: Date;  
data = new Date();  
// 27 de agosto de 1970  
data.setFullYear(1970, 7, 27);  
document.write("Data: " + data);
```



Comparando Duas Datas

```
var minhaData = new Date();
minhaData.setFullYear(1970, 6, 27);
var hoje = new Date();

if (hoje > minhaData)
    alert ("Hoje é depois de 27/07/1970");
else
    alert ("Hoje é antes de 27/07/1970");
```



Referência do Objeto Date

Method	Description
Date()	Returns today's date and time
getDate()	Returns the day of the month from a Date object (from 1-31)
getDay()	Returns the day of the week from a Date object (from 0-6)
getFullYear()	Returns the year, as a four-digit number, from a Date object
getHours()	Returns the hour of a Date object (from 0-23)
getMilliseconds()	Returns the milliseconds of a Date object (from 0-999)
getMinutes()	Returns the minutes of a Date object (from 0-59)
getMonth()	Returns the month from a Date object (from 0-11)
getSeconds()	Returns the seconds of a Date object (from 0-59)
getTime()	Returns the number of milliseconds since midnight Jan 1, 1970
getTimezoneOffset()	Returns the difference in minutes between local time and Greenwich Mean Time (GMT)
getUTCDate()	Returns the day of the month from a Date object according to universal time (from 1-31)
getUTCDay()	Returns the day of the week from a Date object according to universal time (from 0-6)

Referência do Objeto Date

Method	Description
getUTCMonth()	Returns the month from a Date object according to universal time (from 0-11)
getUTCFullYear()	Returns the four-digit year from a Date object according to universal time
getUTCHours()	Returns the hour of a Date object according to universal time (from 0-23)
getUTCMMinutes()	Returns the minutes of a Date object according to universal time (from 0-59)
getUTCSeconds()	Returns the seconds of a Date object according to universal time (from 0-59)
getUTCMilliseconds()	Returns the milliseconds of a Date object according to universal time (from 0-999)
getYear()	Returns the year, as a two-digit or a three/four-digit number, depending on the browser. Use <code>getFullYear()</code> instead !!
parse()	Takes a date string and returns the number of milliseconds since midnight of January 1, 1970
 setDate()	Sets the day of the month in a Date object (from 1-31)
 setFullYear()	Sets the year in a Date object (four digits)
 setHours()	Sets the hour in a Date object (from 0-23)
 setMilliseconds()	Sets the milliseconds in a Date object (from 0-999)
 setMinutes()	Set the minutes in a Date object (from 0-59)

Referência do Objeto Date

Method	Description
<u>setMonth()</u>	Sets the month in a Date object (from 0-11)
<u>setSeconds()</u>	Sets the seconds in a Date object (from 0-59)
<u>setTime()</u>	Calculates a date and time by adding or subtracting a specified number of milliseconds to/from midnight January 1, 1970
<u>setUTCDate()</u>	Sets the day of the month in a Date object according to universal time (from 1-31)
<u>setUTCMonth()</u>	Sets the month in a Date object according to universal time (from 0-11)
<u>setUTCFullYear()</u>	Sets the year in a Date object according to universal time (four digits)
<u>setUTCHours()</u>	Sets the hour in a Date object according to universal time (from 0-23)
<u>setUTCMinutes()</u>	Set the minutes in a Date object according to universal time (from 0-59)
<u>setUTCSeconds()</u>	Set the seconds in a Date object according to universal time (from 0-59)
<u>setUTCMilliseconds()</u>	Sets the milliseconds in a Date object according to universal time (from 0-999)
<u>setYear()</u>	Sets the year in the Date object (two or four digits). Use setFullYear() instead !!

Referência do Objeto Date

Method	Description
toDateString()	Returns the date portion of a Date object in readable form
toGMTString()	Converts a Date object, according to Greenwich time, to a string. Use toUTCString() instead !!
toLocaleDateString()	Converts a Date object, according to local time, to a string and returns the date portion
toLocaleTimeString()	Converts a Date object, according to local time, to a string and returns the time portion
toLocaleString()	Converts a Date object, according to local time, to a string
toSource()	Represents the source code of an object
toString()	Converts a Date object to a string
toTimeString()	Returns the time portion of a Date object in readable form
toUTCString()	Converts a Date object, according to universal time, to a string
UTC()	Takes a date and returns the number of milliseconds since midnight of January 1, 1970 according to universal time
valueOf()	Returns the primitive value of a Date object

Perguntas?



Exercícios

Exercícios – Lista



Data de Hoje

- Faça uma página HTML que mostre a data de hoje no formato:

Quinta-feira, 25 de setembro de 2013

Data de Nascimento

- Faça uma página que pergunte ao usuário os números do dia, do mês e do ano do nascimento dele.
- Informe a data completa, incluindo o dia da semana, por extenso do nascimento do usuário

Exercícios

- Faça uma página com um formulário com um campo de e-mail e um botão.
- O usuário deverá digitar o seu endereço de e-mail e a página deverá mostrar o domínio do e-mail do usuário.
 - Dicas:
 - O método de string chamado de `indexOf(texto)` devolve a primeira ocorrência de texto dentro da string
 - A propriedade de string chamada de `length` equivale ao número de caracteres da string
 - O método de string chamado de `substring(início, fim)` retorna uma parte da string

Exercícios

- Fazer um programa em TypeScript que leia duas strings e monte uma página indicando a posição de todas as ocorrências da segunda string na primeira.
 - Exemplo:
 - Primeira string: "Este é um texto que contém um texto com a palavra texto"
 - Segunda string: "texto"
 - Resultado: 10 30 50

Exercícios

- Fazer uma página que cada vez que é recarregada exibe um banner escolhido aleatoriamente. Desta vez, utilize um vetor de nomes de figuras.
 - Dica:
 - Para obter um número aleatório entre 0 e 4, utilize `parseInt(Math.random()*5)`
 - Coloque o endereço das figuras em um array de strings
 - Obs.: Utilize pelo menos 5 figuras diferentes.

Exercícios

- Obtenha a data de hoje (não é para ler) e monte uma página com o calendário, destacando o dia de hoje (em vermelho, por exemplo).

Calendário						
Fev. 2000						
Dom	Seg	Ter	Qua	Qui	Sex	Sab
			1	2	3	4
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29				

Exercícios

- Escreva uma função que recebe um caracter e retorna verdadeiro (true) se for uma vogal, falso (false) caso contrário.
- Faça uma página HTML para testar a sua função

Exercícios

- Defina uma função soma() e uma função multiplica() que soma e multiplica, respectivamente, todos os numeros em um vetor (array).
- Por exemplo:
 - soma([1,2,3,4]) deve retornar 10
 - multiplica([1,2,3,4]) deve retornar 24
- Faça uma página HTML para testar a sua função.

Exercícios

- Defina uma função `reverso()` que inverta uma string. Por exemplo, `reverso("um teste")` deve retornar "etset mu"
- Faça uma página HTML para testar a sua função

Exercícios

- Escreva uma função chamada achaAMaiorPalavra() que recebe um vetor de palavras e retorna a maior palavra.
- Faça uma página HTML para testar a sua função.

Exercícios

- **Rövarspråket** é um jogo Sueco onde se codifica um texto duplicando cada consoante e incluindo uma letra o entre cada uma delas. Vogais são mantidas intactas como no exemplo a seguir:
 - Este seria um texto!
 - Esostote soseroria umom totexoxtoto!
- Implemente uma página que contenha um formulário para implementar o jogo.

Exercício Extra de Array

(extraido de <https://www.codecademy.com/courses/javascript-beginner-en-OcLlk/0/2>)

- 1) Escreva o valor do terceiro elemento do vetor abaixo no log da console.
- 2) Escreva o tamanho do vetor abaixo no log da console.
- 3) Escreva todos os elementos do vetor no log da console, um elemento em cada linha.

```
var linguagensProgramacao = ["JavaScript", "Java", "C", "Fortran"];
```

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

- 11. Write a TypeScript program to find the sum of squares of a numeric vector.
- 12. Write a TypeScript program to compute the sum and product of an array of integers.
- 14. Write a TypeScript program to remove duplicate items from an array (ignore case sensitivity).

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

- 13. Write a TypeScript program to add items in an blank array and display the items.

Element 0 = 23
Element 1 = 12
Element 2 = 25

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

15. We have the following arrays:

- color = ["Blue ", "Green", "Red", "Orange", "Violet", "Indigo", "Yellow "];
- o = ["st", "nd", "rd", "th"]

Write a TypeScript program to display the colors in the following way :

- "1st choice is Blue ."
- "2nd choice is Green."
- "3rd choice is Red."
- - - - - -

Note : Use ordinal numbers to tell their position.

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

19. There are two arrays with individual values, write a TypeScript program to compute the sum of each individual index value from the given arrays.

Sample array :

- array1 = [1,0,2,3,4];
- array2 = [3,5,6,7,8,13];

Expected Output :

- [4, 5, 8, 10, 12, 13]

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

22. Write a TypeScript program to compute the union of two arrays.

Sample Data :

- `console.log(union([1, 2, 3], [100, 2, 1, 10]));`
- `[1, 2, 3, 10, 100]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

- 23. Write a TypeScript function to find the difference of two arrays.
- Test Data :
 - `console.log(difference([1, 2, 3], [100, 2, 1, 10]));`
 - `["3", "10", "100"]`
 - `console.log(difference([1, 2, 3, 4, 5], [1, [2], [3, [[4]]],[5,6]]));`
 - `["6"]`
 - `console.log(difference([1, 2, 3], [100, 2, 1, 10]));`
 - `["3", "10", "100"]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

8. Write a TypeScript program to find the most frequent item of an array.

Sample array:

- var arr1=[3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];

Sample Output:

- a (5 times)

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

30. Write a TypeScript function to merge two arrays and removes all duplicates elements.

Test data:

- var array1 = [1, 2, 3];
- var array2 = [2, 30, 1];
- console.log(merge_array(array1, array2));

Sample Output:

- [3, 2, 30, 1]

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

34. Write a TypeScript function to get nth largest element from an unsorted array.

Test Data:

- console.log(nthlargest([43, 56, 23, 89, 88, 90, 99, 652], 4));

Sample Output:

- 89

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

36. Write a TypeScript function to create a specified number of elements with pre-filled numeric value array.

Test Data :

- `console.log(array_filled(6, 0));`

Sample Output:

- `[0, 0, 0, 0, 0, 0]`

Test Data :

- `console.log(array_filled(4, 11));`

Sample Output:

- `[11, 11, 11, 11]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

37. Write a TypeScript function to create a specified number of elements with pre-filled string value array.

Test Data:

- `console.log(array_filled(3, 'default value'));`

Sample Output:

- `["default value", "default value", "default value"]`

Test Data:

- `console.log(array_filled(4, 'password'));`

Sample Output:

- `["password", "password", "password", "password"]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

40. Write a TypeScript function to generate an array of specified length, filled with integer numbers, increase by one from starting position.

Test Data:

- `console.log(array_range(1, 4));`

Sample Output:

- `[1, 2, 3, 4]`

Test Data:

- `console.log(array_range(-6, 4));`

Sample Output:

- `[-6, -5, -4, -3]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

41. Write a TypeScript function to generate an array between two integers of 1 step length.

Test Data:

- `console.log(rangeBetween(4, 7));`

Sample Output:

- `[4, 5, 6, 7]`

Test Data:

- `console.log(rangeBetween(-4, 7));`

Sample Output:

- `[-4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7]`

Exercício Extra de Array

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-array-exercises.php>)

42. Write a TypeScript function to find the unique elements from two arrays.

Test Data:

- `console.log(difference([1, 2, 3], [100, 2, 1, 10]));`

Sample Output:

- `["1", "2", "3", "10", "100"]`

Test Data:

- `console.log(difference([1, 2, 3, 4, 5], [1, [2], [3, [[4]]],[5,6]]));`

Sample Output:

- `["1", "2", "3", "4", "5", "6"]`

Test Data:

- `console.log(difference([1, 2, 3], [100, 2, 1, 10]));`

Sample Output:

- `["1", "2", "3", "10", "100"]`

Exercício Extra de Date

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-basic-exercises.php>)

- 7. Write a TypeScript program to find 1st January be a Sunday between 2014 and 2050.

- 9. Write a TypeScript program to calculate days left until next Christmas.

Exercício Extra de String

(extraido de <http://www.w3resource.com/javascript-exercises/javascript-functions-exercises.php>)

- 2. Write a TypeScript function that checks whether a passed string is palindrome or not.
 - A palindrome is word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run.
 - SOCORRAMMESUBINOONIBUSEMMARROCOS

Document Object Model (DOM) em TypeScript

DOM

- Podemos usar os mesmos objetos e métodos de JavaScript
- Muitas vezes, o TypeScript consegue inferir o tipo do objeto HTML referenciado

Objetos do browser

- O navegador cria automaticamente uma hierarquia de objetos refletindo alguns elementos inseridos na página.
- Os atributos da tag viram propriedades do objeto.
- Algumas propriedades podem ter seu valor modificado.
- O navegador mantém a coerência entre o valor da propriedade e o que está sendo visualizado pelo usuário.

O Objeto window

- É o objeto de mais alto nível na hierarquia de objetos do browser em JavaScript
- É o objeto default
- Criado automaticamente quando a página é carregada
- Como objeto default, pode ser omitido
 - `window.document.write(texto);`
 - `document.write(texto);`
- Cada janela possui um objeto window
- Inclui vários métodos e propriedades
 - <https://developer.mozilla.org/en-US/docs/Web/API/Window>

Objeto navigator

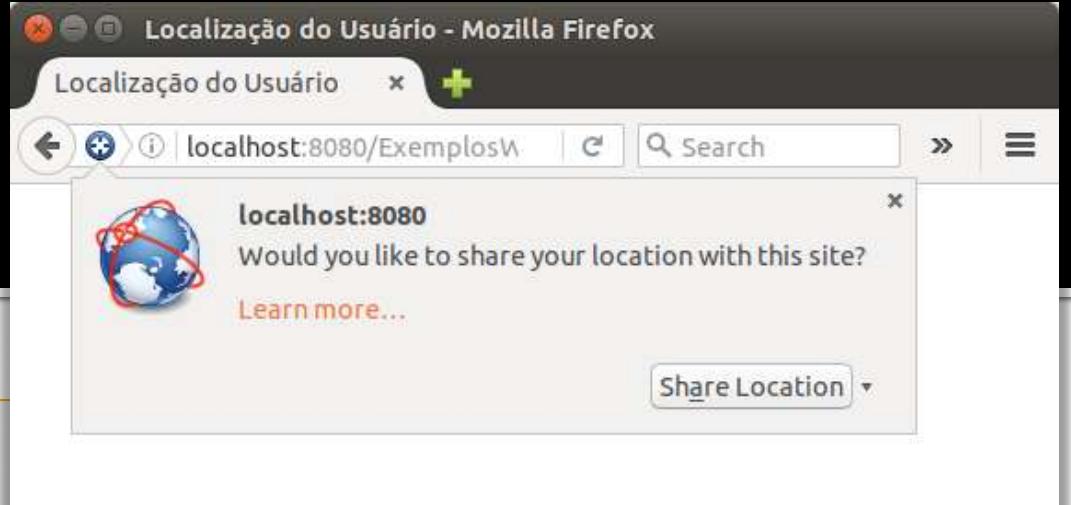
- Objeto navigator
 - Permite configurar as aplicações para navegadores diferentes
 - Propriedades:
 - appCodeName
 - appName
 - appVersion
 - platform
 - language (no explorer userLanguage e systemLanguage)

Objeto navigator

```
window.onload = () => {
    if(navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(funcaoSucesso, funcaoErro);
    }
}

function funcaoSucesso(posicao: any) {
    var lat = posicao.coords.latitude;
    var lng = posicao.coords.longitude;
    document.write("Latitude: " + lat + "<br/>Longitude: " + lng);
}

function funcaoErro() {
    document.write("Deu erro...");
}
```



Objeto navigator

```
window.onload = inicia;
function inicia() {
    if(navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(funcaoSucesso, funcaoErro);
    }
}

function funcaoSucesso(posicao) {
    var lat = posicao.coords.latitude;
    var lng = posicao.coords.longitude;
    document.write("Latitude: " + lat + "<br/>Longitude: " + lng);
}

function funcaoErro() {
    document.write("Deu erro...");
}
```

Latitude: -22.9343232
Longitude: -43.188224

DOM & Typecast

- O atributo valueAsNumber funciona com campos dos seguintes tipos:
 - datetime
 - date
 - month
 - week
 - time
 - datetime-local
 - number
 - range
- Não funciona com:
 - text

DOM.ts

```
onload = () => {
  const valor = document.getElementById('idTexto')
  as HTMLInputElement;
  console.log("texto: ", valor.value);
  console.log("Número:", valor.valueAsNumber);
}
```

O objeto document

- É o objeto mais importante em uma janela ou frame
- O objeto document representa um documento web ou uma página em um navegador
- Inclui vários métodos e propriedades
 - <https://developer.mozilla.org/en-US/docs/Web/API/Document>

JavaScript e DOM HTML

- TypeScript (JavaScript) pode ser utilizado para modificar o conteúdo ou os atributos de elementos HTML
 - Modificando o conteúdo:
`document.getElementById(id).innerHTML = novo HTML`
 - Modificando um atributo:
`document.getElementById(id).atributo = novo valor`
 - Modificando um estilo:
`document.getElementById(id).style.propriedade = novo estilo`

DOM & Typecast

DOM.ts

```
.onload = () => {
    // const link = document.querySelector('a');
    // console.log("link: ", link); <== errado porque link pode ser null, usar ! como mostrado abaixo
    const link = document.querySelector('a')!; // ! → programador garante que não é nulo
    console.log("Endereço: ", link.href);
    console.log("innerHTML: ", link.innerHTML);
    const links = document.querySelectorAll('a');
    console.log("links: ", links);

    const outroLink = document.getElementById('idCursos') as HTMLAnchorElement;
    console.log("Endereço: ", outroLink.href);
    console.log("innerHTML: ", outroLink.innerHTML);
}
```

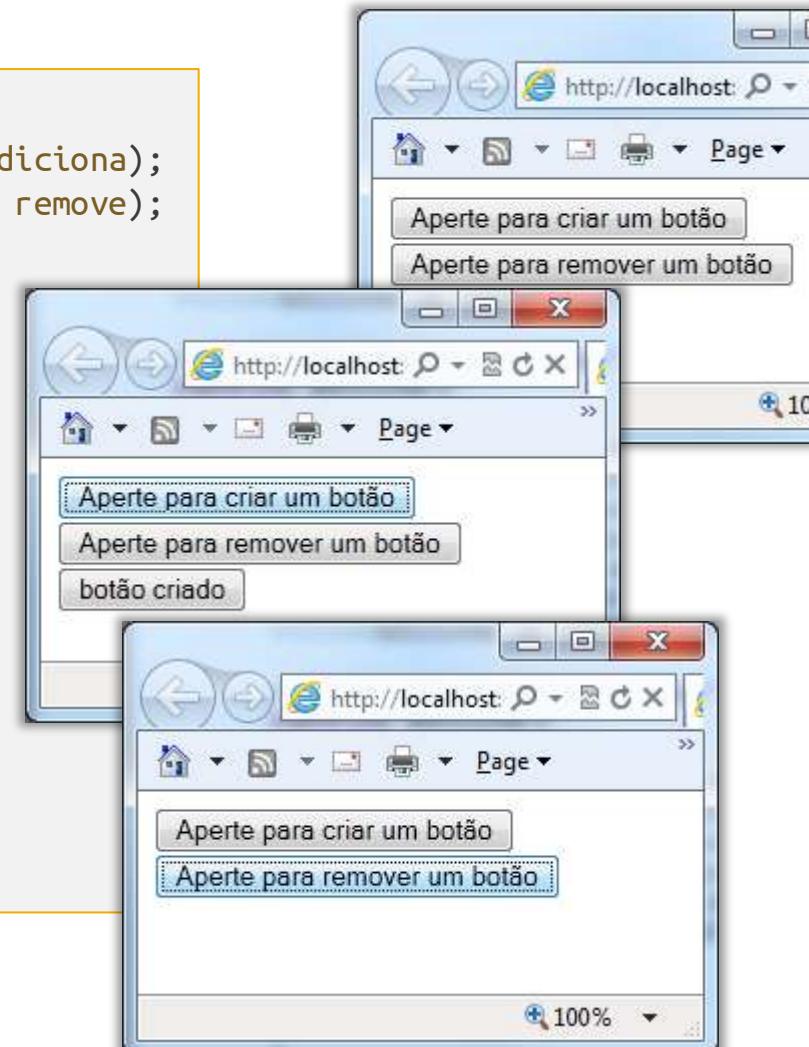
Criando um Objeto HTML

Objetos/DOM.ts

```
.onload = () => {
  (document.getElementById('idCria') as HTMLButtonElement).addEventListener('click', adiciona);
  (document.getElementById('idRemove') as HTMLButtonElement).addEventListener('click', remove);
}

function adiciona() {
  var campoInput = document.createElement("input");
  campoInput.setAttribute("type", "button");
  campoInput.setAttribute("value", "botão criado");
  campoInput.setAttribute("id", "idCriado");
  var formulario = document.getElementById("idFormulario") as HTMLFormElement;
  formulario.appendChild(campoInput);
}

function remove() {
  var campoCriado = document.getElementById("idCriado") as HTMLInputElement;
  campoCriado.remove();
}
```



Elementos HTML (HTMLElement)

HTMLElement	HTMLDialogElement	HTMLImageElement	HTMLObjectElement	HTMLSpanElement
HTMLAnchorElement	HTMLDirectoryElement	HTMLInputElement	HTMLOptGroupElement	HTMLStyleElement
HTMLAppletElement	HTMLDivElement	HTMLLIElement	HTMLOptionElement	HTMLTableCaptionElement
HTMLAreaElement	HTMLEmbedElement	HTMLLabelElement	HTMLOutputElement	HTMLTableCellElement
HTMLBRElement	HTMLFieldSetElement	HTMLLegendElement	HTMLParagraphElement	HTMLTableColElement
HTMLBaseElement	HTMLFontElement	HTMLLinkElement	HTMLParamElement	HTMLTableElement
HTMLBaseFontElement	HTMLFormElement	HTMLMapElement	HTMLPictureElement	HTMLTableRowElement
HTMLBodyElement	HTMLFrameElement	HTMLMarqueeElement	HTMLPreElement	HTMLTableSectionElement
HTMLButtonElement	HTMLFrameSetElement	HTMLMediaElement	HTMLProgressElement	HTMLTemplateElement
HTMLCanvasElement	HTMLHRElement	HTMLMenuElement	HTMLQuoteElement	HTMLTextAreaElement
HTMLDListElement	HTMLHeadElement	HTMLMetaElement	HTMLScriptElement	HTMLTimeElement
HTMLDataElement	HTMLHeadingElement	HTMLMeterElement	HTMLSelectElement	HTMLTitleElement
HTMLDataListElement	HTMLHtmlElement	HTMLModElement	HTMLSlotElement	HTMLTrackElement
HTMLDetailsElement	HTMLIFrameElement	HTMLOLListElement	HTMLSourceElement	HTMLULListElement

innerHTML x appendChild

innerHTML

- Precisa de escape()
- Mais lento
- Não valida a tag e seus atributos
- Sensível a ataques de inserção de código

appendChild

- createElement() já possui escape()
- Muito mais (muito mais mesmo) rápido
- Valida a tag e seus atributos
- Insensível a ataques de inserção de código

Você ainda quer usar innerHTML?

Objetos/innerHTML.ts

```
.onload = () => {
    (document.getElementById('idInnerHTML') as HTMLButtonElement).addEventListener('click', htmlDeDentro);
    (document.getElementById('idAppendChild') as HTMLButtonElement).addEventListener('click', acrescentaFilho);
}

var qtd = 256;
function htmlDeDentro() {
    var inicio = new Date().getTime();
    for(var i=0; i<qtd; i++)
        (document.getElementById("idDiv") as HTMLDivElement).innerHTML += "<input type=text>";
    alert ((new Date().getTime() - inicio) + " ms");
    while((document.getElementById("idDiv") as HTMLDivElement).firstChild) idDiv.removeChild(idDiv.firstChild);
}

function acrescentaFilho() {
    var inicio = new Date().getTime();
    for(var i=0; i<qtd; i++) {
        var campo = document.createElement("input");
        campo.setAttribute("type", "text");
        (document.getElementById("idDiv") as HTMLDivElement).appendChild(campo);
    }
    alert ((new Date().getTime() - inicio) + " ms");
    while((document.getElementById("idDiv") as HTMLDivElement).firstChild) idDiv.removeChild(idDiv.firstChild);
}
```

Nomes das Propriedades do Estilo

CSS

- background-color
- border-radius
- font-size
- list-style-type
- word-spacing
- z-index

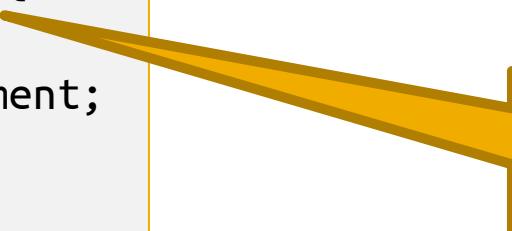
JavaScript

- backgroundColor
- borderRadius
- fontSize
- listStyleType
- wordSpacing
- zIndex

Tratamento de Eventos – Exemplo

DOM.ts

```
function mudaCor(evento: MouseEvent) {  
    var objeto: HTMLElement;  
    objeto = evento.target as HTMLElement;  
    objeto.style.color = "#808080";  
}
```



Veja daqui a pouco os tipos de eventos

Tratamento de Eventos – Exemplo

DOM.ts

```
function eventoDrop(evento: DragEvent) {
    evento.preventDefault();
    // ? para indicar que o programador garante que não é nulo
    var idBloco = evento.dataTransfer?.getData("bloco") as string;
    (evento.target as HTMLElement).appendChild(document.getElementById(idBloco) as HTMLElement);
}

function eventoDragOver(evento: DragEvent) {
    evento.preventDefault();
}

function eventoDragStart(evento: DragEvent) {
    evento.dataTransfer?.setData("bloco", (evento.target as HTMLElement).id);
}
```

Tipos Variáveis Eventos (Event) (não confundir com tipos de eventos)

AnimationEvent	DeviceOrientationEvent	InputEvent	PointerEvent	TrackEvent
BeforeUnloadEvent	DragEvent	KeyboardEvent	PopStateEvent	TransitionEvent
ClipboardEvent	ErrorEvent	MessageEvent	ProgressEvent	UIEvent
CloseEvent	FocusEvent	MouseEvent	RTCDataChannelEvent	WebGLContextEvent
CompositionEvent	GamepadEvent	OfflineAudioCompletionEvent	RTCPeerConnectionIceEvent	WheelEvent
CustomEvent	HashChangeEvent	PageTransitionEvent	StorageEvent	
DeviceMotionEvent	IDBVersionChangeEvent	PaymentRequestUpdateEvent	TouchEvent	Event

Como fazer para ...

- Obter o valor do texto de um `textNode` a partir do seu objeto-pai

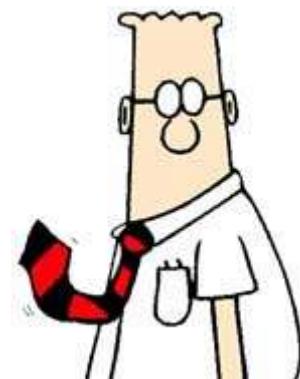
```
var texto = document.getElementById("idPai").firstChild.nodeValue;
```

Como fazer para ...

- Obter o texto de um campo input do tipo "text" ou "password":

```
var texto = document.getElementById('idNome').value;
```

Perguntas?





$$z = k + x$$

$$\frac{z_1}{4} = \frac{\sqrt{2}}{2}$$
$$z = z_{1x} + z_{1y}$$

Exercícios

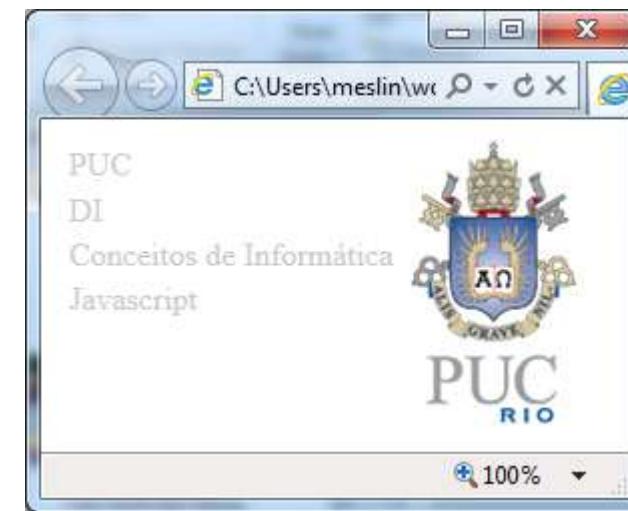
- Faça uma página HTML com uma figura. Ao clicar na figura, esta deverá mudar.

Exercício

- Faça uma página HTML contendo um formulário com os campos:
 - Nome: somente letras maiúsculas
 - Data de nascimento: formato dd/mm/aaaa
 - CEP: formato nnnnn-nnn
 - CPF: formato nnn.nnn.nnn-dd
- Ao selecionar o campo que será preenchido, o usuário deverá visualizar, no topo da página, as instruções referentes ao campos

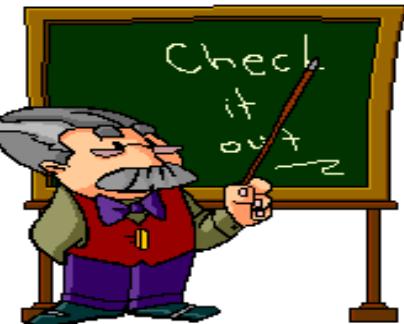
Exercício

- Implemente a página HTML mostrada na figura.
 - Ao passar o mouse no texto, este deverá ser habilitado (ao sair, desabilitado)
 - Ao clicar no texto, a figura deverá mudar
- Desafio: inclua uma descrição em formato texto abaixo da figura (a descrição muda com a figura)



Exercício

- Faça uma página HTML com vários links. Faça com que o estilo do link mude ao passar o mouse por cima. Não se esqueça de voltar ao estilo normal após o mouse sair (você escolhe o estilo).



Exercício

- Faça uma página com um formulário com um campo para o usuário informar o seu nome e outro para informar o gênero. Ao clicar no botão, uma janela de confirmação deverá ser aberta apresentando os dados informados.
 - 1) Use radio para selecionar o gênero
 - 2) Use select para selecionar o gênero
- Veja observação no próximo slide

Uso do campo <select> (múltiplo ou não)

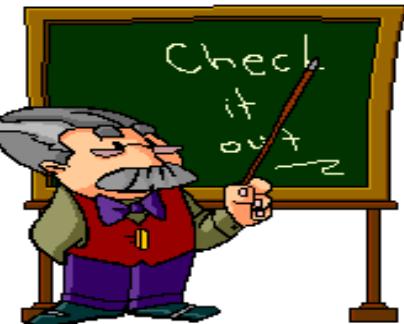
- Para saber o número de opções:
`document.formulario.campoSelect.length`
- Para saber o índice do último item selecionado:
`document.formulario.campoSelect.selectedIndex`
- Para referenciar uma opção em particular em um objeto do tipo select:
`document.formulario.campoSelect.options[indice]`
- Para referenciar o seu valor:
`document.formulario.campoSelect.options[indice].value`
- Para referenciar o seu texto:
`document.formulario.campoSelect.options[indice].text`
- Para saber se a opção foi selecionada (booleano):
`document.formulario.campoSelect.options[indice].selected`
- Para saber se a opção é pré-selecionada (booleano):
`document.formulario.campoSelect.options[indice].defaultSelected`

Exercício

- Faça uma página com um formulário contendo um campo para o usuário informar o seu nome e campos checkbox para que seja informado os tipos de atividades desejadas. Ao clicar no botão, uma janela de confirmação deverá ser aberta apresentando os dados informados.
- **Desafio:** modifique o exercício para que o usuário possa clicar no texto relacionado ao checkbox para marcar o checkbox

Exercícios

- Faça uma página HTML que possua dois botões, um para aumentar o tamanho do fonte e outro para diminuí-lo. Coloque um texto com vários parágrafos



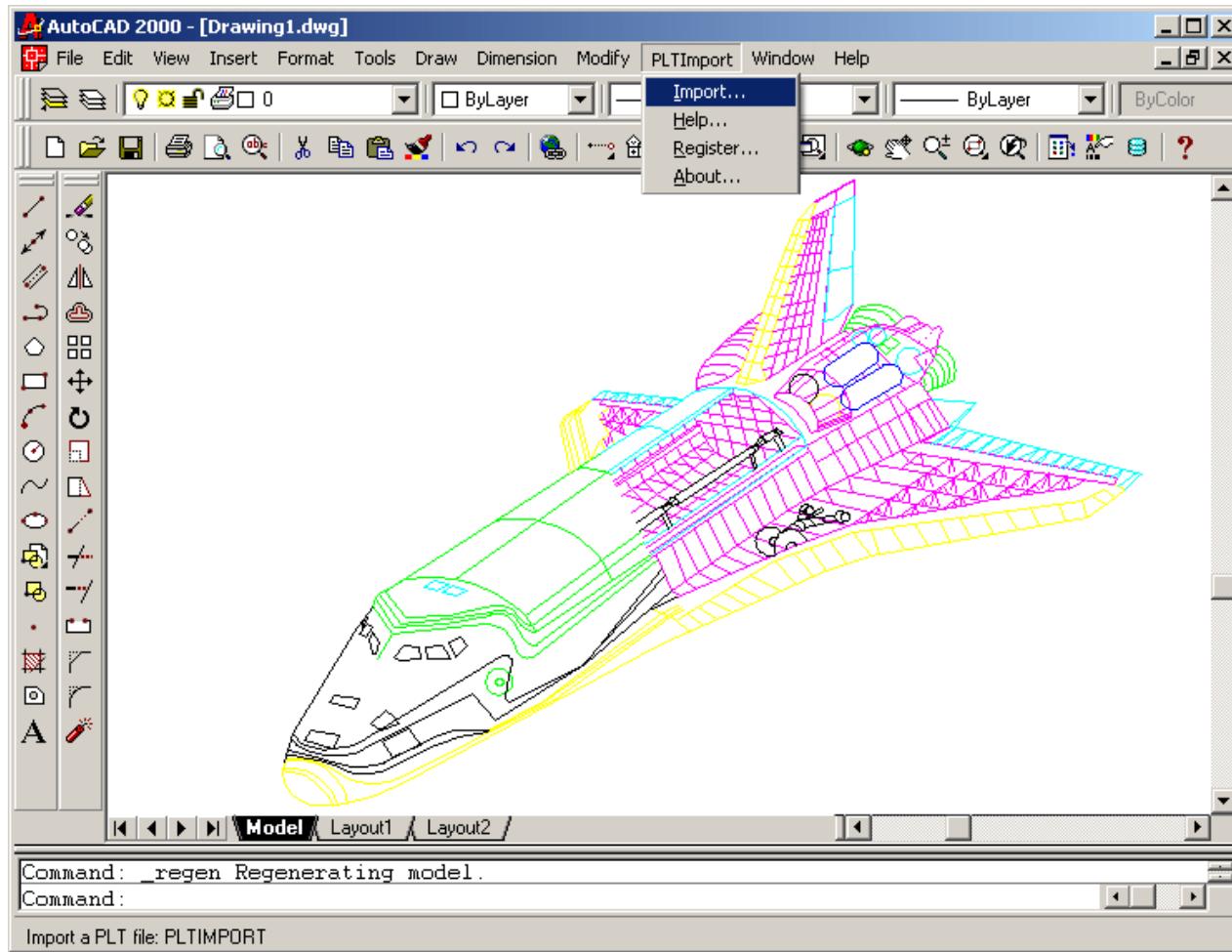
Desafio!

- Refaça o exercício do tamanho do fonte e inclua entre um parágrafo e outro, o exercício da figura que muda ao passar o mouse por cima

Desafio

- Faça um menu no estilo Windows (ou Mac ou Linux)

Definindo Classes e Objetos



Criando Objetos

- Objetos são definidos entre { }
 - Pares compostos por nome e valor
 - : separam nome de valor
 - , separam pares

- Exemplo:

```
var objeto = {x:8, y:"sete", z:5};  
console.log('Objeto: ', objeto.x);
```

Criando um objeto On-The-Fly

- Em Javascript...
 - Para criar um campo em um objeto, basta atribuir um valor a um campo.
 - Se o campo não existir, ele será criado naquele momento.
 - Ex.:

```
var objeto = new Object();
objeto.campo1 = "valor 1";
objeto.campo2 = 8752
alert(objeto.campo1 + objeto.campo2);
```
- Em TypeScript nada disso funciona.

Classes

classes.ts

```
class Pessoa {
  constructor(
    private nome: string,
    readonly email: string,
    public idade: number,
  ) {
  }

  cumprimenta(): void {
    // this.email = "outroEmail@example.com"; <== errado (readonly)
    console.log(`Olá, eu sou ${this.nome}`);
  }
}
```

Classes

classes.ts

```
class Pessoa {  
    constructor(  
        private nome: string,  
        readonly email: string,  
        public idade: number  
    ) {  
    }  
  
    cumprimenta(): void {  
        // this.email = "outroEmail@example.com";  
        console.log(`Olá, eu sou ${this.nome} e tenho ${this.idade} anos`);  
    }  
}
```

```
const pessoa = new Pessoa("Sandra", 'sandra@example.com', 15);  
pessoa.cumprimenta();  
  
let pessoas: Pessoa[] = [];  
pessoas.push(pessoa);  
  
// pessoa.nome = 'Paula'; <== errado (private)  
// pessoa.email = "outroEmail@example.com"; <== errado (readonly)  
  
pessoas.forEach(pessoa => {  
    // console.log("Pessoa:", pessoa.nome); <== errado (private)  
    console.log("Pessoa:", pessoa.email, pessoa.idade);  
});
```

Armadilhas

- Protected e Private
 - Em JavaScript não existe o conceito de métodos ou propriedades privadas
- Cuidado com o nome das propriedades
 - Se você fizer uma atribuição a uma propriedade que não exista em um objeto, esta passará a existir. Logo, cuidado com os nomes das propriedades e dos métodos.



Classes – Modificadores de Acesso

- Modificadores de acessos de atributos
 - public (default)
 - private
 - readonly

classes.ts

```
class Pessoa {  
    constructor(  
        private nome: string,  
        readonly email: string,  
        public idade: number,  
    ) {  
    }  
  
    cumprimenta(): void {  
        // this.email = "outroEmail@example.com"; <== errado  
        console.log(`Olá, eu sou ${this.nome}`);  
    }  
}
```

Classes – Modificadores de Acesso

- Modificadores de acessos de atributos
 - public (default)
 - private
 - readonly

```
const pessoa = new Pessoa("Sandra", 'sandra@example.com', 15);

//pessoa.nome = 'Paula'; <== errado (private)
//pessoa.email = "outroEmail@example.com"; <== errado (readonly)

pessoas.forEach(pessoa => {
  // console.log("Pessoa:", pessoa.nome); <== errado (private)
  console.log("Pessoa:", pessoa.email, pessoa.idade);
});
```

classes.ts

```
class Pessoa {
  private nome: string;
  readonly email: string;
  public idade: number;

  constructor(nome: string, email: string, idade: number) {
    this.nome = nome;
    this.email = email;
    this.idade = idade;
  }

  fala(): void {
    // this.email = "outroEmail@example.com"; <== errado
    console.log(`Olá, eu sou ${this.nome}`);
  }
}
```

Classes – Modificadores de Acesso e Construtores

- Uma forma de criar atributos
- Os atributos são criados juntamente com os parâmetros do construtor
 - Os modificadores de acesso precisam ser explicitados

classes.ts

```
class Pessoa {  
    constructor(  
        private nome: string,  
        readonly email: string,  
        public idade: number,  
    ) {  
    }  
}
```

Classes – Modificadores de Acesso e Construtores

- Outra forma de criar atributos
- Semelhante ao usado em Java

classes.ts

```
class Pessoa {  
    private nome: string;  
    readonly email: string;  
    public idade: number;  
  
    constructor(nome: string,  
                email: string,  
                idade: number) {  
        this.nome = nome;  
        this.email = email;  
        this.idade = idade;  
    }  
}
```

Interface

Interfaces.ts

```
interface Animal {  
    readonly especie: string;  
    anda(distancia: number): void;  
    fala(qtd: number): string;  
}  
  
abstract class Mamifero implements Animal {  
    peso: number;  
  
    constructor(public especie: string, peso: number) {  
        this.peso = peso;  
        console.log(`Criando ${this.especie} com ${this.peso} kg.`);  
    }  
  
    anda(distancia: number): void {  
        console.log(`${this.especie} andando ${distancia} metros.`);  
    }  
    abstract fala(qtd: number): string;  
}
```

Uma interface define um contrato, ou seja, os métodos e propriedades devem ser vistos publicamente.

O uso do atributo `public` em `especie` garante a inicialização da propriedade `especie`

Toda classe que implementar `Animal` precisa implementar `anda` e `fala`

Herança

Interfaces.ts

```
interface Animal {
    readonly especie: string;
    anda(distancia: number): void;
    fala(qtd: number): string;
}

abstract class Mamifero implements Animal {
    peso: number;

    constructor(public especie: string, p: number) {
        this.peso = p;
        console.log(`Criando ${this.especie}`);
    }

    anda(distancia: number): void {
        console.log(`${this.especie} andando ${distancia}m`);
    }
    abstract fala(qtd: number): string;
}
```

```
class Cachorro extends Mamifero {
    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Au! ';
        }
        return voz;
    }
}
```

Overload de construtor
new Porco(<peso>)
new Porco(<peso>, <espécie>)

```
class Porco extends Mamifero {
    constructor(peso: number, especie?: string) {
        if(especie === undefined) super('Porco', peso);
        else super(especie, peso);
    }

    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Ronc! ';
        }
        return voz;
    }
}
```

Herança

Interfaces.ts

```
interface Animal {
    readonly especie: string;
    anda(distancia: number): void;
    fala(qtd: number): string;
}

abstract class Mamifero implements Animal {
    peso: number;

    constructor(public especie: string, peso: number) {
        this.peso = peso;
        console.log(`Criando ${this.especie}`);
    }

    anda(distancia: number): void {
        console.log(`${this.especie} andando ${distancia}m`);
    }
    abstract fala(qtd: number): string;
}
```

```
class Cachorro extends Mamifero {
    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Au! ';
        }
        return voz;
    }
}

class Porco extends Mamifero {
    constructor(peso: number) {
        if(especie === 'Porco') {
            super(especie, peso);
        } else super(especie, peso);
    }

    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Ronc! ';
        }
        return voz;
    }
}

let porco = new Porco(52, "Porco");
let outroPorco = new Porco(87);
let cachorro = new Cachorro("Cachorro", 25);
console.log(porco.fala(5));
console.log(cachorro.fala(2));
porco.anda(3);
cachorro.anda(4);
```

Herança

Interfaces.ts

```
interface Animal {
    readonly especie: string;
    anda(distancia: number): void;
    fala(qtd: number): string;
}

abstract class Mamifero implements Animal {
    peso: number;

    constructor(public especie: string, peso: number) {
        this.peso = peso;
        console.log(`Criando ${this.especie}`);
    }

    anda(distancia: number): void {
        console.log(`${this.especie} andando ${distancia}m`);
    }

    abstract fala(qtd: number): string;
}
```

```
class Cachorro extends Mamifero {
    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Au! ';
        }
        return voz;
    }
}

class Porco extends Mamifero {
    constructor(peso: number) {
        if(especie === 'Porco') {
            super(especie, peso);
        } else super(especie, peso);
    }

    fala(qtd: number): string {
        let voz = '';
        for(let i=0; i<qtd; i++) {
            voz += 'Ronc! ';
        }
        return voz;
    }
}

let porco = new Porco(52, "Porco");
let outroPorco = new Porco(87);
let cachorro = new Cachorro("Cachorro", 25);
console.log(porco.fala(5));
console.log(cachorro.fala(2));
porco.anda(3);
cachorro.anda(4);

let arcaDeNoe: Animal[] = [];
arcaDeNoe.push(porco);
arcaDeNoe.push(cachorro);
console.log("Na Arca, temos", arcaDeNoe);
```

Objetos em TypeScript



Exemplo: Whac-A-Mole

Desenvolva o jogo Whac-A-Mole.

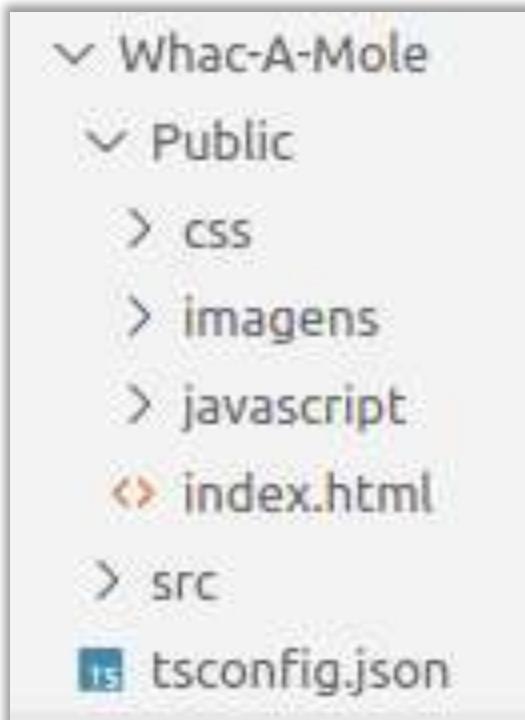
Utilize as imagens disponibilizadas para criar um tabuleiro com 5 buracos de toupeira dispostos em duas linhas. Inicialmente as toupeiras estarão escondidas nos buracos. Aleatoriamente, suba uma toupeira por alguns segundos. Se o usuário clicar na toupeira com ela aparecendo, ele deve ser pontuado. Caso ele clique no buraco sem toupeira, ele deverá ser penalizado. Crie uma penalidade para o usuário de deixar de clicar em uma toupeira quando ela aparecer.

Algumas informações (in)úteis:

- Os arquivos com as toupeiras ou buracos tem 76 px de largura por 61 px de altura.
- Use a cor #853e04 (marrom) ou #567d46 (verde grama) como background para compor a terra a volta dos buracos.
- Você conhece os seguintes estilos?
 - background-color
 - height
 - padding
 - width

Exemplo: Whac-A-Mole

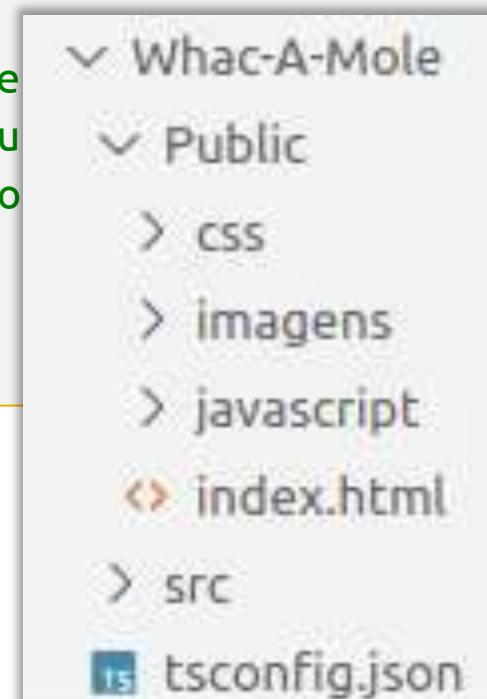
■ Estrutura dos diretórios



Exemplo: Whac-A-Mole

tsconfig.json

```
{  
  "compilerOptions": {  
    /* Basic Options */  
    "target": "es2015", /* Specify ECMAScript target version */  
    "outDir": "../Public/javascript", /* Redirect output */  
    "rootDir": "./src", /* Specify the root directory of the project */  
  },  
  "include": ["src"]  
}
```



, 'ES5', 'ES2015', 'ES2017' target. The 'outDir' option controls the output directory. The 'rootDir' option controls the root directory of the project. The 'include' option specifies the source files to be compiled.

Exemplo: Whac-A-Mole

Public/index.html

```
<!DOCTYPE html>
<html lang="pt-BR" dir="ltr">
<head>
<title>Whac-A-Mole</title>
<link href='css/estilo.css' rel='stylesheet'>
<script src="javascript/mole.js" charset="utf-8"></script>
</head>
<body>
<h1>Whack-A-Mole</h1>
<div class="mesa" id='idMesa'>
<img src='imagens/scene2.png' id='buraco0' class='buraco' alt='mole at top or no mole'>
<img src='imagens/scene2.png' id='buraco1' class='buraco' alt='mole at top or no mole'>
<img src='imagens/scene2.png' id='buraco2' class='buraco' alt='mole at top or no mole'>
<br>
<img src='imagens/scene2.png' id='buraco3' class='buraco' alt='mole at top or no mole'>
<img src='imagens/scene2.png' id='buraco4' class='buraco' alt='mole at top or no mole'>
</div>
<div>Pontuação: <span id='idPontos'>0</span></div>
</body>
</html>
```

Exemplo: Whac-A-Mole

Public/css/estilo.css

```
.buraco {  
    padding: 25px 50px 25px 50px;  
}  
  
.mesa {  
    width: 536px;  
    background-color: #2E8B57;  
    text-align: center;  
    cursor: url(../imagens/hammer.png), default;  
}
```

Exemplo: Whac-A-Mole

src/mole.ts

```
let ponto = 0;
onload = function() {
    setInterval(sobeToupeira, 5 * 1000);
    for(var i=0; i<5; i++)
        document.getElementById('buraco' + i)!.addEventListener('click', martelada);
    document.getElementById('idMesa')!.addEventListener('mousedown', marteloBaixo);
    document.getElementById('idMesa')!.addEventListener('mouseup', marteloCima);
}

function sobeToupeira() {
    var buraco = Math.floor(Math.random() * 5);
    var objBuraco = document.getElementById('buraco' + buraco) as HTMLImageElement;
    objBuraco.src = 'imagens/scene_top.png';
    setTimeout(tiraToupeira, 2 * 1000, buraco);
}

function tiraToupeira(buraco: number) {
    var objBuraco = document.getElementById('buraco' + buraco) as HTMLImageElement;
    objBuraco.src = 'imagens/scene2.png';
}
```

Exemplo: Whac-A-Mole

src/mole.ts

```
function martelada(evento: MouseEvent) {
    if((evento.target as HTMLImageElement).src.includes('scene_top')) {
        // acertou
        ponto++;
        (evento.target as HTMLImageElement).src = 'imagens/scene2.png';
    } else {
        // errou
        ponto--;
    }
    (document.getElementById('idPontos') as HTMLSpanElement).innerHTML = ponto.toString();
}

function marteloBaixo() {
    (document.getElementById('idMesa') as HTMLDivElement).style.cursor = 'url(imagens/hammerDown.png), default';
}

function marteloCima() {
    (document.getElementById('idMesa') as HTMLDivElement).style.cursor = 'url(imagens/hammer.png), default';
}
```

Exemplo: Whac-A-Mole

Public/javascript/mole.js

```
"use strict";
let ponto = 0;
onload = function () {
    setInterval(sobeToupeira, 5 * 1000);
    for (var i = 0; i < 5; i++)
        document.getElementById('buraco' + i).addEventListener('click', martelada);
    document.getElementById('idMesa').addEventListener('mousedown', marteloBaixo);
    document.getElementById('idMesa').addEventListener('mouseup', marteloCima);
};
function sobeToupeira() {
    var buraco = Math.floor(Math.random() * 5);
    var objBuraco = document.getElementById('buraco' + buraco);
    objBuraco.src = 'imagens/scene_top.png';
    setTimeout(tiraToupeira, 2 * 1000, buraco);
}
function tiraToupeira(buraco) {
    var objBuraco = document.getElementById('buraco' + buraco);
    objBuraco.src = 'imagens/scene2.png';
}
function martelada(evento) {
    if (evento.target.src.includes('scene_top')) {
        ponto++;
        evento.target.src = 'imagens/scene2.png';
    } else {
        ponto--;
    }
    document.getElementById('idPontos').innerHTML = ponto.toString();
}
function marteloBaixo() {
    document.getElementById('idMesa').style.cursor = 'url(imagens/hammerDown.png), default';
}
function marteloCima() {
    document.getElementById('idMesa').style.cursor = 'url(imagens/hammer.png), default';
}
```

Exercícios de TypeScript

Exercício

- Refaça os exercícios de JavaScript usando TypeScript

Manipulação de Erros e Exceções



Manipulação de Erros e Exceções

- Erros ou exceções ocorrem durante o tempo de execução
- Como contornar erros e exceções:
 - Utilizar o evento onerror
 - Utilizar o objeto Error
 - Utilizar o comando throw
 - Utilizar o block try-catch-finally

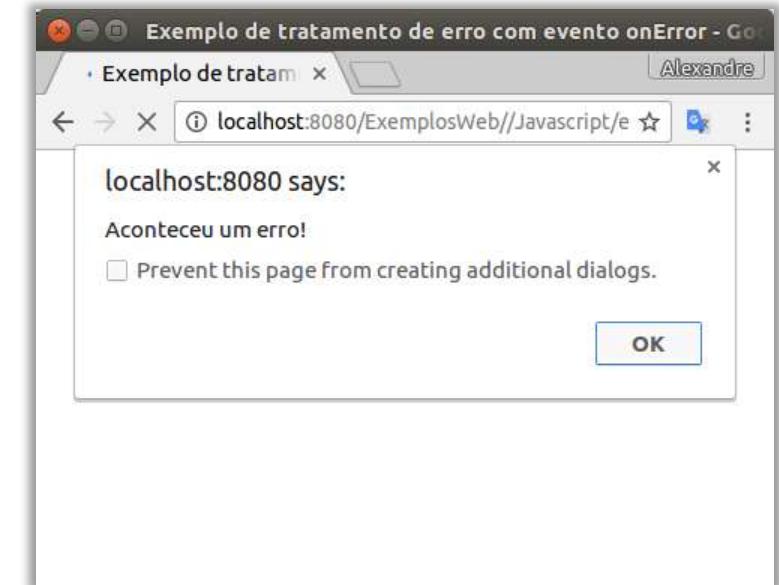
Quando ocorrem erros

- Erros podem ocorrer devido a:
 - Erros de sintaxe: algum comando escrito errado
 - Erros de execução: o script tenta executar alguma coisa que o browser não pode

Uso de onerror

onerror.ts

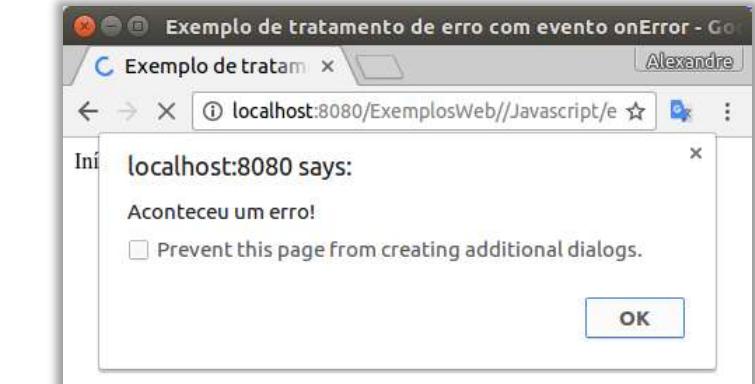
```
function manipuladorDeErros() {  
    alert("Aconteceu um erro!");  
}  
  
window.onerror = manipuladorDeErros;  
  
document.writeln("Início...");  
document.writeln(error de sintaxe);  
document.writeln("FIM!");
```



Uso de onerror

onerror.ts

```
function manipuladorDeErros() {  
    alert("Aconteceu um erro!");  
}  
  
window.onerror = manipuladorDeErros;  
  
document.writeln("Início...");  
var a: any = b;  
document.writeln("FIM!");
```

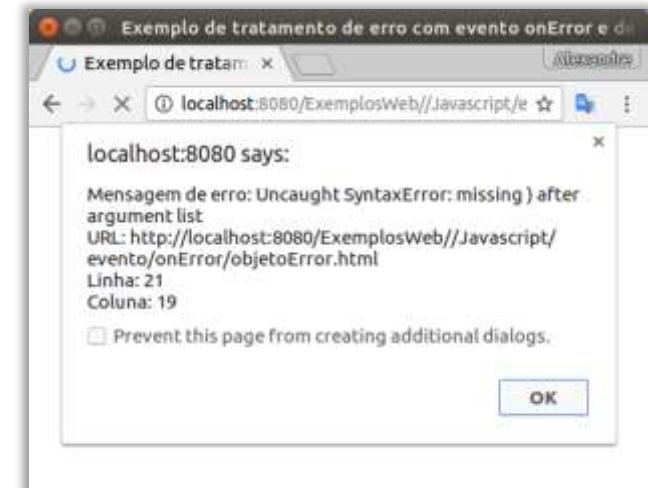


Uso de onerror com detalhes do erro

onerror.ts

```
onerror = function(msg, url, linha, coluna) {
    alert("Aconteceu um erro!");
    alert('Mensagem de erro: ' + msg + '\nURL: ' + url + '\nLinha: '
    + linha + '\nColuna: ' + coluna);
};

document.writeln("Início...");
document.writeln(error de sintaxe);
document.writeln("FIM!");
```



Manipulação de Exceção em TypeScript

- Exceção é um erro gerado pelo script – uma operação inválida
- O código que manipula a exceção é chamado de handler
- Sintaxe:

```
try {  
    código normal que pode resultar em exceção  
}  
catch(objetoError) {  
    código para tratar o erro  
}  
finally { // opcional  
    código que será executado com ou sem erro  
}
```

O Objeto Error

Error Constructors

Name	MSIE	NNav	Notes
Error()	5	6	

Error Methods

Name	MSIE	NNav	Notes
toString()	5	6	

Error Functions

Name	MSIE	NNav	Notes
Error()	5	6	

Error Properties

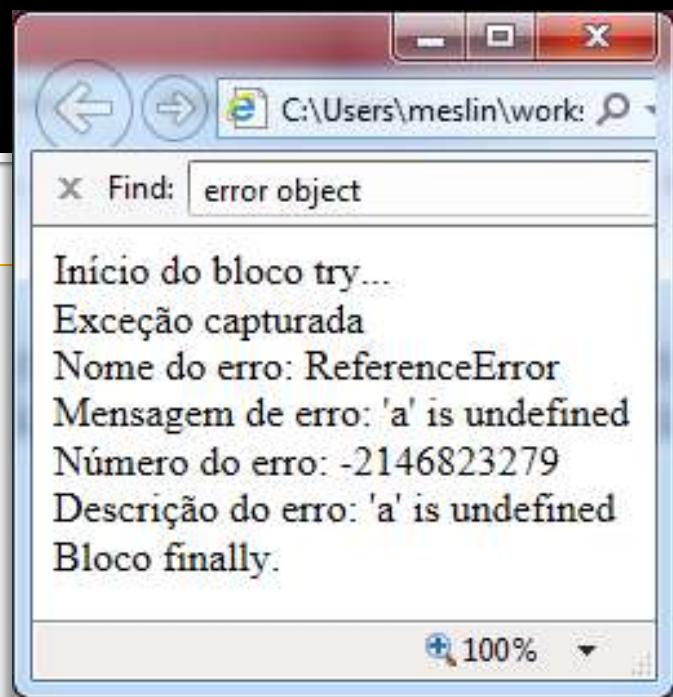
Name	MSIE	NNav	Notes
constructor	5	6	
description	5	-	The ECMA standard describes this property as Error.message. MSIE supports it as the description property.
message	5.5	6	
name	5.5	6	
number	5	6	
prototype	5	6	

Nomes (name) de Exceções

- O TypeScript (JavaScript) pode produzir exceções com os seguintes nomes (name):
 - 'Error'
 - 'EvalError'
 - 'RangeError'
 - 'SyntaxError'
 - 'TypeError'
 - 'URIError'

Exemplo do uso de try-catch-finally

```
try {  
    document.write("Início do bloco try...<br/>");  
    // erro  
    var x = parseInt(a);  
} catch(e) {    // a variável e deve ser any ou unknown  
    document.writeln("Exceção capturada<br/>");  
    document.writeln("Nome do erro: " + (e as Error).name + "<br/>");  
    document.writeln("Mensagem de erro: " + (e as Error).message + "<br/>");  
    document.writeln("Pilha no momento do erro: " + (e as Error).stack + "<br/>");  
} finally {  
    document.write("Bloco finally.<br/>")  
}
```



O comando throw

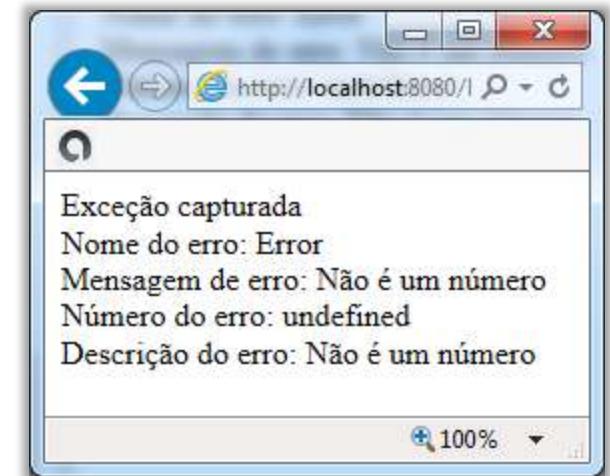
- Sintaxe:

```
throw new Error(motivo);
```

```
throw {  
    name: nomeDaExceção,  
    message: motivo  
}
```

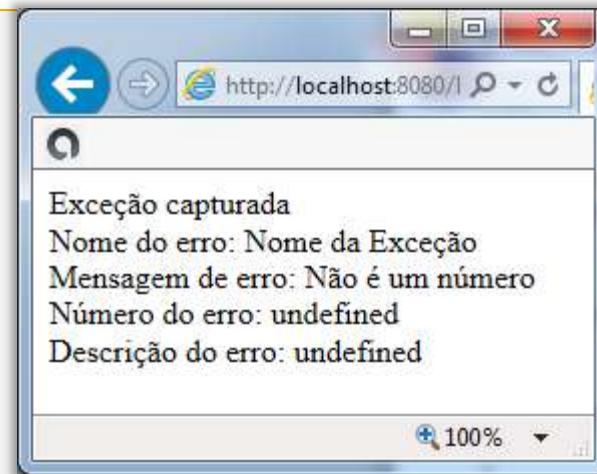
Exemplo de uso de throw

```
let a:number = NaN;  
try {  
    if(isNaN(a))  
        throw new Error("Não é um número");  
} catch(e) {  
    document.writeln("Exceção capturada<br/>");  
    document.writeln("Nome do erro: " + (e as Error).name + "<br/>");  
    document.writeln("Mensagem de erro: " + (e as Error).message + "<br/>");  
    document.writeln("Pilha no momento do erro: " + (e as Error).stack + "<br/>");  
}
```



Outro exemplo de uso de throw

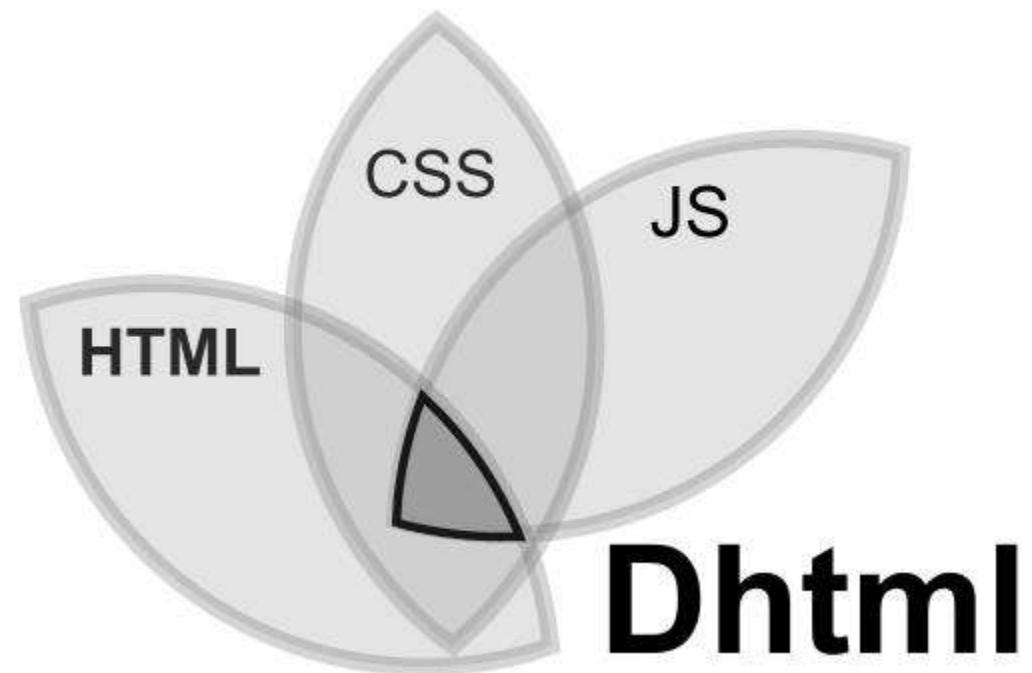
```
let a = NaN;  
try {  
    if(isNaN(a))  
        throw {  
            name: "Nome da Exceção",  
            message: "Não é um número"  
        }  
} catch(e) {  
    document.writeln("Exceção capturada<br/>");  
    document.writeln("Nome do erro: " + e.name + "<br/>");  
    document.writeln("Mensagem de erro: " + e.message + "<br/>");  
    document.writeln("Número do erro: " + e.number + "<br/>");  
    document.writeln("Descrição do erro: " + e.description + "<br/>");  
}
```



Perguntas?



DHTML



DHTML

Da Wikipedia:

- **Dynamic HTML**, or **DHTML**, is an umbrella term for a collection of technologies used together to create interactive and animated web sites by using a combination of a static markup language (such as HTML), a client-side scripting language (such as JavaScript), a presentation definition language (such as CSS), and the Document Object Model.
- DHTML allows scripting languages to change variables in a web page's definition language, which in turn affects the look and function of otherwise "static" HTML page content, *after* the page has been fully loaded and during the viewing process. Thus the dynamic characteristic of DHTML is the way it functions while a page is viewed, not in its ability to generate a unique page with each page load.
- By contrast, a dynamic web page is a broader concept, covering any web page generated differently for each user, load occurrence, or specific variable values. This includes pages created by client-side scripting, and ones created by server-side scripting (such as PHP, Perl, JSP or ASP.NET) where the web server generates content before sending it to the client.
- DHTML is differentiated from AJAX by the fact that a DHTML page is still request/reload-based. With DHTML, there may not be any interaction between the client and server after the page is loaded; all processing happens in JavaScript on the client side. By contrast, an AJAX page uses features of DHTML to initiate a request (or 'subrequest') to the server to perform actions such as loading more content.

Adicionando algum texto em uma página

- Passos
 - 1. Criar um novo elemento
 - 2. Criar um novo texto
 - 3. Adicionar o novo texto ao novo elemento
 - 4. Procurar um elemento já existente
 - 5. Adicionar o novo elemento ao elemento já existente

1) Criando um novo elemento

- Criar um objeto parágrafo <p>
- Associar o novo objeto a uma variável

```
var node: HTMLParagraphElement;  
node = document.createElement("p");
```

2) Criar um novo texto

- Para facilitar, o texto será atribuído a uma variável (opcional)

```
let novoTexto: Text;  
novoTexto = document.createTextNode("Um texto.");
```

3) Adicionar o novo texto ao novo elemento

- Para que o texto apareça na página Web, ele deverá ser adicionado ao novo elemento

```
node.appendChild(novoTexto);
```

4) Procurar um elemento existente

- O texto ainda não foi incluído na página Web
- O novo elemento precisa ser incluído em um elemento já existente
- Para facilitar, o elemento existente será atribuído a uma variável

```
let elemento: HTMLDivElement;  
elemento = document.getElementById("id") as HTMLDivElement;
```

5) Adicionar o novo elemento ao elemento já existente

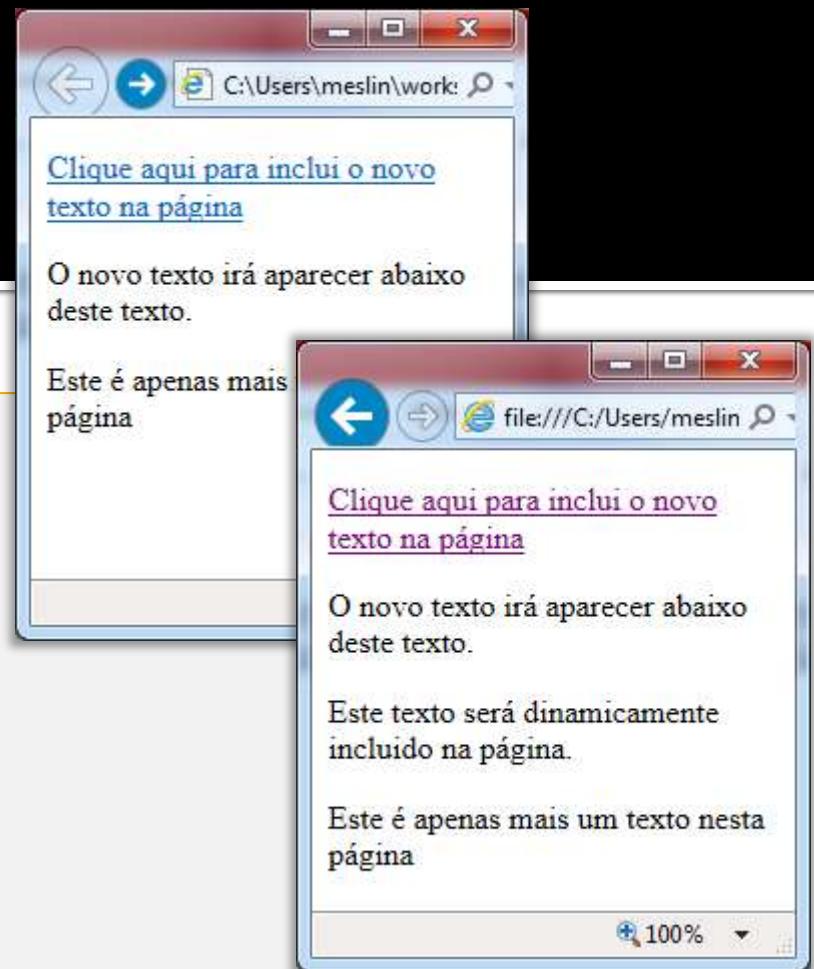
- Finalmente incluir o novo elemento e o seu texto a um elemento já existente

```
elemento.appendChild(node);
```

Colocando tudo junto...

dhtml.ts

```
var texto = "Este texto será dinamicamente incluído na página.";  
function incluiTexto(lugar: string) {  
    var node: HTMLParagraphElement;  
    node = document.createElement("p");  
    let novoTexto: Text;  
    novoTexto = document.createTextNode("Um texto.");  
    node.appendChild(novoTexto);  
    let elemento: HTMLDivElement;  
    elemento = document.getElementById(lugar) as HTMLDivElement;  
    elemento.appendChild(node);  
}
```



Removendo um nó

- Para remover um nó, podemos utilizar o método `removeChild(id)`
- Exemplo:

```
function removeTexto(lugar: string) {  
    var elemento: HTMLElement;  
    elemento = document.getElementById(lugar) as HTMLElement;  
    if(elemento != null && elemento.lastChild != null)  
        elemento.removeChild(elemento.lastChild);  
}
```

Exemplo completo...

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Exemplo de inclusão de elemento</title>
<script src="js/dhtml.js"></script>
</head>

<body id='corpo'>
<p><a id='link'>
Clique aqui para inclui o novo texto na página.
</a></p>
<button id='button'>
Clique aqui para remover o novo texto da página
</button>
<div id="lugar">
0 novo texto irá aparecer abaixo deste texto.
</div>
<p>Este é apenas mais um texto nesta página.</p>
</body>
</html>
```

```
onload = () => {
    (document.getElementById('link') as HTMLLinkElement).addEventListener('click', () =>
incluirTexto('lugar'));
    (document.getElementById('button') as HTMLButtonElement).addEventListener('click', () =>
removerTexto('lugar'));
    // Variável global definida dentro da função
    window.texto = "Este texto será dinamicamente incluído na página.";
}

function incluirTexto(lugar: string) {
    var node: HTMLParagraphElement;
    node = document.createElement("p");
    let novoTexto: Text;
    novoTexto = document.createTextNode(texto);
    node.appendChild(novoTexto);
    let elemento: HTMLDivElement;
    elemento = document.getElementById(lugar) as HTMLDivElement;
    elemento.appendChild(node);
}

function removerTexto(lugar: string) {
    var elemento: HTMLElement;
    elemento = document.getElementById(lugar) as HTMLElement;
    if(elemento != null && elemento.lastChild != null)
        elemento.removeChild(elemento.lastChild);
}
```

Exemplo completo...

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Exemplo de inclusão de elemento</title>
<script src="js/dhtml.js"></script>
</head>

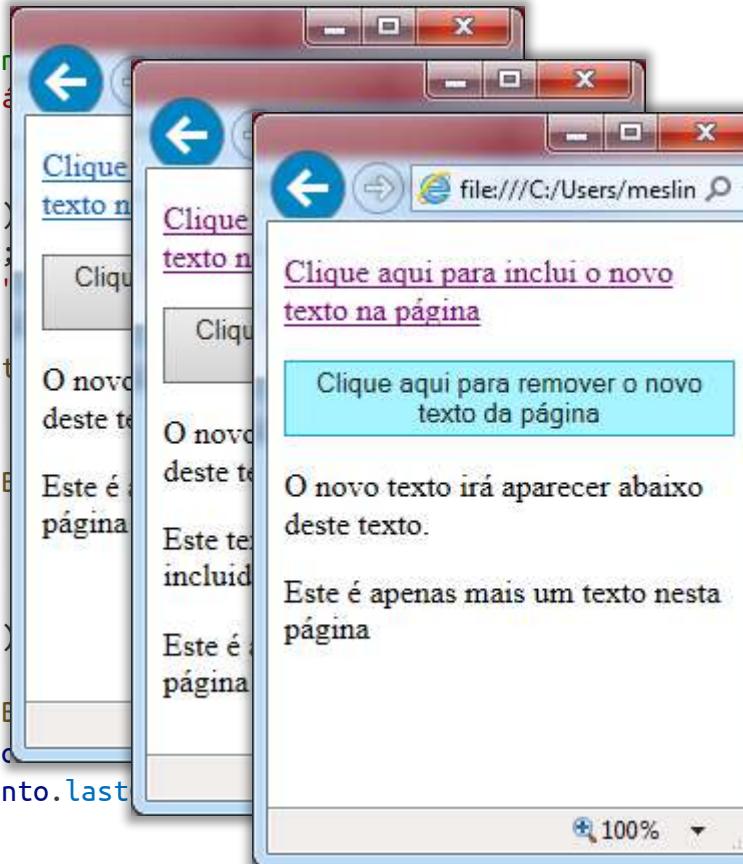
<body id='corpo'>
<p><a id='link'>
Clique aqui para inclui o novo texto na página.
</a></p>
<button id='button'>
Clique aqui para remover o novo texto da página
</button>
<div id="lugar">
O novo texto irá aparecer aqui.
</div>
<p>Este é apenas mais um texto na página.
</body>
</html>
```

Como fazer para selecionar qual texto será removido?

```
onload = () => {
  (document.getElementById('link') as HTMLLinkElement).addEventListener('click', () =>
incluirTexto('lugar'));
  (document.getElementById('button') as HTMLButtonElement).addEventListener('click', () =>
removeTexto('lugar'));
  // Variável global definida dentro da função
  window.texto = "Este texto será removido quando o botão for clicado";
}

function incluirTexto(lugar: string) {
  var node: HTMLParagraphElement;
  node = document.createElement('p');
  let novoTexto: Text;
  novoTexto = document.createTextNode("Clique aqui para inclui o novo texto na página");
  node.appendChild(novoTexto);
  let elemento: HTMLDivElement;
  elemento = document.getElementById(lugar);
  elemento.appendChild(node);
}

function removeTexto(lugar: string) {
  var elemento: HTMLElement;
  elemento = document.getElementById(lugar);
  if(elemento != null && elemento.lastChild != null) {
    elemento.removeChild(elemento.lastChild);
  }
}
```

A screenshot of a Windows desktop environment showing three overlapping browser windows. The top window is a standard browser window with the URL 'file:///C:/Users/meslin' visible. It contains a paragraph of text: 'Clique aqui para inclui o novo texto na página'. Below it is a smaller window with the same text. A third window is partially visible behind them, containing the text 'Este é apenas mais um texto na página'. All three windows have a blue circular button in the top-left corner, likely for closing or minimizing.

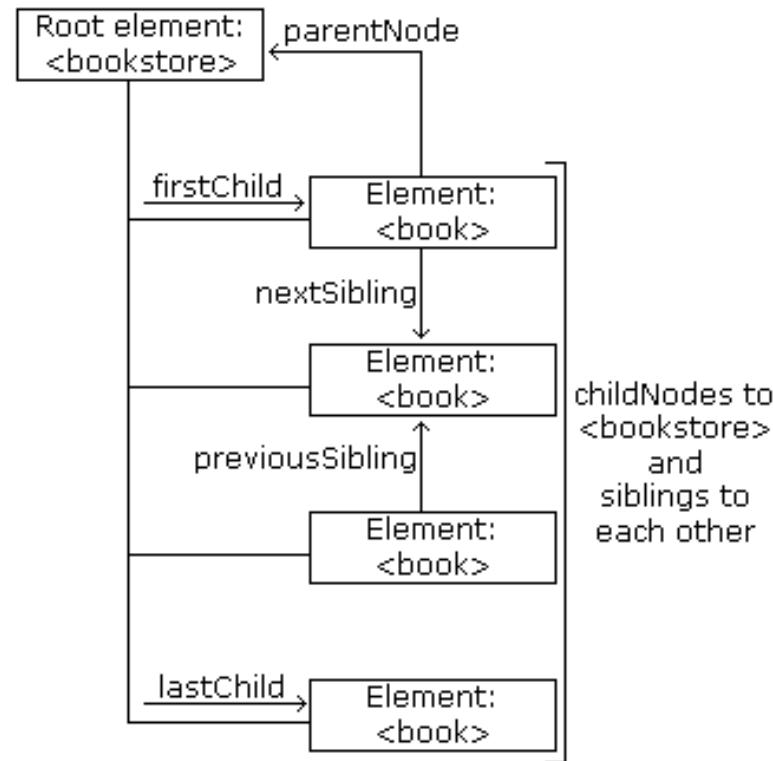
Método `getElementsByName()`

- `getElementById()` permite trabalhar elementos individualmente
- `getElementsByName()` permite trabalhar com grupos de elementos
 - Retorna um vetor (array)

Propriedades de node

- Algumas propriedades de node para ajudar a percorrer a árvore de nós:

```
let no: ParentNode | ChildNode | null;  
let nos: NodeList<ChildNode>;  
let node = document.getElementById('id') as HTML  
no = node.parentNode;  
nos = node.childNodes;  
no = node.firstChild;  
no = node.lastChild;  
no = node.nextSibling;  
no = node.previousSibling;
```



Atributos dos nós

```
let atributo = node.getAttribute("atributo");  
  
node.setAttribute('atributo', 'valor');
```

Exercícios



Exercício – Jogo dos 15

- Implemente o jogo ao lado
- O objetivo é colocar todos os números em sequência crescente, não importando onde fica o espaço em branco

8	7	5	2
3	6	9	10
11	12	13	4
14	15		1

Fazendo deployment

Implantando com o Heroku ou pythonanywhere

Fazendo deployment

pythonanywhere

- <https://www.pythonanywhere.com/>

Heroku

- <https://www.heroku.com/>

A2 Hosting (pago)

- <https://www.a2hosting.com/django-hosting>

AWS (pago)

- <https://aws.amazon.com/>

HostUpon (pago)

- <https://hostupon.com/index.php>

TMD Hosting (pago)

- <https://www.netguru.com/blog/django-hosting>

DigitalOcean (pago)

- <https://www.digitalocean.com/>

(Aceito sugestões)

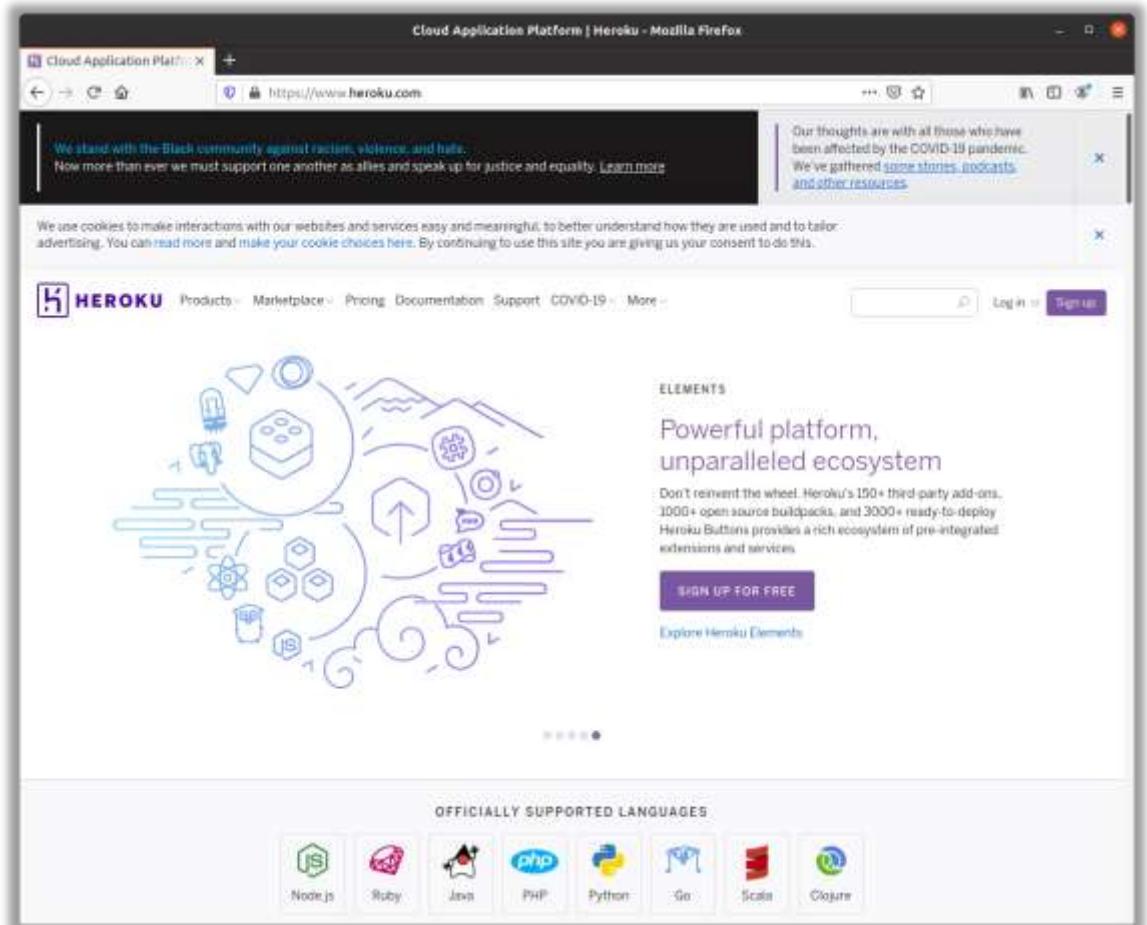
- (sua sugestão aqui)

Fazendo deployment

Usando a interface Web do Heroku

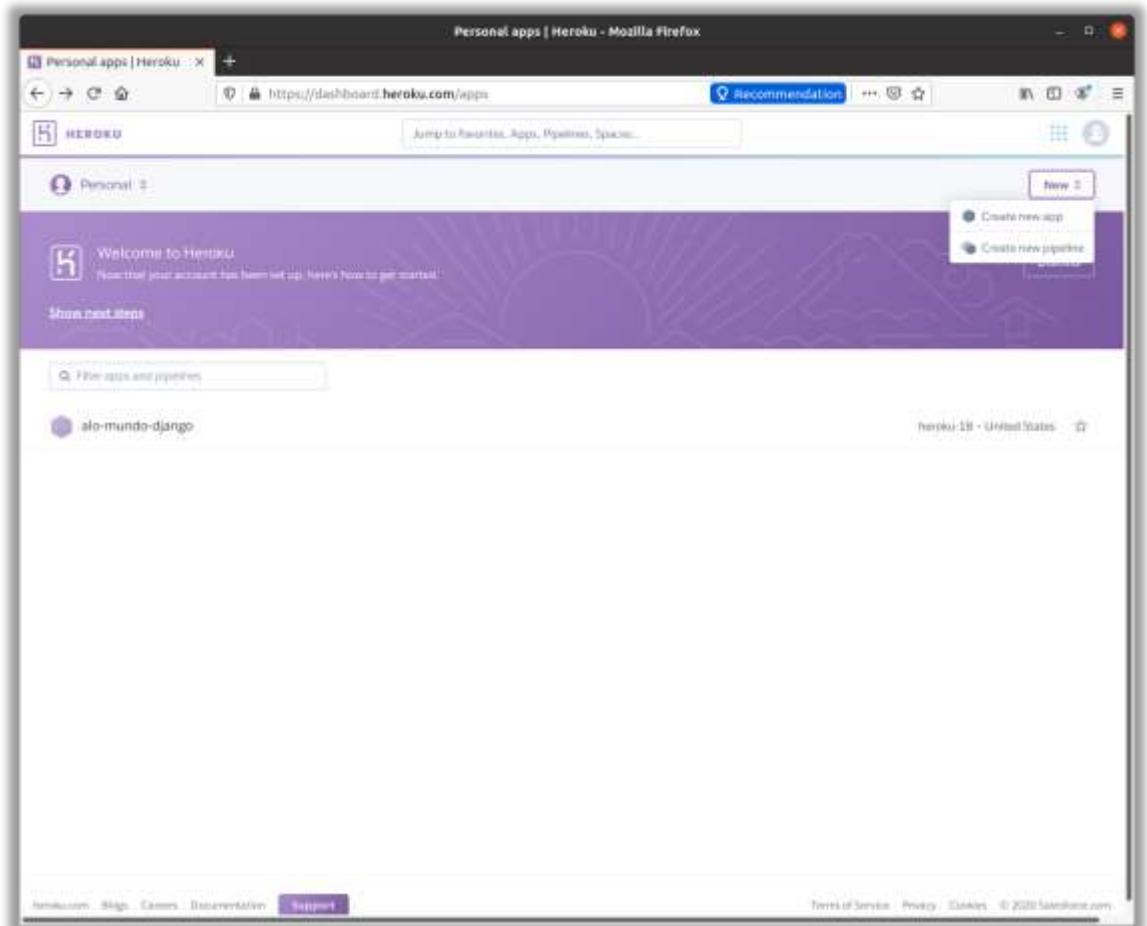
Deployment com Heroku

- Crie uma conta em www.heroku.com
- Existem milhões de outros sites para deployment
 - Pagos
 - Gratuitos
- Faça login



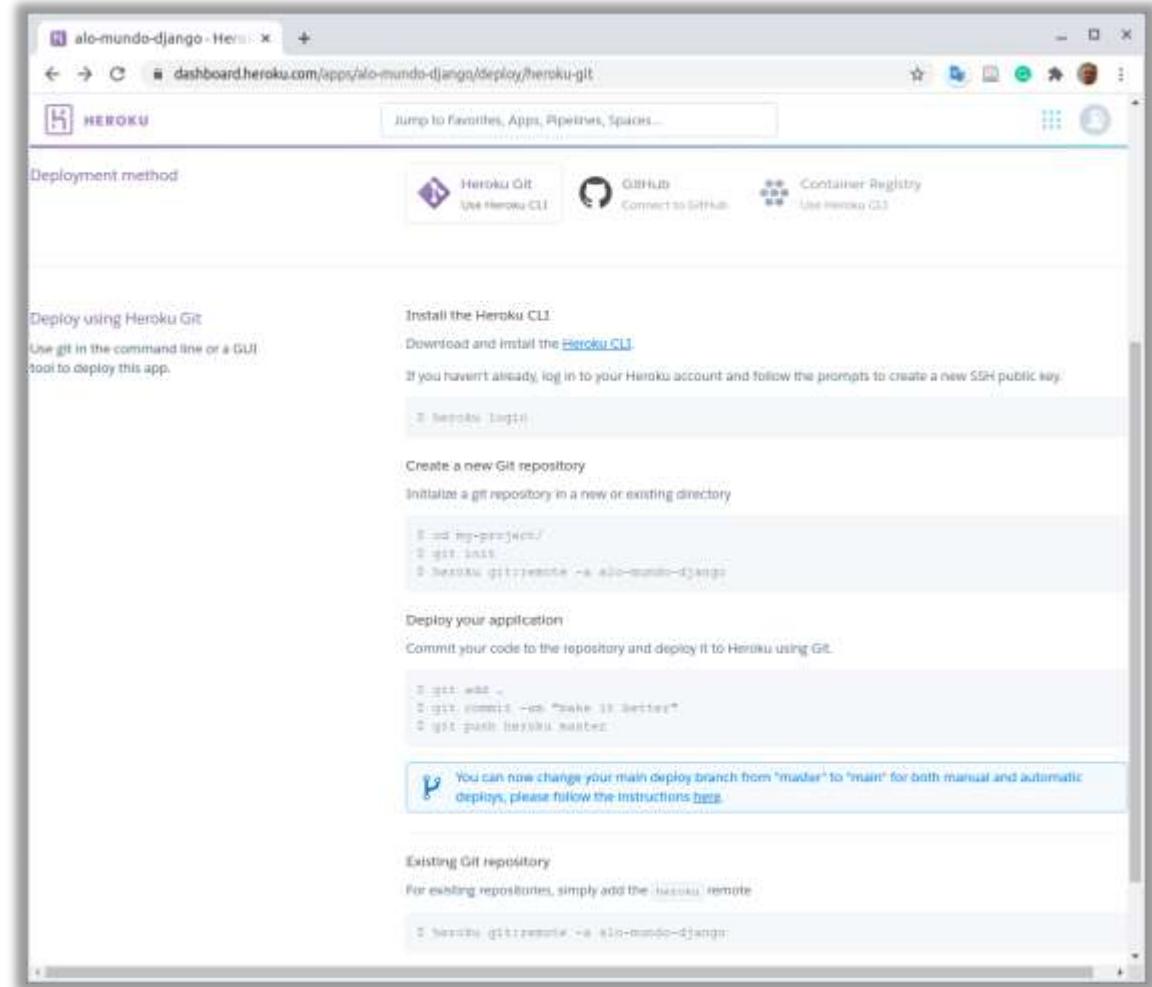
Deployment com Heroku

- Clique em New → Create new app



Deployment com Heroku

■ Siga as instruções



Deployment com Heroku

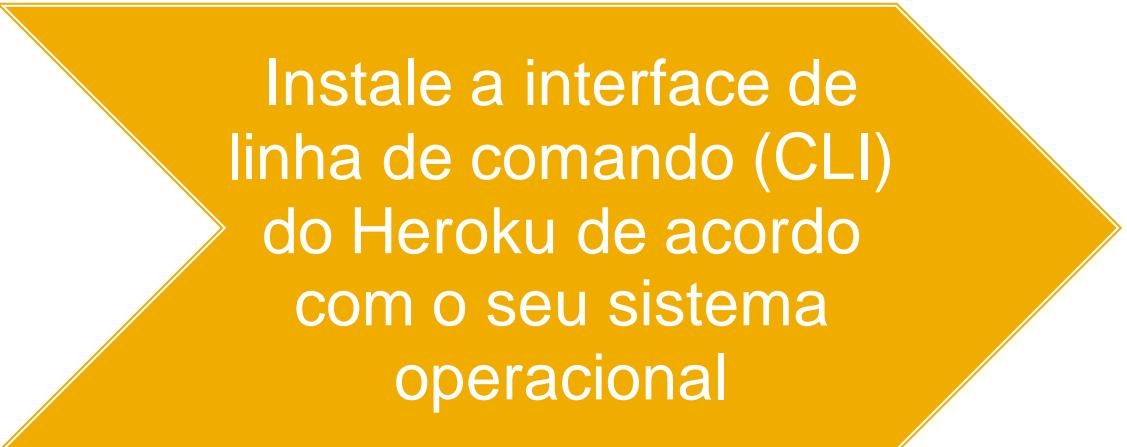
- Vá para o diretório do seu site
- (nesse ponto, você já deve ter clonado o site no GitHub)
- Faça o deployment da aplicação
- Dúvidas:
<https://devcenter.heroku.com/articles/getting-started-with-python>

```
$ heroku git:remote -a alo-mundo-django
heroku: Press any key to open up the
browser to login or q to exit:
Opening browser to https://cli-
auth.heroku.com/auth/cli/browser/15787b47-
1665-454e-9a76-95ef6b4e2ae0
Logging in... done
set git remote heroku to
https://git.heroku.com/alo-mundo-
django.git
```

Fazendo deployment

Usando o aplicativo do Heroku

Deployment com Heroku



Instale a interface de linha de comando (CLI) do Heroku de acordo com o seu sistema operacional

macOS

- `$ brew install heroku/brew/heroku`

Windows

- <https://cli-assets.heroku.com/heroku-x64.exe>

Ubuntu

- `$ sudo snap install heroku --classic`

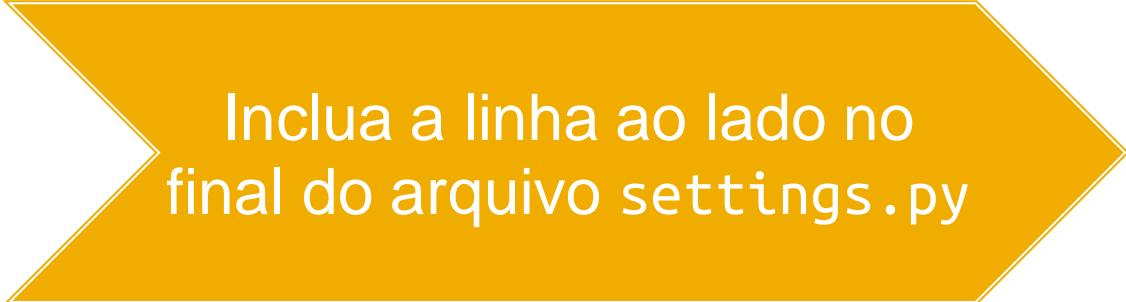
Deployment com Heroku

Crie um arquivo chamado requirements.txt na raiz do seu projeto

requirements.txt

```
# requirements.txt  
  
Django==3.1.3  
gunicorn  
django-heroku
```

Deployment com Heroku



Inclua a linha ao lado no final do arquivo `settings.py`

<Sua Aplicação>/settings.py

```
# Configure Django App for Heroku.  
try:  
    import django_heroku  
    django_heroku.settings(locals())  
except:  
    pass
```

Deployment com Heroku

- Para realizar a migração do banco de dados Django ao realizar o deployment, crie o arquivo Procfile na raiz do seu projeto com o seguinte conteúdo

Procfile

```
release: python manage.py migrate  
web: gunicorn <Sua Aplicação>.wsgi --log-file -
```

Deployment com Heroku

- Informe a versão do Python criando o arquivo runtime.txt na raiz do seu projeto com o seguinte conteúdo

```
runtime.txt  
python-3.7.9
```

Deployment com Heroku

- Antes, instale e configure o Git
- Faça login no Heroku pela linha de comando

```
$ heroku login
```

```
> Warning: Our terms of service have changed:  
> https://dashboard.heroku.com/terms-of-service  
heroku: Press any key to open up the browser to  
login or q to exit:  
Opening browser to https://cli-  
auth.heroku.com/auth/cli/browser/52ef2189-e1b7-  
40a6-957f-  
513ae1c217e2?requestor=SFMyNTY.g3QAAAACZAAEZGF0YW0  
AAAANMTg5LjYwLjE1My44MmQABnNpZ25lZG4GAKXx0AF2AQ.Mr  
QZQewmnURjXn-MONwyHCFI8rV3zvNQvrBlqctOYM  
Logging in... done  
Logged in as meslin@example.com
```

Deployment com Heroku

- Crie um app no Heroku:

‣ Name must start with a letter, end with a letter or digit and can only
‣ contain lowercase letters, digits, and dashes.

```
$ heroku create meslin-meusite  
Creating → meslin-meusite... done  
https://meslin-meusite.herokuapp.com/ |  
https://git.heroku.com/meslin-meusite.git
```

ANTES, não se esqueça de criar o repositório!

```
$ git init  
$ git config --local user.email "meslin@example.com"  
$ git config --local user.name "usuário"  
$ git add --all  
$ git commit -m "initial commit"
```

Deployment com Heroku

- Desabilite a coleta de diretórios estáticos pelo Heroku:

```
$ heroku config:set DISABLE_COLLECTSTATIC=1
```

- Faça o deployment do seu site:

```
$ git push heroku HEAD:master
```

- Crie pelo menos uma instância do site:

```
$ heroku ps:scale web=1
```

- Abra o site...

```
$ heroku open
```

Perguntas?



Problemas conhecidos e suas soluções

Problemas conhecidos e suas "soluções"

- Sintoma
 - Um ou mais blocos injetados de uma página não são mostrados
- Problema
 - Provavelmente os nomes dos blocos devem estar com algum problema de digitação
 - {`% block nomeDeUmTipo %`} {`% block nomeDeOutroTipo %`}
- Solução
 - Verifique os nomes dos blocos na página base e na base que realiza a injeção
 - {`% block nomeDoMesmoTipo %`} {`% block nomeDoMesmoTipo %`}

Problemas conhecidos e suas "soluções"

■ Sintoma:

- Erro de sintaxe ao carregar um template HTML com referência a algum arquivo estático

```
TemplateSyntaxError at /contatos/formulario/
Invalid block tag on line 5: 'static'. Did you forget to register
or load this tag?
```

■ Problema:

- A tag para criar links estáticos não foi registrada

■ Solução:

- Inclua no início do arquivo HTML a tag `{% load static %}`

TemplateSyntaxError at /contatos/formulario/

Invalid block tag on line 5: 'static'. Did you forget to register or load this tag?

Request Method: GET

Request URL: http://localhost:8000/contatos/formulario/

Django Version: 2.2.4

Exception Type: TemplateSyntaxError

Exception Value: Invalid block tag on line 5: 'static'. Did you forget to register or load this tag?

Exception Location: /usr/local/lib/python3.5/dist-packages/django/template/base.py in invalid_block_tag, line 534

Python Executable: /usr/bin/python3

Python Version: 3.5.2

Python Path: ['/media/meslin/643CA9553CA92352/Users/meslin/Google 'Drive/workspace-python-web-ubuntu/ExemplosWebPython', '/media/meslin/643CA9553CA92352/Users/meslin/Google 'Drive/workspace-python-web-ubuntu/ExemplosWebPython', '/usr/lib/python3.5', '/usr/lib/python3.5/plat-x86_64-linux-gnu', '/usr/lib/python3.5/lib-dynload', '/home/meslin/.local/lib/python3.5/site-packages', '/usr/local/lib/python3.5/dist-packages', '/usr/lib/python3/dist-packages', '/usr/lib/python35.zip']

Server time: Sun, 29 Sep 2019 11:52:59 -0300

Problemas conhecidos e suas "soluções"

- Sintoma:
 - Mensagem de erro:

TypeError: 'module' object is not iterable

...

django.core.exceptions.ImproperlyConfigured: The included URLconf '*<Seu Projeto>.urls*' does not appear to have any patterns in it. If you see valid patterns in the file then the issue is probably caused by a circular import.
- Problema:
 - Algum arquivo na sua aplicação pode estar errado (isso é óbvio!).
- Solução:
 - Verifique os seguintes arquivos:
 - forms.py
 - views.py

Problemas conhecidos e suas "soluções"

■ Sintoma

- Mensagem de erro:

`django.template.exceptions.TemplateSyntaxError: Could not
parse the remainder: ''<sua aplicação>/página.html' from
'<sua aplicação>/página.html'`

■ Problema:

- Algum erro de digitação.

■ Solução:

- Verificar se tem erro de digitação na tag
`{% extends 'página.html' %}`

Problemas conhecidos e suas "soluções"

- Sintoma:
 - Template não encontrado mesmo tendo sido criado no lugar certo
- Problema:
 - Ou o endereço está errado (faltou <aplicação>/<seu template>.html)
 - Ou a aplicação não está registrada em INSTALLED_APPS em settings.py
- Solução:
 - Acertar o endereço ou fazer o que está faltando

```
TemplateDoesNotExist at /sessao/contador/
sessao/contador.html

Request Method: GET
Request URL: http://localhost:8001/sessao/contador/
Django Version: 2.2.4
Exception Type: TemplateDoesNotExist
Exception Value: sessao/contador.html
Exception Location: /usr/local/lib/python3.5/dist-packages/django/template/loader.py in get_template, line 19
Python Executable: /usr/bin/python3
Python Version: 3.5.2
Python Path: ['/media/meslin/643CA9553CA92352/Users/meslin/Google',
'Drive/workspace-python-web-ubuntu/ExemplosWebTeste',
'/media/meslin/643CA9553CA92352/Users/meslin/Google',
'Drive/workspace-python-web-ubuntu/ExemplosWebTeste',
'/usr/lib/python3.5',
'/usr/lib/python3.5/plat-x86_64-linux-gnu',
'/usr/lib/python3.5/lib-dynload',
'/home/meslin/.local/lib/python3.5/site-packages',
'/usr/local/lib/python3.5/dist-packages',
'/usr/lib/python3/dist-packages',
'/usr/lib/python35.zip']
Server time: Qui, 10 Out 2019 09:56:58 -0300
```

Problemas conhecidos e suas "soluções"

- Sintoma:
 - TemplateDoesNotExist at <AlgumLugar>
- Problema:
 - O template existe dentro do diretório templates
- Solução:
 - Verifique se a aplicação está registrada em INSTALLED_APPS dentro de settings.py

```
TemplateDoesNotExist at /accounts/login/
chatsec/base.html

Request Method: GET
Request URL: http://localhost:8000/accounts/login/
Django Version: 2.2.4
Exception Type: TemplateDoesNotExist
Exception Value: chatsec/base.html
Exception Location: /usr/local/lib/python3.5/dist-packages/django/template/backends/django.py in reraise, line 84
Python Executable: /usr/bin/python3
Python Version: 3.5.2
Python Path: ['/media/meslin/643CA9553CA92352/Users/meslin/Google '
'Drive/workspace-python-web-ubuntu/mysite',
'/media/meslin/643CA9553CA92352/Users/meslin/Google '
'Drive/workspace-python-web-ubuntu/mysite',
'/usr/lib/python3.5',
'/usr/lib/python3.5/plat-x86_64-linux-gnu',
'/usr/lib/python3.5/lib-dynload',
'/home/meslin/.local/lib/python3.5/site-packages',
'/usr/local/lib/python3.5/dist-packages',
'/usr/lib/python3/dist-packages',
'/usr/lib/python35.zip']
Server time: Tue, 22 Oct 2019 14:32:37 -0300
```

Problemas conhecidos e suas "soluções"

■ Sintoma:

NoReverseMatch at /

'aplicação' is not a registered namespace

- Impossível carregar qualquer página que tenha referência a links em determinado namespace

■ Problema:

- O namespace pode não ter sido registrado
- O namespace foi escrito errado

■ Solução:

- Registrar o namespace no arquivo <Aplicação>/urls.py com o comando:

app_name = '*formularios*'

NoReverseMatch at /

'formularios' is not a registered namespace

Request Method: GET

Request URL: http://localhost:8000/

Django Version: 2.2.4

Exception Type: NoReverseMatch

Exception Value: 'formularios' is not a registered namespace

Exception Location: /usr/local/lib/python3.5/dist-packages/django/urls/base.py in reverse, line 86

Python Executable: /usr/bin/python3

Python Version: 3.5.2

Python Path: ['/media/meslin/643CA9553CA92352/Users/meslin/Google ' Drive/workspace-python-web-ubuntu/ExemplosWeb', '/media/meslin/643CA9553CA92352/Users/meslin/Google ' Drive/workspace-python-web-ubuntu/ExemplosWeb', '/usr/lib/python3.5', '/usr/lib/python3.5/plat-x86_64-linux-gnu', '/usr/lib/python3.5/lib-dynload', '/home/meslin/.local/lib/python3.5/site-packages', '/usr/local/lib/python3.5/dist-packages', '/usr/lib/python3/dist-packages', '/usr/lib/python35.zip']

Server time: Tue, 5 Nov 2019 14:13:38 -0300

Problemas conhecidos e suas "soluções"

■ Sintoma:

NoReverseMatch at /accounts/

Reverse for 'user-update' with keyword arguments
'{"pk": 2}' not found. 1 pattern(s) tried:
['accounts/UserUpdateView/\$']

- Impossível carregar qualquer página que tenha referência a links um determinado link

■ Problema:

- Está faltando algum parâmetro no link no template ou em urls.py

■ Solução:

- Verificar se o link foi criado corretamente em urls.py:
`<parametro:valor>`
- Verificar se o link foi referenciado corretamente no template:
`{% url 'nome-link' parametro=valor %}`

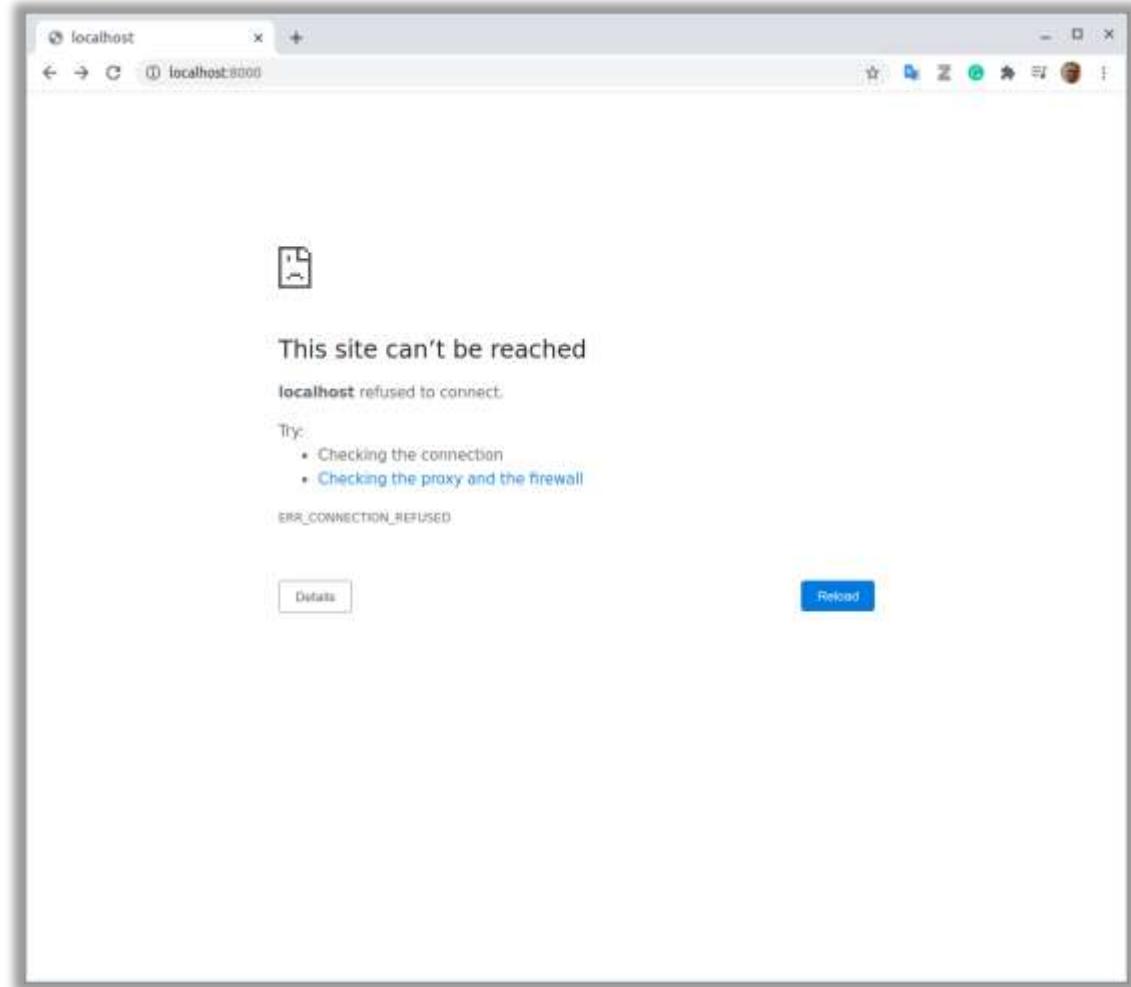
NoReverseMatch at /accounts/

Reverse for 'user-update' with keyword arguments '{'pk': 2}' not found. 1 pattern(s) tried: ['accounts/UserUpdateView/\$']

```
Request Method: GET
Request URL: http://127.0.0.1:8000/accounts/
Django Version: 3.1.3
Exception Type: NoReverseMatch
Exception Value: Reverse for 'user-update' with keyword arguments '{'pk': 2}' not found. 1 pattern(s) tried: ['accounts/UserUpdateView/$']
Exception Location: /usr/local/lib/python3.8/dist-packages/django/urls/resolvers.py, line 685, in _reverse_with_prefix
Python Execution: /usr/bin/python3
Python Version: 3.8.10
Python Path: ['/media/meslin/4E7E1B07E11EE1/vers/meslin/Documents/Cursos/Web-Python/sites/Exemplos_2022_2',
             '/usr/lib/python3.8',
             '/usr/lib/python3.8/lib-dynload',
             '/home/meslin/.local/lib/python3.8/site-packages',
             '/usr/local/lib/python3.8/dist-packages',
             '/usr/local/lib/python3.8/dist-packages/dlib-19.16.0-py3.8-linux-x86_64.egg',
             '/usr/local/lib/python3.8/dist-packages/pyserial-3.5-py3.8.egg',
             '/usr/lib/python3/dist-packages']
Server time: Fri, 04 Nov 2022 17:30:43 +0000
```

Problemas conhecidos e suas "soluções"

- Sintoma:
 - O site não pode ser encontrado
- Problema:
 1. O endereço está errado
 2. A porta está errada
 3. O site está fora do ar
- Solução:
 1. Corrija o endereço
 2. Acerte ou informe o número da porta
 3. Use o comando 'python3 manage.py runserver' para colocar o site no ar



Problemas conhecidos e suas "soluções"

■ Sintoma:

- Erro ao tentar criar superuser (ou outro usuário qualquer ou até mesmo uma entrada ou tabela no BD)

■ Problema:

- A base de dados pode estar precisando ter migração aplicada

■ Solução:

- Aplique a migração com o comando 'python3 manage.py migrate'

```
$ python3 manage.py createsuperuser
```

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

Traceback (most recent call last):

```
File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 84, in _execute
    return self.cursor.execute(query, params)
File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/sqlite3/base.py", line 413, in execute
    return Database.Cursor.execute(self, query, params)
sqlite3.OperationalError: no such table: auth_user

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "/home/meslin/.local/lib/python3.8/site-packages/django/core/management/commands/createsuperuser.py", line 84, in execute_from_command_line
    utility.execute()
  File "/home/meslin/.local/lib/python3.8/site-packages/django/core/management/base.py", line 395, in execute
    self.fetch_command(subcommand).run_from_argv(self.argv)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/core/management/base.py", line 330, in run_from_argv
    self.fetch_command(argv[0]).run_from_argv(argv)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/contrib/auth/management/commands/createsuperuser.py", line 79, in execute
    return super().execute(*args, **options)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/contrib/auth/management/commands/createsuperuser.py", line 371, in execute
    output = self.handle(*args, **options)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/contrib/auth/management/commands/createsuperuser.py", line 100, in handle
    default_username = get_default_username()
  File "/home/meslin/.local/lib/python3.8/site-packages/django/contrib/auth/management/_init_.py", line 140, in get_default_username
    auth_user._default_manager.get(username=default_username)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/models/manager.py", line 85, in manager_method
    return getattr(self._db, name)(*args, **kwargs)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/models/query.py", line 425, in get
    return next(c)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/models/query.py", line 269, in __len__
    self._fetch_all()
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/models/query.py", line 1303, in _fetch_all
    self._result_cache = list(self.iterator())
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/models/query.py", line 53, in __iter__
    results = connection.cursor().executesql(chunked_fetchsql, fetch, chunk_size=self.chunk_size)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/sql/compiler.py", line 1154, in execute_sql
    cursor.execute(sql, params)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 98, in execute
    return self.cursor.execute(sql, params)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 66, in execute
    return self._execute_with_wrappers(sql, params, many=False, executor=self._execute)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 75, in _execute_with_wrappers
    return executor(sql, params, many, context)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 84, in _execute
    self._check_for_backend_error(cursor, sql, params, context)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/utils.py", line 90, in __exit__
    raise dj_exc_value.with_traceback(traceback) from exc_value
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/utils.py", line 84, in _execute
    return self.cursor.execute(sql, params)
  File "/home/meslin/.local/lib/python3.8/site-packages/django/db/backends/sqlite3/base.py", line 413, in execute
    return Database.Cursor.execute(self, query, params)
sqlite3.OperationalError: no such table: auth_user
```

Problemas conhecidos e suas "soluções"

- Sintoma:
 - Esqueceu a senha do superusuário
- Problema:
 - (não vou nem comentar...)
- Solução:
 - Dentro do diretório raiz do projeto, use o comando 'python3 manage.py changepassword' para trocar a senha

```
$ python3 manage.py changepassword
Changing password for user 'meslin'
Password:
Password (again):
```

Problemas conhecidos e suas "soluções"

- Sintoma:
 - Erro ao carregar a página por falta de retorno adequado do view

The view `noticias.views.home` didn't return an `HttpResponse` object. It returned `None` instead.
- Problema:
 - O view não retornou um objeto da classe `HttpResponse`
- Solução:
 - Faltou `return <HttpResponse>`
 - Usou render sem `return`
 - Usou `HttpResponse` sem `return`

ValueError at /noticias/

The view `noticias.views.home` didn't return an `HttpResponse` object. It returned `None` instead.

```
Request Method: GET
Request URL: http://localhost:8000/noticias/
Django Version: 3.1
Exception Type: ValueError
Exception Value: The view noticias.views.home didn't return an HttpResponse object. It returned None instead.
Exception Location: /home/meslin/local/lib/python3.8/site-packages/django/core/handlers/base.py, line 307, in check_response
Python Executable: /usr/bin/python3
Python Version: 3.8.2
Python Path: ['/media/meslin/DA32492932490BC7/Users/meslin/Google Drive/Cursos/Web-Python/sites/CMS_Noticias', '/usr/lib/python38.zip', '/usr/lib/python3.8', '/usr/lib/python3.8/lib-dynload', '/home/meslin/.local/lib/python3.8/site-packages', '/usr/local/lib/python3.8/dist-packages', '/usr/lib/python3/dist-packages']
Server time: Fri, 07 Aug 2020 20:33:29 +0000
```

Problemas conhecidos e suas "soluções"

■ Sintoma:

- As imagens não são localizadas por URLs

■ Problema:

- Não configurou corretamente a variável STATIC_URL e/ou STATICFILES_DIRS em settings.py

■ Solução:

- Configurar corretamente as variáveis

```
STATIC_URL = '/static/'
```

```
STATICFILES_DIRS = [
    os.path.join(BASE_DIR, 'static'),
    os.path.join(BASE_DIR,
    'MeuSite/static'),
]
```



Problemas conhecidos e suas "soluções"

■ Sintoma:

Not Found: /favicon.ico
[22/Oct/2020 21:05:17] "GET /favicon.ico HTTP/1.1" 404 2220

■ Problema:

- O arquivo favicon.ico (16x16 pixels) não foi encontrado.
- Crie um favicon.ico – existem vários editores online gratuitos.

■ Solução:

- Crie um arquivo com o ícone do seu site na resolução 16x16 pixels no formato de ícone (.ico)
 - Salve o arquivo em um diretório estático
 - Configure os diretórios estáticos
 - Em urls.py, redirecione o pedido para o arquivo
- ```
from django.views.generic import RedirectView
urlpatterns = [..., path('favicon.ico', RedirectView.as_view(url='/static/favicon.ico')), ...]
```

# Problemas conhecidos e suas "soluções"

## ■ Sintoma:

- Erro 500 ao tentar carregar uma página
- Mensagem
  - `ModuleNotFoundError`

## ■ Problema:

- Algum módulo não foi encontrado

## ■ Solução:

- Verificar se existe algum erro de digitação de nome de módulo em algum comando `import`
- ou em algum comando ou variável no arquivo `settings.py`
- Verificar o módulo não encontrado realmente existe ou não foi instalado

ModuleNotFoundError at /accounts/password\_reset/

No module named 'django.core.email'

Request Method: POST  
Request URL: http://127.0.0.1:8000/accounts/password\_reset/  
Django Version: 3.1.3  
Exception Type: `ModuleNotFoundError`  
Exception Value: No module named 'django.core.email'  
Exception Location: <frozen importlib.\_bootstrap>, line 973, in \_find\_and\_load\_unlocked  
Python Executable: /usr/bin/python3  
Python Version: 3.8.10  
Python Path: ['/media/meslin/4E7E313D7E311EE1/Users/meslin/Google 'Drive/Cursos/Web-Python/Aulas/2021-2/Sites/Exemplos', '/usr/lib/python38.zip', '/usr/lib/python3.8', '/usr/lib/python3.8/lib-dynload', '/home/meslin/.local/lib/python3.8/site-packages', '/usr/local/lib/python3.8/dist-packages', '/usr/local/lib/python3.8/dist-packages/dlib-19.16.0-py3.8-linux-x86\_64.egg', '/usr/lib/python3/dist-packages']  
Server time: Mon, 08 Nov 2021 21:57:31 +0000

# Problemas conhecidos e suas "soluções"

- Sintoma:
  - Data aparece no formulário com formato errado
- Problema:
  - Alguma configuração faltando ou errada
- Solução:
  - Verifique se LANGUAGE\_CODE está correto em settings.py
  - Verifique se dtNasc = forms.DateField(input\_formats=['%d/%m/%Y']) está correto em forms.py e coerente com LANGUAGE\_CODE

# Problemas conhecidos e suas "soluções"

- Sintoma:
  - Não consegue fazer login com o superusuário
- Problema:
  - Esqueceu a senha do superusuário
- Solução:
  - Para redefinir a senha de um usuário no Django, você pode seguir as etapas abaixo:
    - 1. Acesse o shell interativo do Django executando o seguinte comando no terminal:`  
\$ python3 manage.py shell
    - 2. No shell interativo do Django, importe o modelo `User`:  
In [1]: from django.contrib.auth.models import User
    - 3. Encontre o usuário para o qual você deseja redefinir a senha. Você pode pesquisar pelo nome de usuário ou pelo endereço de e-mail, por exemplo:  
In [2]: user = User.objects.get(username='usuário')
    - 4. Em seguida, você pode usar o método `set\_password()` para definir a nova senha para o usuário:  
In [3]: user.set\_password('nova\_senha')  
In [4]: user.save()  
In [5]: exit
    - Substitua `'nova\_senha'` pela senha desejada.

# Problemas conhecidos e suas "soluções"

- Sintoma:
  - Django acusa a ausência de `api.html`, mesmo você não fazendo referência a ela
- Problema:
  - A API RestFramework não foi instalada corretamente
- Solução:
  - Instale a api `rest_framework` com o seguinte comando:  
`$ pip install rest_framework`
  - Configure a API como um dos `INSTALLED_APPS` em `settings.py`:  
`'rest_framework',`

```
TemplateDoesNotExist at /carros/lista/
rest_framework/api.html

Request Method: GET
Request URL: http://127.0.0.1:8000/carros/lista/
Django Version: 4.1.6
Exception Type: TemplateDoesNotExist
Exception Value: rest_framework/api.html
Exception Location: /media/meslin/AE8A7E628A7E274D/Users/meslin/Documents/Cursos/Web-
Python/sites/Teste_2023_2/venv/lib/python3.10/site-packages/django/template/loader.py, line 19, in
get_template
Raised during: carros.views.CarView
Python Executable: /media/meslin/AE8A7E628A7E274D/Users/meslin/Documents/Cursos/Cursos/Web-
Python/sites/Teste_2023_2/venv/bin/python3
Python Version: 3.10.6
Python Path: ['/media/meslin/AE8A7E628A7E274D/Users/meslin/Documents/Cursos/Web-Python/sites/Teste_2023_2',
 '/usr/lib/python310.zip',
 '/usr/lib/python3.10',
 '/usr/lib/python3.10/lib-dynload',
 '/media/meslin/AE8A7E628A7E274D/Users/meslin/Documents/Cursos/Cursos/Web-
Python/sites/Teste_2023_2/venv/lib/python3.10/site-packages']
Server time: Tue, 30 May 2023 01:32:44 +0000
```

# Problemas conhecidos e suas "soluções"

- Sintoma:

- Falta o método sort em FormData

```
type FormData is not assignable to type URLSearchParams
because the property 'sort' is missing in type 'FormData'
but required in type 'URLSearchParams'.
```

- Problema:

- O método sort está faltando em lib.dom.ts
  - <https://youtrack.jetbrains.com/issue/WEB-50119>

- Solução:

- Sem solução (não use)

# Problemas conhecidos e suas "soluções"

- Sintoma:  
AttributeError: type object 'Token' has no attribute 'objects'
- Problema:
  - O Django não conhece o objeto Token
- Solução:
  - Importe o objeto Token de `rest_framework.authtoken.models`  
`from rest_framework.authtoken.models import Token`
  - Registre a aplicação `rest_framework.authtoken` em `INSTALLED_APPS` no arquivo `settings.py`





Alexandre Meslin  
meslin@puc-rio.br

# Programação para a Web em Python

