Unidade 4 – Objetos

4.1 O Objeto Math

O objeto Math é um objeto com um comportamento um pouco diferente dos demais. Para utilizarmos o objeto Math, **não é necessário criar um nova instância** deste em uma variável, basta chamarmos seus métodos e propriedades diretamente.

MÉTODOS	SINTAXE	DESCRIÇÃO
abs()	Math.abs(x);	Devolve o valor absoluto (valor positivo) de x, isto é, elimina o símbolo negativo de um número. *Ex.: num = Math.abs(-3);* *Resultado: num = 3*
ceil()	Math.ceil(x);	Devolve o inteiro superior ou igual ao x. Este método não arredonda o número. *Ex.: num = Math.ceil(1.01);* *Resultado: num = 2*
floor()	Math.floor(x)	Devolve o inteiro inferior ou igual ao x. Este método não arredonda o número. *Ex.: num = Math.floor(1.01);* *Resultado: num = 1*
round()	Math.round(x)	Arredonda o número ao inteiro mais próximo. *Ex.: num = Math.round(1.81);* *Resultado: num = 2*
max()	Math.max(x,y)	Devolve o maior de 2 números x e y. *Ex.: num = Math.max(4,8);* *Resultado: num = 8*
min()	Math.min(x,y)	Devolve o menor de 2 números x e y. *Ex.: num = Math.min(4,8);* *Resultado: num = 4*
pow()	Math.pow(x,y)	Calcula o valor de um número x elevado na potência y. *Ex.: num = Math.pow(2,4);* *Resultado: num = 16*
random()	Math.random()	Devolve o valor de um número aleatório escolhido entre 0 e 1. *Ex.: num = Math.random();* *Resultado: qualquer número entre 0 e 1*
sqrt()	Math.sqrt(x)	Devolve a raiz quadrada de x. *Ex.: num = Math.sqrt(25);* *Resultado: num = 5*

Podemos ver um exemplo de utilização do objeto Math no exemplo a seguir.

```
<script language="javascript1.1" type="text/javascript">
var sNomes    = new Array();
var nNumero1 = 10.5;
var nNumero2 = -10.5;
var nNumero3 = 4;
var nNumero4 = 12;

alert("Método ABS(10.5): " + Math.abs(nNumero1));
alert("Método ABS(-10.5): " + Math.abs(nNumero2));
alert("Método SQRT(Raiz Quadrada 4): " + Math.sqrt(nNumero3));
alert("Método Maximo(4,12): " + Math.max(nNumero3, nNumero4));
</script>
```

Este objeto é utilizado para realizar cálculos matemáticos mais avançados, que não podem ser realizados utilizando os operadores mais simples do JavaScript.

Se for necessário arredondar um número real para um valor com ***n* casas decimais** pode-se utilizar o seguinte procedimento.

```
x = Math.round(10.5286 * 100)/100;//duas casas decimais para 3 utilize *1000...
document.write("Valor real arredondado "+x+"</br>");
```

Outra alternativa pode ser a função **tofixed()**, conforme abaixo.

```
var numb = 123.23454;
numb = numb.toFixed(2);
document.write("Valor real arredondado " + numb + "</br>");
```

Geração de números aleatórios inteiros em um determinado intervalo:

```
aleatorio=Math.floor(Math.random() * 10); // números entre 0 e 9

aleatorio = Math.floor(Math.random() * 10) + 1; // números entre 1  e 10
```

4.2 O Objeto String

O objeto string representa variáveis de texto dentro do Javascript. Os objetos string são criados através do **comando String() ou através da atribuição direta de um valor para uma variável**. O objeto string possui uma série de métodos que permitem manipular as strings no Javascript. **A posição da string começa sempre em zero. Os métodos charAt(), indexOf() e lastindexOf() retornam -1 caso não encontrem o conteúdo na string. As funções diferenciam maiúsculas e minúsculas.**

PROPRIEDADE	DESCRIÇÃO
Length	Devolve um inteiro que indica o comprimento da cadeia de caracteres.
MÉTODOS	**DESCRIÇÃO**
charAt()	Mostra o caracter na posição pedida.
indexOf()	Devolve a posição de um caracter ou cadeia de caracteres (pesquisa feita da esquerda para a direita).
lastIndexOf()	Devolve a posição de um caracter ou cadeia de caracteres (pesquisa feita da direita para a esquerda).
substring(x,y)	Devolve uma string parcial situada entre a posição x e a posição y-1.
toLowerCase()	Transforma todas as letras em minúsculas.
toUpperCase()	Transforma todas as letras em Maiúsculas.

Podemos ver a implementação de alguns desses métodos no exemplo abaixo:

```
<script language="javascript1.1" type="text/javascript">
var sString1 = "string criada diretamente pela variável";
var sString2 = new String("STRING CRIADA PELO NEW");
sString1 = sString1.toUpperCase();
sString2 = sString2.toLowerCase();
alert(sString1);
alert(sString2);
alert(sString1.substring(0,6));
alert("O tamanho da String1 é: " + sString1.length);
</script>
```

4.3 O Objeto Date

O JavaScript lida com datas de forma semelhante ao Java. As duas linguagens têm muitos dos mesmos métodos de data e as duas linguagens **armazenam datas como o número de milissegundos desde 1º de janeiro de 1970, 00:00:00 (método Date.now()).**

O objeto Date é um dos objetos intrínsecos do Javascript, utilizado para o gerenciamento de datas. Para criar uma nova instancia do objeto, basta criar uma nova variável e atribuir uma nova instancia do objeto Date. **Caso não for passado valor no momento da criação**, o objeto date é sempre inicializado com a **Data e Hora atuais**. Um exemplo de criação de um objeto Date pode ser visto a seguir.

```
<script language="javascript1.1" type="text/javascript">
var dtData, nDia, nMes, nAno

dtData = new Date(); //atribui a variável dtData, a data do sistema
nDia = dtData.getDate(); // atribui o dia
nMes = dtData.getMonth(); /* atribui o mês. Retorna um valor de 0 a 11, onde o 0
é igual a Janeiro */
nAno = dtData.getFullYear(); // atribui o ano com 4 dígitos
alert(nDia + "/" + (nMes+1) + "/" + nAno);

nHora = dtData.getHours(); // atribui a hora
nMinutos = dtData.getMinutes(); // atribui os minutos
nSegundos = dtData.getSeconds(); // atribui os segundos
alert(nHora + ":" + nMinutos + ":" + nSegundos);

nDiaSemana = dtData.getDay(); /* atribui o dia da semana. Retorna um valor de 0
a 6, onde 0 é igual a domingo */
alert("Dia da semana: " + nDiaSemana);
</script>
```

MÉTODOS	DESCRIÇÃO
getDate()	Devolve o dia do mês.
getMonth()	Devolve o mês (começa por 0).
getFullYear()	Retorna o ano com todos os dígitos.
getDay()	Devolve o dia da semana (começa por domingo e vale 0).
getHours()	Retorna a hora.
getMinutes()	Devolve os minutos.
getSeconds()	Devolve os segundos.

Todos métodos get destacados na tabela acima, tem seus respectivos métodos set para setar

datas(ex. setDate(), setMonth(),etc).

Métodos de Formatação de Data:

toLocaleDateString()	Returns the date portion of a Date object as a string, using **locale conventions**
toLocaleTimeString()	Returns the time portion of a Date object as a string, using **locale conventions**
toLocaleString()	Converts a Date object to a string, using **locale conventions**
toString()	Converts a Date object to a string

Existem geralmente 3 tipos de formatos de entrada de data JavaScript:

Tipo	Exemplo
ISO Date (ano-mês-dia)	"2018-03-25" (The International Standard) (Atenção !! ver time zone)
Short Date (mês/dia/ano)	"03/25/2018"
Long Date (mês dia ano)	"Mar 25 2018" or "25 Mar 2018"

Checar se uma data é válida e utilizar o formato DD/MM/YYYY (testar por exemplo, 30/02/2020):

os tipos input usar o atributo "value".

```
var data_tela = prompt("Digite uma data no formato DD/MM/YYYY ");

var ano = data_tela.substring(6, 10);
var mes = data_tela.substring(3, 5);
var dia = data_tela.substring(0, 2);

var data = new Date(mes + '/' + dia + "/" + ano);

document.write("Data Digitada " + data.toLocaleDateString());
if (isNaN(data) || parseInt(dia) != parseInt(data.getDate())) {
    alert("Cuidado, Data Inválida!!");
}
```

Cálculos com Datas:

Soma de dias:

```
var dataatual = new Date();
dataatual.setDate(dataatual.getDate() + 7);
document.write('Data + 7 dias ' + dataatual.toLocaleDateString());
```

Diferença entre datas:

```
var dataatual = new Date();
var dtvencto = new Date('2020-03-01');
document.write('Dias vencidos: ' + Math.floor((dataatual - dtvencto) / (1000 * 60
* 60 * 24)));
//(1000 * 60 * 60 * 24) -> calcula o número de milisegundos em um dia porque a
diferença entre duas datas é dada em milisegundos
```

Data e hora:

```
//criar data e hora
var data = new Date('04/18/2019 15:20:30')
document.write('Data e Hora ' + data.toLocaleString());
```

4.4 Objetos criados pelo usuário

Além dos objetos intrínsecos, o Javascript também **nos permite criar objetos definidos pelo usuário**, com propriedades próprias definidas no próprio script.

Para criarmos um objeto definido pelo usuário no Javascript, **devemos criar uma variável e atribuir uma nova instancia de um objeto do tipo Object**. Após criar esta variável, basta atribuir as propriedades ao objeto que elas serão criadas de forma automática. Podemos visualizar a utilização de objetos definidos pelo usuário no exemplo abaixo:

```
var person = {
  firstName: "John",
  lastName: "Doe",
  id: 5566,
  dt_nascto: '01/13/1987'
};
```