




Lógica de Programação

JavaScript





 profalex.deus@fiap.com.br

Alex Sander Resende de Deus

Professor

Há 25 anos ensinando programação, é um apaixonado por tecnologia. Atualmente é coordenador de cursos na ETEC Albert Einstein do Centro Paula Souza.

É professor da FIAP School e FIAP Corporate, lecionando C#, Banco de Dados e Desenvolvimento Mobile.

AGENDA

1

AULA 1

Definição e histórico

Fundamentos Básicos da Programação – Tipos de dados, variáveis e estruturas básicas; Usando o GIT

2

AULA 2

Estruturas condicionais, operadores lógicos e switch case

3

AULA 3

Laços de Repetição: For, while,

4

AULA 4

Objetos nativos JavaScript – Date, String, Math

AGENDA

5

AULA 6

Arrays, Filter, Map

Reduce, Splite, forEach, for in, for of

6

AULA 6

Funções: tipos, declarações, escopo, retorno, clousers,
call-back, IIFE, factory

7

AULA 7

DOM – Document Object Model

8

AULA 8

DOM – Document Object Model

AGENDA

9

AULA 9

- Eventos
- Objetos: Criação, prototype, getters, setters

10

AULA 10

Classes JavaScript Assíncrono

AULA 4

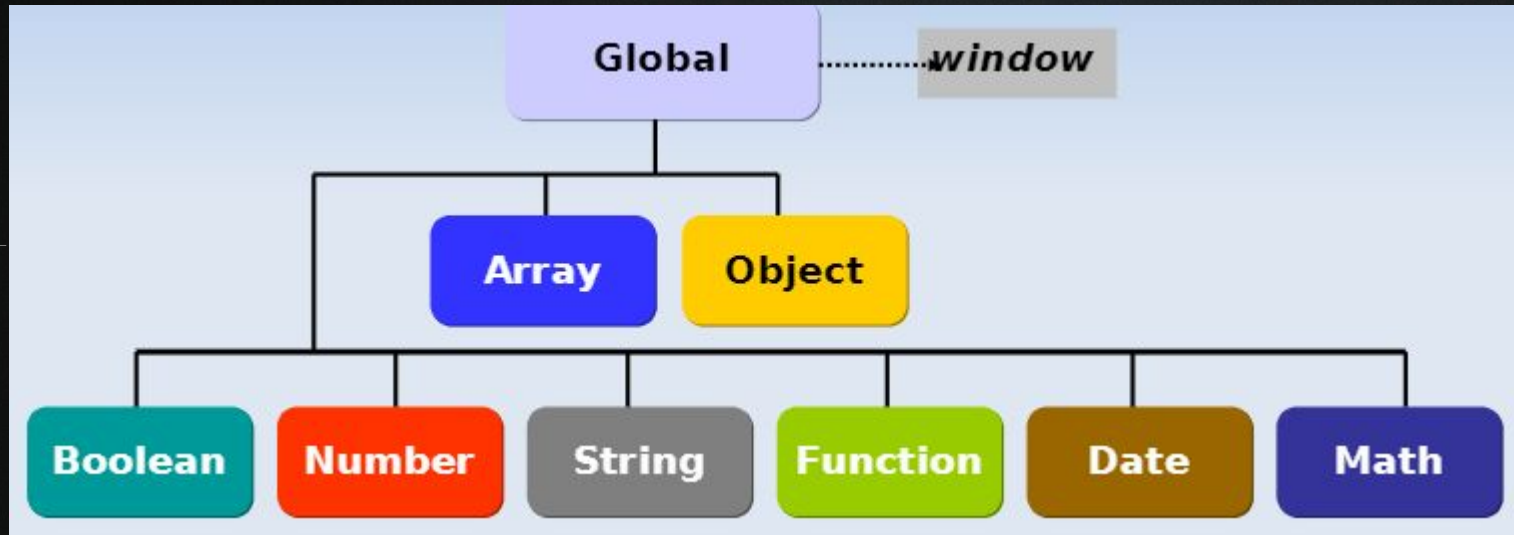
Objetos Nativos JavaScript

Objetos nativos da linguagem JavaScript são criados e mantidos pelo browser para acessar os elementos do documento.

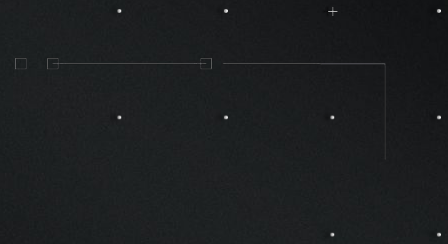
Fazem parte do núcleo da Linguagem. Não são criados automaticamente, com exceção: Global e Math. Nem todos os objetos nativos tem construtores.

Englobam praticamente tudo o que a linguagem nos oferece.

Qualquer elemento se torna um objeto em JavaScript, desde uma string primitiva (que é convertida em objeto quando necessário) até um array.



Tipo Object



Tipo genérico de objeto usado para representar qualquer objeto criado com new.

Métodos

toString(): transforma qualquer objeto em uma representação

stringValueOf(): converte qualquer objeto em seu valor

primitivo

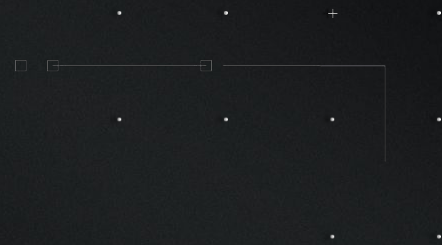
Exemplo

```
js > JS script.js
```

```
1  d=new Date();  
2  alert(d.toString());  
3  
4  n=new Number(1000);  
5  alert(n.valueOf());
```

```
1  <!DOCTYPE html>  
2  
3  <html lang="pt-br">  
4  |   <head>  
5  | |   <title>Objetos Nativos</title>  
6  | |   </head>  
7  |   <body>  
8  | |   <script type="text/javascript" src="js/script.js"></script>  
9  |   </body>  
10 </html>
```


Tipo Number



Usado para representar números como objetos.

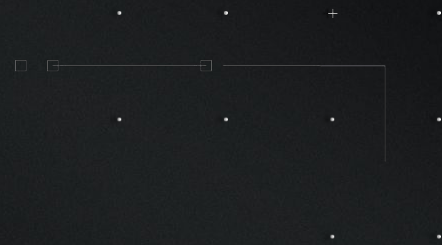
A principal utilidade é disponibilizar algumas constantes

globais:

Função	Valor
Number.MAX_VALUE	1.79e+308
Number.MIN_VALUE	5e-324
Number.NaN	NaN
Number.POSITIVE_INFINITY	-Infinity
Number.NEGATIVE_INFINITY	Infinity



Funções do Objeto **Number**



toFixed()

```
1  var n = new Number ("80.90674");
2  /*o método toFixed retorna o número de casas
3  decimais definidas dentro do parâmetro*/
4  alert ( n.toFixed());
5  /*com os parênteses vazios, ele retorna um número
6  sem casas decimais.
7  Resultado: 81 /*aqui ele arredondou para 81*/
8
9
10 var n = new Number ("80.90674");
11 /*o método toFixed retorna o número de casas decimais definidas dentro do parâmetro.*/
12 alert ( n.toFixed(3));
13 /*Para obter-se a quantidade de casas decimais desejadas,
14 declarar isso dentro do parênteses.
15 Resultado: 80.907*/
```

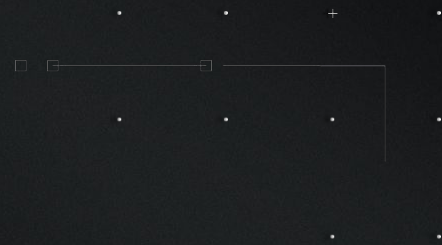

toFixed()

```
1  var n = new Number ("54.47849");
2  alert (n.toFixed());
3  /*este parâmetro é opcional se vazio retornará o número informado
4  preenchido retornará o número com a quantidade de dígitos definida no parâmetro
5  Resultado: 54.47849*/
6
7  var n = new Number ("54.47849");
8  alert (n.toFixed(1));
9  /*retorna o número com a quantidade de dígitos definida no parâmetro
10 Resultado: 5e+1
11 /*aqui ele retornou em notação científica o que cabe dentro de 1*/
12
13 var n = new Number ("54.47849");
14 alert (n.toFixed(5));
15 /*retorna o número com a quantidade de dígitos definida no parâmetro
16 Resultado: 54.478
17 /*aqui ele retornou um número igual ao informado, com 5 dígitos
18 Obs: com precisão menor que 3, o número será convertido para notação científica.*/
```

toExponential()

```
1  /*0 parâmetro nessa função é opcional. Se for passado algum parâmetro,
2  deve estar compreendido entre 0 e 20*/
3
4  var n = new Number ("54.47849");
5  alert (n.toExponential());
6  /*aqui se não for passado nenhum parâmetro, o número será transformado
7  em notação científica, com a precisão necessária para representar o número todo.
8  Resultado: 5.447849e + 1*/
9
10 var n = new Number ("54.47849");
11 alert (n.toExponential(5));
12 /*se for declarado um parâmetro, nesse exemplo o 5, o número será
13 transformado em notação científica, com a precisão possível ao número declarado.
14 Resultado: 5.44785e + 1*/
```

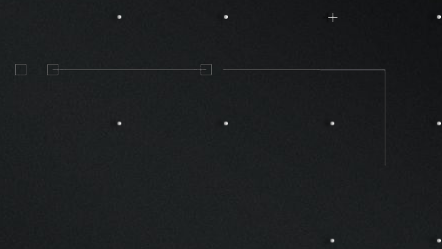
Tipo String



String é uma palavra, texto ou número, colocados dentro de aspas duplas, de preferência, ou simples.

Exemplo

```
1  var x = new String ("Criando uma String");  
2  alert (x);/*será apresentada em uma caixa de alerta  
3  Resultado: Criando uma String*/
```



Funções do Objeto String

length

```
1  ✓ /*length
2    retorna o tamanho da String.*/
3
4    var x = new String("Criando uma String");
5    alert( x.length );
6  ✓ /*Resultado: 18
7    conta todos os caracteres, inclusive os espaços*/
```



charAt()

```
1  var x = new String("Criando uma String");
2  alert( x.charAt(4) );
3  /*Resultado: n
4  esse método retorna a posição indicada no parâmetro e começa na posição 0.
5  Nesse caso como foi passado como parâmetro o número 4,
6  o resultado será n. (Na posição 4 da string fica a letra n.)
7  Se fosse na posição 7 por exemplo, a janela de alerta apareceria vazia,
8  representando o espaço em branco, pois ele também entra na contagem*/
```



charCodeAt()

```
1  var x = new String("Criando uma String");
2  alert( x.charCodeAt(x.length - 1) );
3  /*Resultado: 67
4  o método charCodeAt retorna da mesma forma que charAt,
5  só que no padrão Unicode.*/
```

Padrão Unicode

O que aparece como texto na tela está armazenado como **valores numéricos** no arquivo de texto. O computador traduz os valores numéricos em **caracteres visíveis**. Ele faz isso usando um **padrão de codificação**. Um padrão de codificação é um esquema numérico que atribui cada caractere de texto, em um conjunto de caracteres, a um valor numérico. Um conjunto de caracteres pode incluir caracteres alfa-numéricos, números e outros símbolos. **Idiomas diferentes consistem normalmente de conjuntos diferentes de caracteres**. Muitos padrões de codificação diferentes existem para representar os conjuntos de caracteres usados em idiomas diferentes.

concat()

```
1  var x = new String("Criando uma String");
2  alert( x.concat(" e concatenando com outra") );
3  /*Resultado: criando uma string e concatenando com outra
4  o método concat, concatena duas strings.*/
5  |
```



fromCharCode()

```
1  var x = new String("Criando uma String");
2  alert(String.fromCharCode(66));
3  /*Resultado: B
4  Método estático (acessado direto da classe String. Converte o
5  valor unicode para uma string. Ex: o valor unicode da letra a é.....*/
```



indexOf()

```
1  var x = new String("Criando uma String");
2  alert(x.indexOf ("uma"));
3
4  /*retorna a posição de uma determinada string.
5  Nesse exemplo uma, começa na posição 8
6  Resultado: 8*/
```

lastIndexOf()

```
1  var x = new String("String. Criando uma String");
2  alert(x.lastIndexOf ("String"));
3  /*Resultado: 20
4  retorna a última posição de uma determinada string.
5  Nesse exemplo string, começa na posição 20*/
```



match()

```
1  /*A função match é usada junto com Regex, que são as expressões regulares.*/
2  var x = new String("String. Criando uma String");
3  var re = /i/; /*sempre dentro de barras*/
4  alert(x.match(re) );
5  /*Resultado: i*/
6
7  var re2=/z/;
8  alert (x.match(re2)); /*Como a letra z não está presente na
9  string, o retorno é null*/
```



replace()

```
1  /*Substitui uma determinada string por outra em um texto.*/  
2  var x = new String("String. Criando uma String");  
3  alert(x.replace("String", "Teste") );  
4  /*faz a atualização na primeira ocorrência da palavra String  
5  Resultado: i*/
```



substring()

```
1  /*Recorta uma determinada string, especificada dentro do parâmetro.*/  
2  
3  var x = new String("String. Criando uma String");  
4  alert(x.substring( 7, 15) );  
5  /*alert(x.substring( 15, 7) );*/  
6  /*O número maior primeiro não afeta  
7  | Resultado: Criando*/
```



substr()

```
1  ✓ /*Também tem dois parâmetros e extrai os caracteres entre dois índices especificados.  
2  Informa o parâmetro do índice inicial, nesse caso o C, e mais 15 caracteres na frente.*/  
3  
4  var x = new String("String. Criando uma String");  
5  alert(x.substr( 7, 15) );  
6  /*Resultado: Criando uma St*/
```



substr()

```
1  /*É o mesmo que substring, só que mais rigoroso.*/  
2  var x = new String("String. Criando uma String");  
3  alert(x.substr( 7, 15) );  
4  alert(x.substr( 15, 7) );  
5  ✓ /*Se o número maior vier primeiro ele não dará retorno  
6  Resultado: sem retorno*/
```

split()

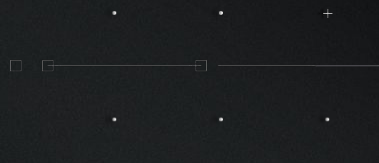
```
1  /*Faz o recorte com base em um separador.*/  
2  var x = new String("String, Criando uma String");  
3  alert(x.split(",") [0] );  
4  /*Resultado: Criando uma String*/
```



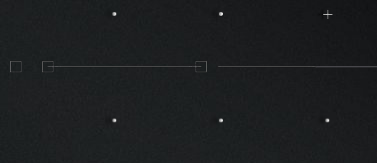
toUpperCase() e toLowerCase()

```
1  /*0 primeiro deixa o texto em letras maiúsculas e o
2  segundo todo em minúsculas.*/
3
4  var x = new String("String, Criando uma String");
5  alert(x.toUpperCase() );
6  alert(x.toLowerCase() );
7  |
8  /*Resultado: STRING. CRIANDO UMA STRING
9  Resultado: string. criando uma string*/
```

Tipo Date



O Objeto Date utiliza datas disponíveis no sistema operacional da máquina que está rodando o script. Ao solicitar para o JavaScript, data e hora atual, estas serão recuperadas do SO do computador do usuário. Se forem alteradas, isso terá consequências no script utilizado. Deve-se observar a diferença nos dois horários.



Lembrete: Existem funções no objeto Date que servem para trabalhar a hora local e a hora mundial.

Ahora é determinada pelo meridiano de Greenwich, que serve de referência para calcular distâncias em longitudes e estabelecer fusos horários. Cada fuso horário corresponde a uma faixa de quinze graus de longitude de largura, sendo a hora de Greenwich chamada de Greenwich Mean Time (GMT). Alguém que está no Brasil, não poderá se orientar, se uma vídeo conferência for marcada na hora local da Inglaterra, por exemplo. Mas se for feita, no horário mundial, os envolvidos, terão a mesma referência: Isto é resolvido, com as funções que têm UTC = Universal Time Coordinated.

UTC = GMT (Greenwich Mean Time).

Hora Brasil = Hora GMT - 3 Horas

Funções do Objeto **Date**

construtor

```
1  ✓ /*Sem parâmetros:
2    Retorna o dia e a hora atual*/
3
4    var data = new Date();
5    alert(data);
6    /*Retorna o dia e a hora atual*/
7
8  ✓ /*Com Strings
9    Informando strings dentro dos parênteses.*/
10
11   var data = new Date("Dec/03/1958 11:20:30");
12   alert(data);
13   /*Resultado: Wed Dec 03 1958 11:20:30 GMT-0200(Hora oficial do Brasil*/
14
15   /*Informando strings dentro dos parênteses.*/
16
17   var data = new Date( 1986, 4, 24, 8, 30);
18   /*começa o mês em zero*/
19   alert(data);
20   /*Resultado: Thu Apr 24 08:30:00 UTC-0300 1986*/
```

Hora Local e Universal

```
1  /*Diferença entre Hora Local e Hora Universal.*/
2  var data = new Date( );
3  alert(`Hora local ${data.getHours()}`);
4  alert(`Hora Universal ${data.getUTCHours()}`);
5
6  /*Formato americano*/
7  var data = new Date("Dec/22/2013 09:51:20");
8  alert(`Hora local ${data.getHours()}`);
9  alert(`Hora Universal ${data.getUTCHours()}`);
```

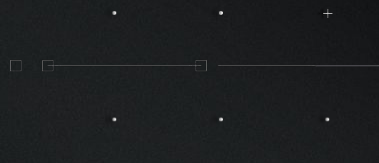

Método get()

```
1  /*Os métodos gets são para recuperar uma informação*/
2  var dias = ["Domingo", "segunda", "terça", "quarta", "quinta", "sexta", "sábado"];
3  var data = new Date(); /*construtor vazio, retorna a hora atual*/
4  alert(data.getHours()); /*retorna a hora*/
5  alert(data.getMilliseconds()); /*retorna os milisegundos dessa data*/
6  alert(data.getMinutes()); /*retorna os minutos*/
7  alert(dias[data.getDay()]); /*retorna o dia da semana*/
8  alert(data.getDate()); /*retorna o dia do mês*/
9  alert(data.getFullYear()); /*ano com quatro dígitos*/
```


Método set()

```
1  var dias = ["Domingo", "segunda", "terça", "quarta", "quinta", "sexta", "sábado"];
2
3  var mes = ["janeiro", "fevereiro", "março", "abril", "maio", "junho", "julho", "agosto",
4  "setembro", "outubro", "novembro", "dezembro"];
5
6  var data = new Date("Oct/17/2021 13:30:40");
7  data.setFullYear(2016);
8  data.setDate(04);
9  data.setMonth(11);
10
11  /*modifica o dia mês e ano passado no construtor*/
12  alert(data);
```

Tipo Math



O objeto Math auxilia na criação de scripts para realizar operações matemáticas. Este objeto não precisa de um construtor. Para os outros objetos cria-se um new, mas math tem as funções e as propriedades estáticas (acesso direto pela classe Math.) As propriedades do objeto são constantes

matemáticas notáveis

Funções do Objeto **Math**

```
1  alert(Math.PI); //valor de PI
2
3  alert(Math.abs(5.9)); //número absoluto
4
5  alert(Math.max(5.9, 6, 12, -80)); //menor número
6
7  alert(Math.round(12.6)); //inteiro mais próximo
8
9  alert(Math.floor(45.5)); //arredonda para baixo
10
11 alert(Math.ceil(45.5)); //arredonda para cima
12
13 alert(Math.pow(12,3)); //12 elevado a terceira potência
14
15 alert(Math.sqrt(16)); //raiz quadrada de 16
```

Números Aleatórios

Um gerador de números aleatórios é um dispositivo computacional ou físico que gera uma sequência de números ou símbolos sem qualquer padrão, (pseudo)aleatoriamente. Ele gera números entre 0 e 1. Em JavaScript esses números são gerados com o método `random()` do objeto `Math` que permite obter números (pseudo)aleatórios para diversos usos em scripts.

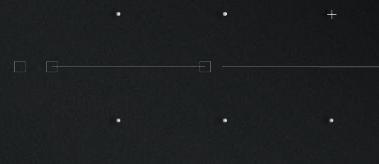
random()

```
1  alert(Math.random()); //o resultado é um número entre 0 e 1, com vários algarismos
2
3  alert(Math.random( ) * 6 ); //nesse caso o resultado é um número de 0 a 5, gerado com vários algarismos
4
5  /*Para gerar números decimais de 0 até 10 multiplica-se por 10, de 0 até 15,
6  multiplica-se por 15, e assim sucessivamente. Depende da quantidade de números desejada.*/
7
8  /*Para gerar número aleatórios inteiros usa-se a função math.floor,
9  que arredonda o número para baixo.*/
10 var n = Math.floor(Math.random()*10); //gera um número aleatório inteiro, entre 0 e 9
11 alert(n); //retorna um número entre 0 e 9
12
13 /*(lembre-se que o número gerado é sempre um número antes do número máximo declarado)*/
14 var n = Math.floor(Math.random()*10+1); //gera um número aleatório inteiro, entre 0 e 10
15 alert(n); //retorna um número entre 0 e 10
16 /*Note que +1 faz com que o número máximo também possa ser gerado,
17 sendo que aqui esse número foi o número 10*/
18
19 /*Gerando número aleatório dentro de um Limite*/
20 numAleat = Math.random() * 3 ; //quantidade de números possíveis
21 numAleat = Math.floor(numAleat); //arredonda o número. Floor arredonda para baixo
22 numLimiteInf = 5; //limite inferior
23 Aleat += LimiteInf //soma o limite inferior ao numero gerado já arredondado
24 alert( numAleat); //escreve o numero na janela de alerta
```


Funções diversas

```
1  alert(Math.log(5)); //logarítmo de 5
2
3  alert(Math.sin(5));
4  Resultado: -0.9589242746631385 //retorna o seno do ângulo informado no parâmetro
5
6  alert(Math.cos(5)); //retorna o cosseno do ângulo informado no parâmetro
7
8  alert(Math.tan(5)); //retorna a tangente do ângulo informado no parâmetro
9
10 alert(Math.asin(0)); //retorna o valor em radianos, representando o arco,
11 //cujo seno foi informado no parâmetro.
12
13 alert(Math.acos(0)); /*retorna o valor em radianos, representando o arco,
14 cujo cosseno foi informado no parâmetro.*/
15
16 alert(Math.atan(0)); /*retorna um valor numérico entre -pi/2 e pi/2, representando o arco em radianos,
17 cuja atangente foi informada no parâmetro da função.*/
18
19 alert(Math.atan2(2,2));
20 | /*retorna um valor numérico entre -pi e pi, representando o arco em radianos,
21 cuja atangente é igual ao quociente dos parâmetros informados.*/
```


Momento Hands On



Na compra de duas unidades de um mesmo medicamento, o cliente recebe como o desconto os centavos do valor final.

Elaborar um programa que leia descrição e preço de um medicamento. Informe o valor do produto na promoção.

Exemplo:

Medicamento: Aspirina

Preço R\$: 7.30

Promoção da Aspirina Leve 2 por apenas R\$: 14.00

Faça um site HTML com código JS, que pede o raio de um círculo para o internauta.

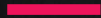
Em seguida exiba a área do círculo e o comprimento da circunferência com aquele raio.

Para resolver esse exercício, vamos precisar de duas fórmulas bem conhecidas da Matemática:

Área do círculo: $\pi * \text{raio}^2$

Comprimento da circunferência: $2 * \pi * \text{raio}$

Lembre-se de versionar seus
projetos no git.



Referências

- <https://developer.mozilla.org/pt-BR/docs/Aprender/JavaScript>
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/String
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Math
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array
- https://developer.mozilla.org/pt-BR/docs/DOM/Referencia_do_DOM
- <https://www.w3schools.com/>
- <http://docplayer.com.br/17393758-Javascript-eventos-e-objetos-nativos.html>
- <https://www.adalgisa-souza.appspot.com/javascript/>

OBRIGADO!



profalex.deus@fiap.com.br



/alexsanderresende

FIAP

Copyright © 2018 | Professor (a) Alex Sander Resende de Deus

Todos os direitos reservados. A reprodução ou divulgação total ou parcial deste documento é expressamente proibida sem o consentimento formal, por escrito, do professor/autor.

