SIMULADOS & TUTORIAIS

TUTORIAIS JAVASCRIPT

O Objeto Array

O Objeto Array

Introdução

O que é uma Array?

Uma **Array** é usada para armazenar múltiplos valores em uma única variável, ou seja, é uma variável especial que pode armazenar mais de um valor por vez. Por exemplo, se você tivesse uma lista de itens (nomes de pessoas) da forma convencional você armazenaria esses nomes da seguinte forma:

```
var pessoa1 = "Maria";
var pessoa2 = "Joana";
var pessoa3 = "Carol";
var pessoa4 = "Emma";
```

Agora imagine se você tivesse que armazenar 200 nomes e não quatro? Já pensou? Então a solução seria armazenar em uma **array**. Esse mesmo exemplo você poderia fazer da seguinte forma:

```
var pessoas = new Array();
pessoas[0] = "Maria";
pessoas[1] = "Joana";
pessoas[2] = "Carol";
pessoas[3] = "Emma";
```

Uma array pode armazenar muitos valores em um único nome, e você poderá acessar qualquer um desses nomes utilizando o índice da **array**. Por exemplo, para você acessar **Carol** você faria assim:

```
var minhaFilha = pessoas[2];
```

Como criar uma Array?

Uma **Array** pode ser criada de três formas:

1. Regular:

```
var pessoas = new Array();
pessoas[0] = "Maria";
pessoas[1] = "Joana";
pessoas[2] = "Carol";
pessoas[3] = "Emma";
```

2. Condensada:

```
var pessoas = new Array("Maria","Joana","Carol","Emma");
```

3. Literal:

```
var pessoas = ["Maria","Joana","Carol","Emma"];
```

Como acessar uma Array?

O elemento de uma **Array** pode ser acessado utilizando-se o índice (**index**) desse elemento na array. Por exemplo, para criar uma variável fazendo referência ao segundo elemento da array você poderia fazer assim:

```
var pessoa = pessoas[1];
```

Nota: O zero é o primeiro índice de uma array, então o segundo elemento é o de índice 1, o terceiro é o de índice 2 e assim por diante.

Você pode ter diferentes objetos em uma array. Por exemplo:

```
myArray[0] = Date.now;
myArray[1] = minhaFuncao;
myArray[2] = pessoas;
```

Todas as variáveis em JavaScript são objetos. Os elementos de uma array também são objetos. Funções são objetos. Devido a isso, você pode ter diferentes tipos de objetos na mesma **Array**. Você pode ter funções em uma array, como também arrays em uma **Array**.

Propriedades e métodos de uma Array

O objeto **Array** possui propriedades e métodos pré-definidos, conforme mostramos a seguir:

Propriedades:

- constructor
- length
- prototype

Métodos:

- concat()
- indexOf()
- join()
- lastIndexOf()
- pop()
- push()
- reverse()
- shift()
- slice()
- sort()
- splice()
- toString()
- unshift()
- valueOf()

Vejamos alguns deles:

O método concat()

Esse método é usado para unir duas ou mais arrays. Ele não altera as arrays originais, ele retorna uma nova array contendo os valores das arrays unidas.

Sua sintaxe é a seguinte:

```
array1.concat(array2);
```

Vejamos agora um exemplo prático:

Exemplo 1

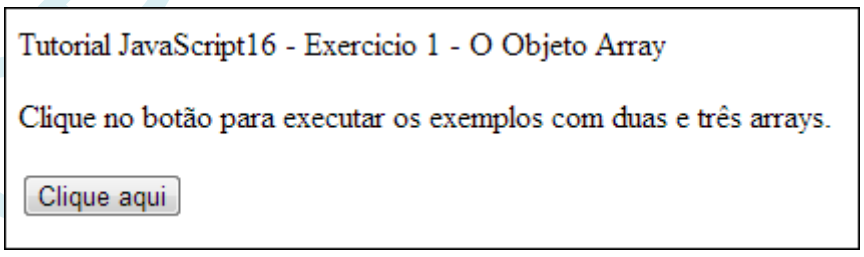
Nesse exemplo vamos utilizar o método **concat()** para unir duas e três arrays.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo1.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 – Exemplo 1 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 – Exemplo 1 - O Objeto Array</title><p>
<script language="javascript">
function minhaFuncao()
{
        var carros1 = ["Honda", "Ferrari"];
        var carros2 = ["Corolla", "Audi", "Jaguar"];
        var carros3 = ["Viper"];
        var carroes1 = carros1.concat(carros3);
        var carroes2 = carros3.concat(carros2,carros1);
        document.write(carros1 + "<br>") ;
        document.write(carros2);
}
</script>
</head>
<body>
<p>Clique no botão para executar os exemplos com duas e três arrays.</p>
<button onclick="minhaFuncao()">Clique aqui</button>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



3. Quando o botão for clicado, o resultado será o seguinte:

```
Honda,Ferrari,Viper
Viper,Corolla,Audi,Jaguar,Honda,Ferrari
```

O método indexOf()

Esse método procura por um item especifico (um elemento) em uma array, e retorna sua posição. A pesquisa inciará na posição especificada, ou no começo se não for especificada uma posição inicial, e a pesquisa termina no final da array. Se o item procurado não for encontrado será retornado **-1**. Se o item estiver presente mais de uma vez na array, o método **indexOf()** retorna a posição da primeira ocorrência. Lembre-se que o primeiro ele-

mento de uma array tem índice 0. Se você quiser procurar a pesquisar a partir do final da array utilize o método **lastIndexOf()**.

Sua sintaxe é a seguinte:

```
array.indexOf();
```

Vejamos um exemplo prático:

Exemplo 2

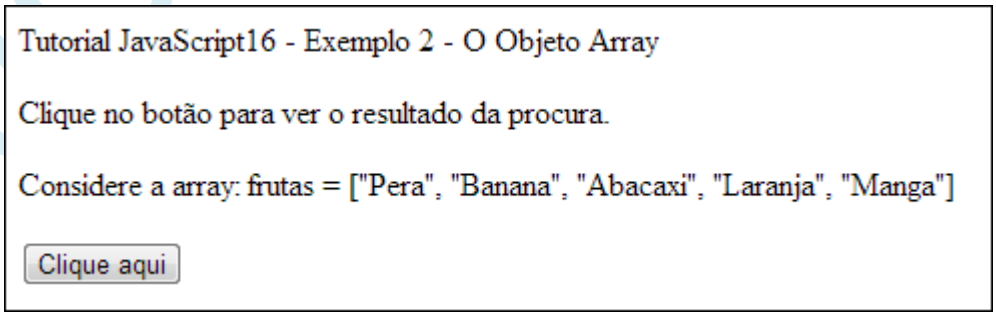
Nesse exemplo vamos utilizar o método **indexOf()** para procurar dois elementos em uma array. Primeiramente um existente e em seguida um não existente.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo2.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 2 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 2 - O Objeto Array</title><p>
<script language="javascript">
function minhaFuncao()
{
var frutas = ["Pera", "Banana", "Abacaxi", "Laranja", "Manga"];
var a = frutas.indexOf("Laranja")
document.write("Resultado da pesquisa por Laranja: " + a + "<p>");
var b = frutas.indexOf("Maçã");
document.write("Resultado da pesquisa por Maçã: " + b);
}
</script>
</head>
<body>
<p>Clique no botão para ver o resultado da procura.</p>
Considere a array: frutas = ["Pera", "Banana", "Abacaxi", "Laranja", "Manga"]<p>
<button onclick="minhaFuncao()">Clique aqui</button>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



3. Quando o botão for clicado, o resultado será o seguinte:

```
Resultado da pesquisa por Laranja: 3

Resultado da pesquisa por Maçã: -1
```

O método join()

Esse método une os elementos de uma array em uma string e retorna a string. Os elementos serão separados por um separador específico. O separador padrão é a vírgula (,).

Sua sintaxe é a seguinte:

```
array.join();
```

Vejamos um exemplo prático:

Exemplo 3

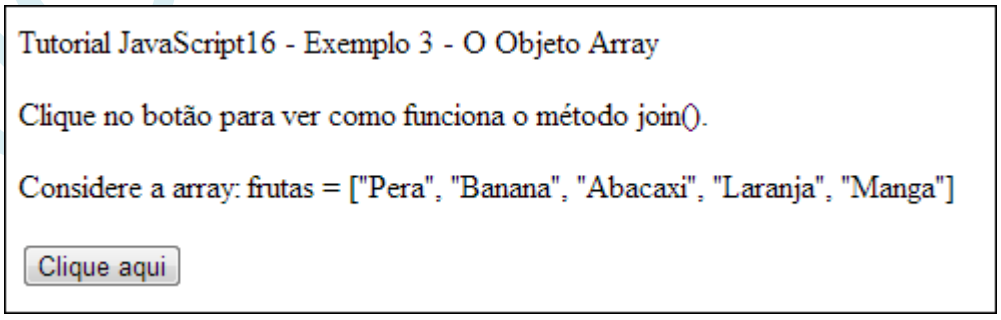
Nesse exemplo vamos utilizar o método **join()** para converter uma array em uma string separando seus elementos por vírgula.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo3.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 3 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 3 - O Objeto Array</title><p>
<script language="javascript">
function minhaFuncao()
{
var frutas = ["Pera", "Banana", "Abacaxi", "Laranja", "Manga"];
document.write(frutas.join());
}
</script>
</head>
<body>
<p>Clique no botão para ver como funciona o método join().</p>
Considere a array: frutas = ["Pera", "Banana", "Abacaxi", "Laranja", "Manga"]<p>
<button onclick="minhaFuncao()">Clique aqui</button>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



3. Quando o botão for clicado, o resultado será o seguinte:

```
Pera,Banana,Abacaxi,Laranja,Manga
```

Os métodos push() e pop()

O método **push()** aceita qualquer número de argumentos e adiciona-os no final da array retornando o novo comprimento da array. O método **pop()** remove o último elemento da array, decrementa o comprimento da array e retorna esse elemento.

Suas sintaxes são as seguintes:

```
array.push();
```

```
array.pop();
```

Vejamos um exemplo prático:

Exemplo 4

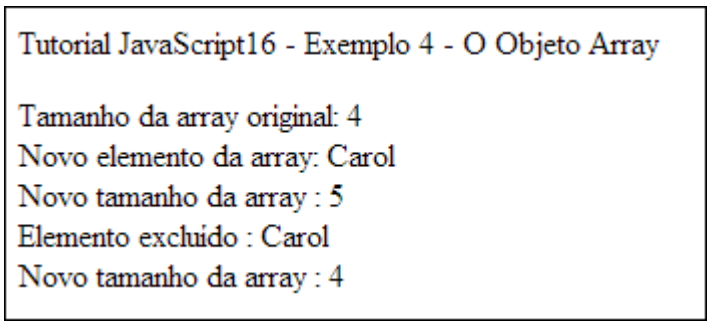
Nesse exemplo vamos utilizar o método **push()** para inserir um novo elemento no final de uma array dada, e em seguida retirá-lo utilizando o método **pop()**.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo4.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 4 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 4 - O Objeto Array</title><p>
<script language="javascript">
      // Cria uma array.
var pessoas = new Array();
      // Insere quatro elementos e armazena na variável tamanho.
        var tamanho = pessoas.push("Emília", "Jorge", "Maria", "Emma");
      // Mostra o tamanho da array.
        document.writeln("Tamanho da array original: " + tamanho + "<br>");
      // Insere um novo elemento da array.
        var item = pessoas.push("Carol");
       // Mostra o novo elemento da array.
      document.writeln("Novo elemento da array: " + pessoas[4] + "<br>");
     // Mostra o novo tamanho da array.
      document.writeln("Novo tamanho da array : " + item + "<br>");
      // Retira o último elemento da array.
      var item = pessoas.pop();
      // Mostra o elemento excluído.
      document.writeln("Elemento excluído : " + item + "<br>");
      // Mostra o novo tamanho da array.
      document.writeln("Novo tamanho da array : " + pessoas.length);
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



O método shift()

Esse método remove o primeiro elemento de uma array e retorna o comprimento dessa array decrementado em uma unidade.

Sua sintaxe é a seguinte:

```
array.shift();
```

Vejamos um exemplo prático:

Exemplo 5

Nesse exemplo vamos utilizar o método **shift()** para remover o primeiro elemento de uma array dada.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo5.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 5 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 5 - O Objeto Array</title><p>
<script language="javascript">
        // Cria uma array vazia.
        var pessoas = new Array();
        // Insere quatro elementos na array.
        pessoas = ["Emília", "Jorge", "Maria", "Emma"];
        // Mostra os elementos da array.
        document.writeln("Array original: " + pessoas + "<br>");
        // Mostra o tamanho da array.
        document.writeln("Novo tamanho da array : " + pessoas.length + "<br>");
        // Retira o primeiro elemento da array.
        var item = pessoas.shift();
        // Mostra o elemento excluído.
        document.writeln("Elemento excluído : " + item + "<br>");
        // Mostra o novo tamanho da array.
        document.writeln("Novo tamanho da array : " + pessoas.length + "<br>");
        // Mostra os elementos da array.
        document.writeln("A array sem o primeiro elemento: " + pessoas);
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:

Tutorial JavaScript16 - Exemplo 5 - O Objeto Array

Array original: Emilia,Jorge,Maria,Emma
Novo tamanho da array : 4
Elemento excluído : Emilia
Novo tamanho da array : 3
A array sem o primeiro elemento: Jorge,Maria,Emma

O método slice()

Esse método seleciona os elementos de uma array começando no parâmetro **inicio** e terminando no parâmetro **fim** (mas não incluído), e retorna uma nova array com os elementos selecionados. A array original não será alterada.

Sua sintaxe é a seguinte:

```
array.slice(inicio, fim)
```

Onde:

- **inicio** – Esse parâmetro é obrigatório. Um inteiro que especifica onde a seleção deve começar. O primeiro elemento tem indice 0. Use números negativos para selecionar os elementos a partir do final da array.
- **fim** – Esse parâmetro é opcional. Um inteiro que especifica onde a seleção deve terminar. Se for omitido, todos os elementos a partir da posição inicial até o final da array serão selecionados. Use números negativos para selecionar os elementos a partir do final da array.

Vejamos um exemplo prático:

Exemplo 6

Nesse exemplo vamos utilizar o método **slice()** para selecionar alguns elementos de uma array dada e retornar esses elementos em novas arrays.

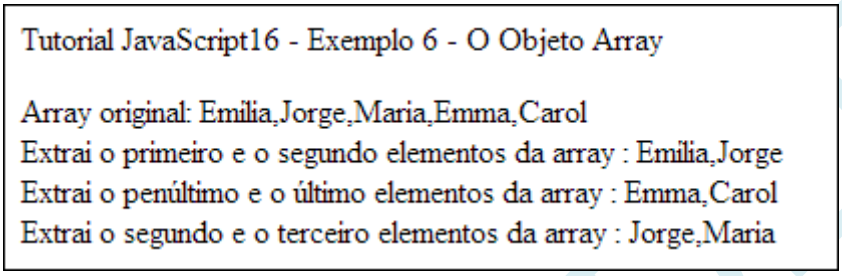
Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo6.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 6 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 6 - O Objeto Array</title><p>
<script language="javascript">
        // Cria uma array vazia.
        var pessoas = new Array();
        // Insere quatro elementos na array.
        pessoas = ["Emília", "Jorge", "Maria", "Emma", "Carol"];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Extrai o primeiro e o segundo elementos da array.
        document.writeln("Extrai o primeiro e o segundo elementos da array : " +
pessoas.slice(0,2) + "<br>");
        // Extrai o penúltimo e o último elementos da array.
```

```
        document.writeln("Extrai o penúltimo e o último elementos da array : " +
pessoas.slice(3,5) + "<br>");
        // Extrai o segundo e o terceiro elementos da array.
        document.writeln("Extrai o segundo e o terceiro elementos da array : " +
pessoas.slice(-4,-2) + "<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



O método splice()

Esse método adiciona/remove os elementos de/para uma array, e retorna uma nova array contendo os elementos removidos, se houver. Esse método altera a array original.

Sua sintaxe é a seguinte:

```
array.splice(indice, nrElementos, item1,...,itemX)
```

Onde:

- **indice** – Esse parâmetro é obrigatório. Um inteiro que especifica a partir de que posição da array os elementos deverão ser adicionados/removidos. O primeiro elemento tem indice 0. Use números negativos para selecionar os elementos a partir do final da array.
- **nrElementos** – Esse parâmetro é obrigatório. Especifica o número de elementos a serem removidos. Se for 0 (zero), nenhum item será removido.
- **item1,...,itemX** – Esse parâmetro é opcional. Aqui você informará os novos elementos a serem adicionados à array.

Vejamos um exemplo prático:

Exemplo 7

Nesse exemplo vamos utilizar o método **splice()** para adicionar dois elementos a partir do terceiro (índice 2) sem remover nenhum elemento, e em seguida remover os três últimos elementos.

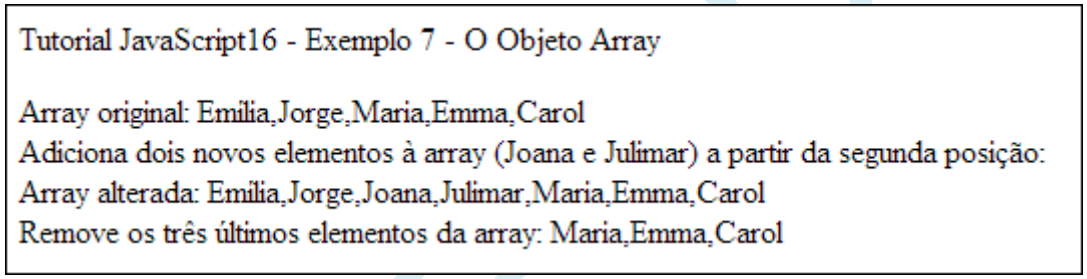
Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo7.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 7 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 7 - O Objeto Array</title><p>
```

```
<script language="javascript">
        // Cria uma array vazia.
        var pessoas = new Array();
        // Insere quatro elementos na array.
        pessoas = ["Emília", "Jorge", "Maria", "Emma", "Carol"];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Adiciona dois novos elementos à array a partir da terceira posição.
        document.writeln("Adiciona dois novos elementos à array (Joana e Julimar) a
partir da segunda posição: " + pessoas.splice(2,0,"Joana","Julimar") + "<br>");
        document.writeln("Array alterada: " + pessoas + "<br>");
        // Remove os três últimos elementos da array.
        document.writeln("Remove os três últimos elementos da array: " + pesso-
as.splice(4,3) + "<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



O método unshift()

Esse método adiciona novos elementos no começo de uma array e retorna o novo comprimento dessa array. Esse método altera o comprimento da array.

Sua sintaxe é a seguinte:

```
array.unshift(item1, item2, ..., itemX)
```

Onde:

- **item1, item2, ..., itemX** – Esse parâmetro é obrigatório. Indica os elementos que deverão ser adicionados no início da array.

Vejamos um exemplo prático:

Exemplo 8

Nesse exemplo vamos utilizar o método **unshift()** para adicionar dois elementos no início de uma array dada.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo8.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 8 - O Objeto Array
```

```
<head>
   <title>Tutorial JavaScript16 - Exemplo 8 - O Objeto Array</title><p>
<script language="javascript">
        // Cria uma array vazia.
        var pessoas = new Array();
        // Insere quatro elementos na array.
        pessoas = ["Emília", "Jorge", "Maria"];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Adiciona dois novos elementos no início da array.
        document.writeln("Adiciona dois novos elementos no início da array (Joana e
Julimar): " + pessoas.unshift("Joana","Julimar") + "<br>");
        document.writeln("Array alterada: " + pessoas + "<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:

Tutorial JavaScript16 - Exemplo 8 - O Objeto Array

Array original: Emília,Jorge,Maria
Adiciona dois novos elementos no início da array (Joana e Julimar): 5
Array alterada: Joana,Julimar,Emília,Jorge,Maria

O método sort()

Esse método classifica os elementos de uma array. A ordem de classificação pode ser alfabética ou numérica, e ascendente ou descendente. A ordem de classificação padrão é alfabética e ascendente. Quando os números são sorteados alfabeticamente, não fica muito legal, pois nesse caso, 40 vem antes de 5.

Para executar uma classificação numérica, você deve passar uma função como argumento quando chamar o método **sort()**. A função deve especificar se os números deverão ser classificados de forma ascendente ou descendente. Pode ser que seja difícil entender como essa função funciona, mas com o exemplo a seguir você entenderá melhor. Esse método altera a array original, e retorna um objeto **Array** como os elementos classificados.

Sua sintaxe é a seguinte:

```
array.sort(funcaoSort)
```

Onde:

- **funcaoSort** – Esse parâmetro é opcional. É uma função que define a ordem de classificação dos elementos.

Vejamos um exemplo prático:

Exemplo 9

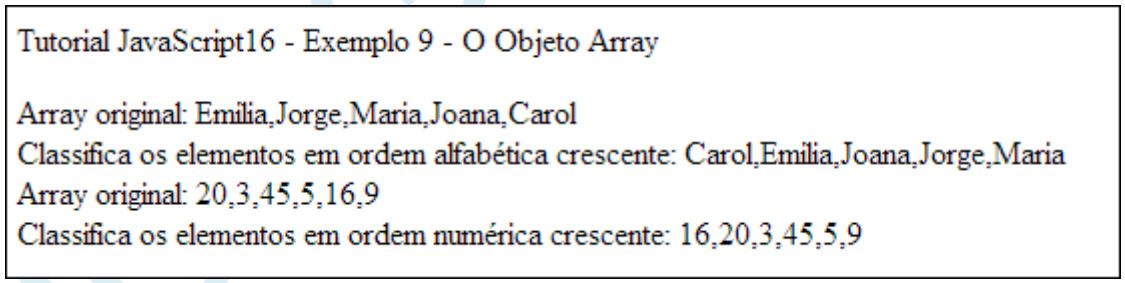
Nesse exemplo vamos utilizar o método **sort()** para classificar os elementos de uma array dada, sendo uma classificação alfabética e outra numérica.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo9.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 9 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 9 - O Objeto Array</title><p>
<script language="javascript">
        // Cria uma array vazia.
        var pessoas = new Array();
        // Insere seis elementos na array.
        pessoas = ["Emília", "Jorge", "Maria", "Joana", "Carol"];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Classifica os elementos em ordem alfabética crescente.
        document.writeln("Classifica os elementos em ordem alfabética crescente: " +
pessoas.sort() + "<br>");
        // Cria uma array vazia.
        var numeros = new Array();
        // Insere seis elementos na array.
        numeros = [20, 3, 45, 5, 16, 9];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + numeros + "<br>");
        // Classifica os elementos em ordem numérica crescente.
        document.writeln("Classifica os elementos em ordem numérica crescente: " +
numeros.sort() + "<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



Conforme você deve ter observado, a classificação numérica não ficou conforme o esperado, tendo em vista que o método só considera o primeiro número, ignorando o segundo número, ou seja, para o método os números seriam: 1, 2, 3, 4, 5, 9. Para que uma classificação numérica seja realista teremos que criar uma função como parâmetro do método, conforme veremos no próximo exemplo.

Exemplo 10

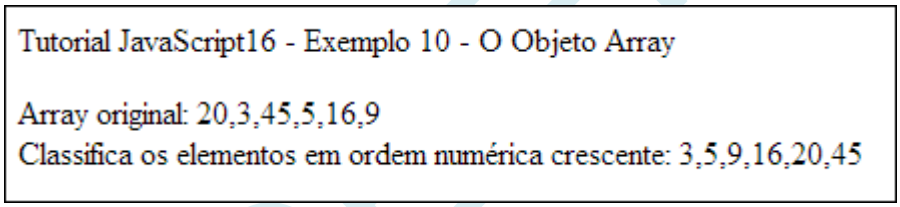
Nesse exemplo vamos utilizar o método **sort()** para classificar os elementos de uma array numérica utilizando uma função como parâmetro.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo10.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 10 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 10 - O Objeto Array</title><p>
<script language="javascript">
        // Cria uma array vazia.
        var numeros = new Array();
        // Insere seis elementos na array.
        numeros = [20, 3, 45, 5, 16, 9];
        // Mostra os elementos da array original.
        document.writeln("Array original: " + numeros + "<br>");
        // Cria uma função para o método e armazena em uma variável.
        var classifica = numeros.sort(function(a,b){return a-b});
        // Classifica os elementos em ordem numérica crescente.
        document.writeln("Classifica os elementos em ordem numérica crescente: " +
classifica + "<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



O método reverse()

Esse método inverte a ordem dos elementos de uma array qualquer, ou seja, de trás pra frente.

Sua sintaxe é a seguinte:

```
array.reverse();
```

Vejamos um exemplo prático:

Exemplo 11

Nesse exemplo vamos utilizar o método **reverse()** para inverter os elementos de duas arrays, sendo uma alfabética e outra numérica.

Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo11.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 11 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 11 - O Objeto Array</title><p>
```

```
<script language="javascript">
        // Cria uma array com seis elementos.
        var pessoas = new Array("Emília", "Jorge", "Maria", "Joana", "Carol");
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Inverte os elementos da array.
        document.writeln("Inverte os elementos da array: " + pessoas.reverse() +
"<br>");
        // Cria uma array com valores numéricos.
        var numeros = new Array(20, 3, 45, 5, 16, 9);
        // Mostra os elementos da array original.
        document.writeln("Array original: " + numeros + "<br>");
        // Inverte os elementos da array.
        document.writeln("Inverte os elementos da array: " + numeros.reverse() +
"<br>");
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:

Tutorial JavaScript16 - Exemplo 11 - O Objeto Array

Array original: Emilia,Jorge,Maria,Joana,Carol
Inverte os elementos da array: Carol,Joana,Maria,Jorge,Emilia
Array original: 20,3,45,5,16,9
Inverte os elementos da array: 9,16,5,45,3,20

O método valueOf()

Esse método retorna o valor primitivo do objeto, ou seja, retorna os elementos da array original separados por vírgula. Se uma array contiver outra array, os conteúdos serão otimizados quando esse método for utilizado.

Sua sintaxe é a seguinte:

```
array.valueOf();
```

Vejamos um exemplo prático:

Exemplo 12

Nesse exemplo vamos utilizar o método **valueOf()** para retornar os elementos de uma array contendo outra array separados por vírgula.

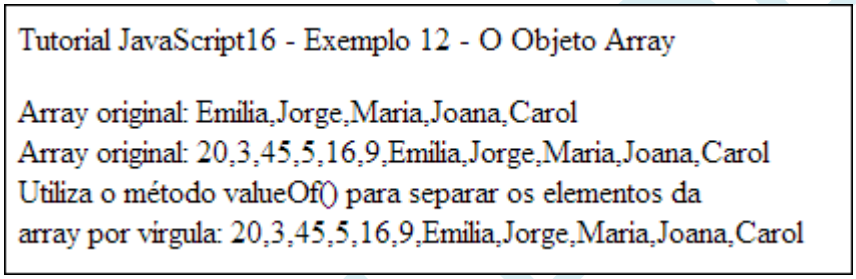
Para isso:

1. Digite o código abaixo no seu editor de texto e salve-o como: **js16-exemplo12.html**.

```
<!DOCTYPE html>
<html>
Tutorial JavaScript16 - Exemplo 12 - O Objeto Array
<head>
   <title>Tutorial JavaScript16 - Exemplo 12 - O Objeto Array</title><p>
```

```
<script language="javascript">
        // Cria uma array com seis elementos.
        var pessoas = new Array("Emília", "Jorge", "Maria", "Joana", "Carol");
        // Mostra os elementos da array original.
        document.writeln("Array original: " + pessoas + "<br>");
        // Cria uma array com seis elementos.
        var numeros = new Array(20, 3, 45, 5, 16, 9, pessoas);
        // Mostra os elementos da array original.
        document.writeln("Array original: " + numeros + "<br>");
        // Utiliza o método valueOf() para separar os elementos da array por vírgula.
        document.writeln("Utiliza o método valueOf() para separar os elementos
da<br> array por vírgula: " + numeros.valueOf());
</script>
</head>
<body>
</body>
</html>
```

2. O resultado desse código após executado será o seguinte:



*****

Exercícios de fixação

1) Qual dos métodos abaixo é utilizado para inverter a ordem dos elementos de uma array qualquer, ou seja, de trás pra frente?

   a) revert()
   b) slice()
   c) splice()
   d) reverse()

2) Esse método classifica os elementos de uma array. A ordem de classificação pode ser alfabética ou numérica, e ascendente ou descendente.

   a) sort()
   b) order()
   c) classify()
   d) unshift()

3) O método _____ é utilizado para retornar os elementos de uma array contendo outra array separados por vírgula.

   a) shift()
   b) valueOf()
   c) slice()
   d) splice()

4) A ordem de classificação padrão do método **sort()** é:

   a) alfabética e descendente
   b) numérica e crescente
   c) alfabética e ascendente
   d) numérica e descendente

5) O método _____ aceita qualquer número de argumentos e adiciona-os no final da array retornando o novo comprimento da array.

   a) shift()
   b) pop()
   c) split()
   d) push()

6) O método _____ remove o último elemento da array, decrementa o comprimento da array e retorna esse elemento.

   a) pop()
   b) splice()
   c) slice()
   d) shift()

7) Considere a array abaixo:

```
var pessoas = new Array("Maria","Joana","Carol","Emma","Marina","Jomara");
```

   Quais os elementos que correspondem aos índices 2 e 4?

   a) Joana e Emma
   b) Carol e Marina
   c) Joana e Marina

d) Carol e Jomara

8) Considere duas arrays dadas, **array1** e **array2**. Qual a sintaxe correta do método **concat()** para concatenar essas duas arrays?

a) array1(concat(array2));
b) array1.concat[array2];
c) array1[concat(array2)];
d) array1.concat(array2);

9) Que método seleciona os elementos de uma array começando no parâmetro **inicio** e terminando no parâmetro **fim** (mas não incluído), e retorna uma nova array com os elementos selecionados?

a) shift()
b) slice()
c) join()
d) concat()

10) Qual a forma correta de se criar uma nova **Array** vazia?

a) var pessoas = new Array();
b) var pessoas = Array();
c) var pessoas = new Array{};
d) var pessoas = Array[];

11) O separador padrão do método **join()** é a vírgula (,). Essa declaração é:

a) Verdadeira
b) Falsa

12) Esse método remove o primeiro elemento de uma array e retorna o comprimento dessa array decrementado em uma unidade.

a) slice()
b) join()
c) concat()
d) shift()

13) uma array pode armazenar muitos tipos diferentes de valores. Essa declração é:

a) Verdadeira
b) Falsa

14) Listas e tabelas de valores não podem ser armazenadas em arrays. Essa declração é:

a) Verdadeira
b) Falsa

15) Que propriedade devemos utilizar para mostrar o número de elementos de uma array?

a) slice
b) length
c) concat
d) shift

*****

Exercícios propostos

1. Crie uma rotina para acrescentar os meses faltantes na array dada:

```
var meses = ["Março","Maio","Junho","Setembro","Novembro","Dezembro"];
```

2. Concatene as duas arrays abaixo e em seguida classifique o resultado em ordem crescente:

```
var nrPares = [20, 6, 2, 24, 8, 10, 0, 50];
```

```
var nrImpares = [33, 55, 1, 13, 7, 17, 3];
```

3. Desenvolva uma rotina utilizando uma array com 30 números quaisquer e em seguida calcule a média aritmética desses números.

4. Utilize a estrutura de repetição **FOR** para preencher uma array somente com números pares até 50, e mostre o resultado em uma janela de diálogo.

*****