Programação Estruturada

Aula 11 - Arrays: Introdução, Acesso e Métodos

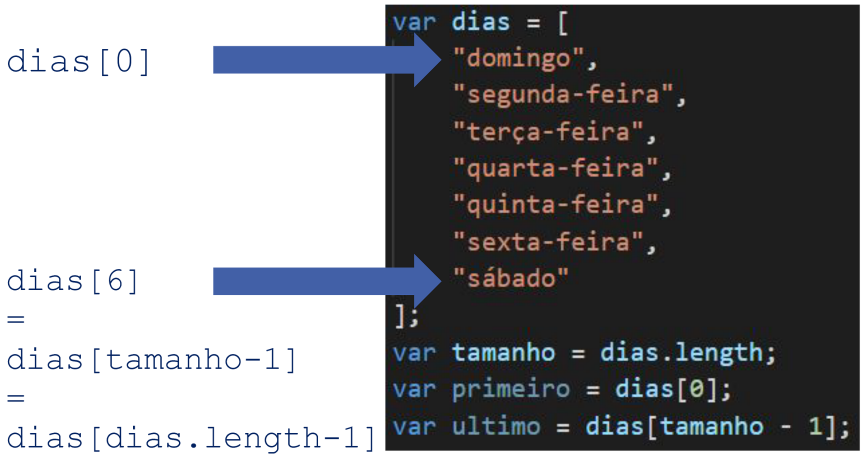
Videoaula 02: Acessando Elementos de um Array

Videoaula 02: Acessando Elementos de um Array

Olá, seja bem-vindo de volta! Vamos continuar o seu aprendizado sobre arrays?

Nesta videoaula, você aprenderá como acessar elementos de um array e iteragir sobre ele. Primeiramente, você precisa conhecer uma propriedade muito importante do array, o seu tamanho. Essa propriedade pode ser acessada usando a palavra-chave `length`. No exemplo que vemos no slide (Figura 1), depois da declaração do array, estamos atribuindo o valor de `dias.length` à variável `tamanho`. Assim, a variável receberá o valor 7, que é o tamanho do array.

Figura 1 - Acessando Elementos de um Array



11_5 Dias da Semana Acessando.html e 11_5 Dias da Semana Acessando.js

Como já foi visto, você pode acessar um elemento do array usando o nome da variável e indicando o índice da posição a ser acessada entre colchetes. Além disso, sabemos que em JavaScript os índices dos arrays começam por 0. Portanto, para acessar o primeiro elemento do array, usamos o índice 0. No slide (Figura 1), estamos atribuindo à variável `primeiro` o valor do primeiro elemento do array, ou seja, `dias[0]`. Desse modo, essa variável receberá o valor `"domingo"`.

Por outro lado, como os índices dos arrays começam em 0, o último índice do array é sempre o valor do seu tamanho menos 1. Portanto, para acessar o valor do último elemento do array, usamos a expressão `dias[dias.length-1]`. Como nós já atribuímos `dias.length` à variável `tamanho`, utilizamos essa variável na atribuição de `dias[tamanho-1]` à variável `ultimo`. Nessa atribuição, a variável `ultimo` recebe o valor do último elemento do array, ou seja, `"sábado"`.

Código 1 - 11_5 Dias da Semana Acessando.html e 11_5 Dias da Semana Acessando.js

```html
<html >
   <head>
      <meta charset="UTF-8" />
      <title>Programação Estruturada - Aula 11</title>
   </head>
   <body>
      <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>

      <h1>Dias da Semana</h1>

      <button onclick="dia_da_semana()">INICIAR</button>
      <p id="resultado"></p>

      <script src="script.js"></script>
   </body>
</html>

```

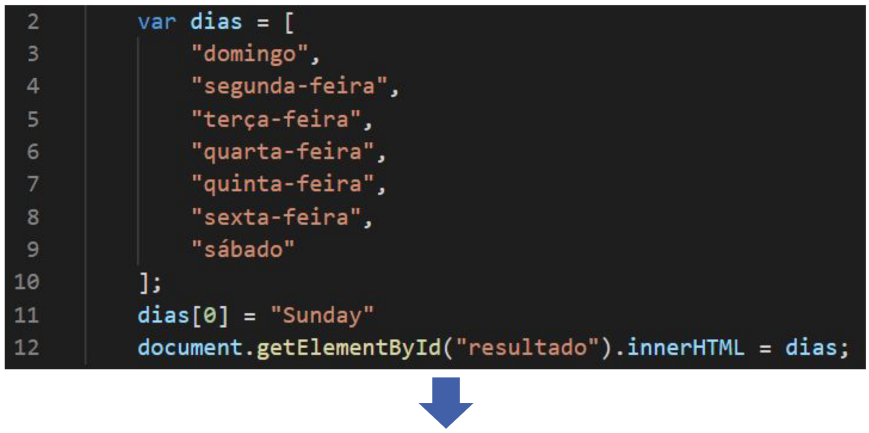
```javascript
function dia_da_semana() {
   var dias = [
      "domingo",
      "segunda-feira",
      "terça-feira",
      "quarta-feira",
      "quinta-feira",
      "sexta-feira",
      "sábado"
   ];
   var tamanho = dias.length;
   var primeiro = dias[0];
   var ultimo = dias[tamanho - 1];
   texto = "Tamanho = "+ tamanho + "<br>";
   texto = texto + "Primeiro = " + primeiro + "<br>";
   texto = texto + "Ultimo = " + ultimo;

   document.getElementById("resultado").innerHTML = texto;
```

```
19 }
20
```

Para acessar todo o array podemos simplesmente fazer referência ao seu nome. No exemplo que apresentamos no slide (Figura 2), alteramos o texto do parágrafo da página HTML para o valor da variável `dias`, ou seja, o nosso array. Isso faz com que a página exiba o texto apresentado no slide (Figura 2), o qual é o resultado da chamada ao método `toString()` nesse array, o qual, como veremos na próxima videoaula, simplesmente converte o array em uma sequência com os seus valores separados por vírgula.

Figura 2 - Acessando todo o Array



Olhe atentamente para o texto gerado. Você notou que no começo do texto nós temos `"Sunday"` e não `"domingo"`? O que aconteceu se o primeiro elemento do nosso array era `"domingo"`?

Note que na linha 11 alteramos o valor do primeiro elemento do array para o texto "Sunday" simplesmente atribuindo esse texto a `dias[0]`. Essa atribuição altera o valor do array no índice utilizado. Notou como é simples alterar o valor de um elemento do array?

Código 2 - 11_6 Dias da Semana Full.html e 11_6 Dias da Semana Full.js

```html
<html >
  <head>
    <meta charset="UTF-8" />
    <title>Programação Estruturada - Aula 11</title>
  </head>
  <body>
    <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>

    <h1>Dias da Semana</h1>

    <button onclick="dia_da_semana()">INICIAR</button>
    <p id="resultado"></p>

    <script src="script.js"></script>
  </body>
</html>

```

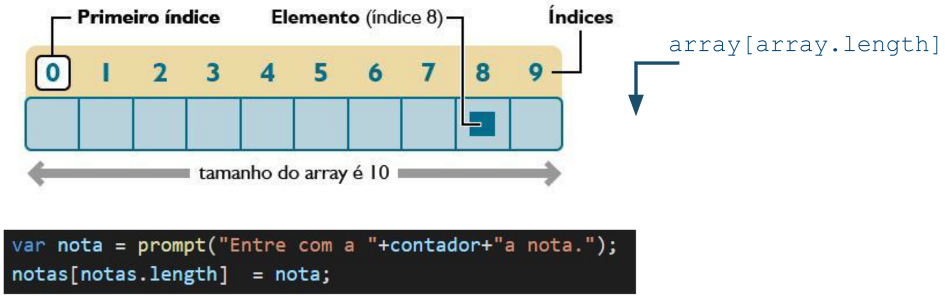
```javascript
function dia_da_semana() {
  var dias = [
    "domingo",
    "segunda-feira",
    "terça-feira",
    "quarta-feira",
    "quinta-feira",
    "sexta-feira",
    "sábado"
  ];
  dias[0] = "Sunday"
  document.getElementById("resultado").innerHTML = dias;
}

```

Agora você irá aprender a adicionar elementos a um array. A essa altura, você já sabe bem que os índices dos arrays em JavaScript começam com 0 e que, por isso, o último índice do array é igual ao tamanho do array menos um, ou seja, `array.length-1`. Assim sendo, se atribuirmos qualquer valor ao índice `array.length` do array, estaremos inserindo mais um elemento no final dele. Portanto, uma das maneiras de adicionar um novo elemento a um array é fazer uma atribuição do valor a ser inserido ao índice igual ao tamanho do array usando a propriedade `length`.

No código que apresentamos no slide (Figura 3), atribuímos o valor inserido pelo usuário à variável `nota` e depois inserimos esse valor no array de `notas` utilizando o índice `notas.length` no índice do array que recebe o valor.

Figura 3 - Adicionando Elementos ao Array



Na próxima videoaula, veremos outras formas de fazer essa inserção de elementos e algumas maneiras de remover elementos.

Para concluir esta videoaula, veja como podemos iterar sobre os elementos do array (Figura 4). A maneira mais segura de fazer isso é utilizando um laço `for` com a variável de controle assumindo todos os valores de índice do array, ou seja, de 0 até o tamanho do array menos 1.

Figura 4 - Iterando sobre Elementos de um Array

```
for(i = 0; i < notas.length; i++) {
    // Faça algo com o elemento notas[i]
}
```

Existem outras maneiras interessantes de iteragir sobre os arrays operando em todos os itens dele. Porém, nesta videoaula, apresentarei apenas a primeira maneira, ou seja, usando o laço `for`. As outras maneiras você conhecerá na próxima aula.

Para mostrar como fazer essa iteração, vamos voltar ao nosso exemplo de leitura de notas. Desta vez, vamos utilizar um array para armazenar as notas lidas. Assim, podemos tornar o nosso exemplo mais flexível. Agora, ele inicialmente receberá a quantidade de notas a serem inseridas e depois solicitará essas notas. Ao final, o nosso código utilizará o laço `for` para iteragir sobre o array a fim de exibir todas as notas inseridas.

Nesse exemplo (Ver Código 3), iremos receber a quantidade de notas através da página HTML. Logo após, iterativamente, a página perguntará quais são essas notas. Então, se eu colocar a quantidade de notas sendo 3, a página perguntará quais são as três notas e no final serão impressas na tela todas essas notas, tá certo?

Código 3 - 11_7 Nota Array.html e 11_7 Nota Array.js

```html
<html >
  <head>
    <meta charset="UTF-8" />
    <title>Programação Estruturada - Aula 11</title>
  </head>
  <body>
    <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>

    <h1>Notas com Arrays</h1>

    Quantidade de Notas: <input type="number" id="total" value="">
    <button onclick="ler_notas()">INICIAR</button>
    <p id="resultado"></p>

    <script src="script.js"></script>
  </body>
</html>

```

```javascript
function ler_notas() {
  var texto = "";
  var total = Number(document.getElementById("total").value);

  if (total < 1) {
    texto = "Escolha um valor maior do que 0."
  } else {
    var notas = [];
    var i;
    for(i = 1; i <= total; i++) {
      var nota = prompt("Entre com a "+i+"a nota.");
```

```
12          notas[notas.length]  = nota;
13
14          // O código abaixo cria undefined whole. Use N = 1
15          // notas[5]  = 100;
16          // Mencionar o push na próxima videoaula
17        }
18
19        // Usando for padrão
20        for(i = 0; i < notas.length; i++) {
21           texto = texto + "A nota " + (i+1) + " foi "+notas[i]+"<br>";
22        }
23        // Usando for/of
24        //for (nota of notas) {
25        //    texto = texto + "A nota foi "+nota+"<br>";
26        //}
27  }
28     document.getElementById("resultado").innerHTML = texto;
29  }
30
```

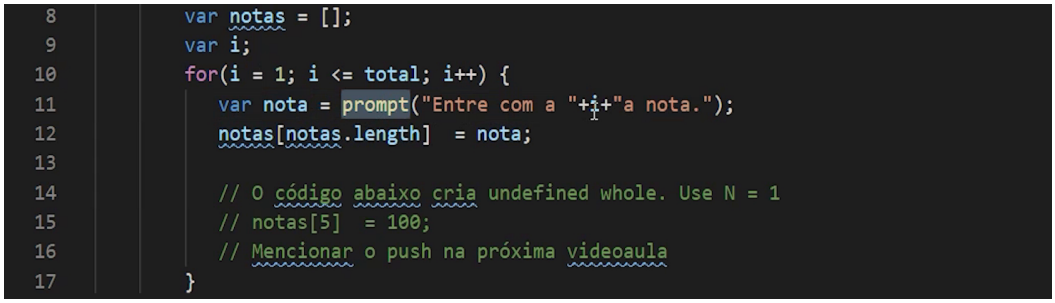
Exposto no HTML, o que temos é bem simples, um campo numérico chamado `"total"`. E, ao clicar no botão INICIAR, chamaremos a função `ler_notas()`, localizada na linha 12. No final, escreveremos algo nesse campo `"resultado"` da linha 13, ok?

No JavaScript, a função `ler_notas()` terá uma variável `texto` que será utilizada apenas no final, na linha 28, para escrever no campo `"resultado"` do HTML. Veja que começamos o texto, localizado na linha 2 , com "" (vazio), pegamos na variável `total`, localizada na linha 3, o valor do campo `"total"` (campo numérico da página) e atribuímos a ele essa mesma variável. Se esse valor for menor que 1, isso significa que ele, ou é 0, ou é algum valor negativo. Como não faz sentido eu pegar 0 notas, então o texto que aparecerá na página HTML será: "Escolha um valor maior do que 0". Mas, se esse valor for maior ou igual a 1, o `else` da linha 7 garante que teremos notas para serem lidas.

Observe na Figura 5 que criamos um array vazio de notas e declaramos o contador. Note que, nesse contador, que varia de 1 até o total de notas e é incrementado no final, serão lidas as notas. Assim, no trecho entre a linha 10 e a linha 17, as notas que o usuário inserir estão sendo lidas. Considere que essa quantidade de notas é `total`. Ao

abrir o `prompt`, veremos que o usuário vai entrar com a primeira nota, que é a nota 1, a segunda nota, a terceira nota, e assim por diante; e, por fim, será dito que `notas[notas.length]` é igual à `nota`.

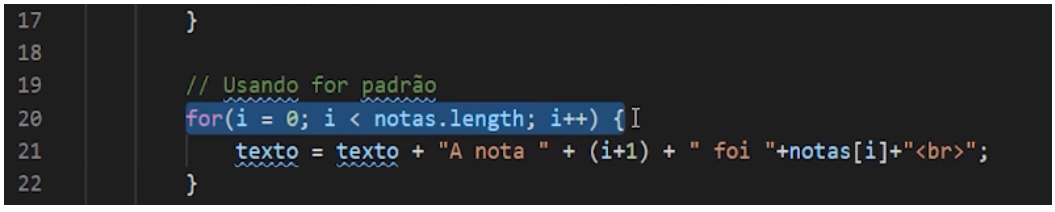
Figura 5 - Criando o Array de Notas



Isso vai colocar, como já dito, essa `nota` inserida no final do array, e faremos isso para um número total de notas. Em seguida, vou iterar sobre esse array que criamos pegando as notas, para exibir essa `nota`.

Veja na Figura 6 que na linha 20 temos um laço `for`, já visto em aulas anteriores, e o `i` variando de 0 até `notas.length-1`. Ou seja, enquanto `i` for menor que `notas.length`, ou seja, até `notas.length-1`, o `texto` será aquele que tínhamos, que havia começado com vazio, concatenado com o texto `"A nota"` concatenado com `(i+1)`, isso porque o 0 é a primeira nota, o 1 é a segunda nota, e assim por diante, concatenado com `"foi"`; e aí, pegamos `notas[i]`. Dessa forma, a primeira nota será `notas[0]`; a segunda nota será `notas[1]`; a terceira nota será `notas[2]`, e assim por diante.

Figura 6 - Exibindo as notas lidas



Verifique o comportamento desse código na página HTML. Como exemplo, temos o que acontece na UFRN: inserção de três notas. Faça o teste, colocando 5 na primeira nota, 7 na segunda e 9 na terceira. Aparecerá na tela: "A nota 1 foi 5", "A nota 2 foi 7" e

"A nota 3 foi 9". Ou seja, iteramos sobre essas notas.

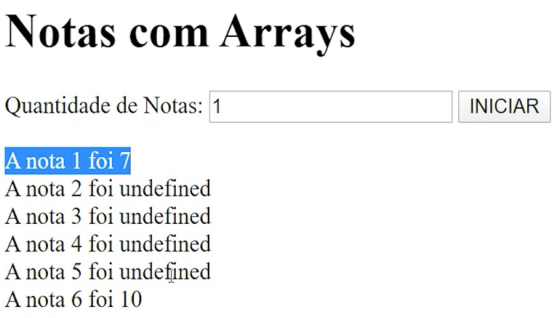
Neste exemplo (Figura 7), você verá um comportamento diferente e bem interessante. Nele, foi pedido para ser inserido apenas uma nota. Assim, foi lida apenas uma nota e ela passou uma única vez por esse `for` localizado na linha 10. Perceba que na linha 15 temos `notas[5] = 10`.

Figura 7 - Criando arrays com "buracos"

```
6         texto = "Escolha um valor maior do que 0."
7     } else {
8         var notas = [];
9         var i;
10        for(i = 1; i <= total; i++) {
11            var nota = prompt("Entre com a "+i+"a nota.");
12            notas[notas.length]  = nota;
13
14            // O código abaixo cria undefined whole. Use N = 1
15            notas[5]  = 10;
16            //Mencionar o push na próxima videoaula
17        }
```

Veja o comportamento desse código na página HTML. Ao ser inserida apenas uma nota, o número 7, e após clicar no botão INICIAR, aparecerá na tela o que está sendo exposto na Figura 8. Se você voltar ao código verá que a nota 6 é `notas[5]`, que é a sexta nota que aparece. Lembre-se que os índices começam em 0, mas entre o índice 0 e o índice 6, foram criados "buracos" `undefined`. Por isso que as notas 2, 3, 4 e 5 recebem o valor `undefined`.

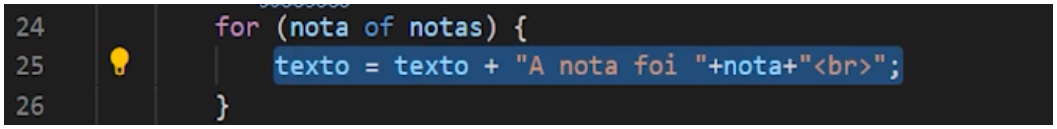
Figura 8 - Exibindo arrays com "buracos"



Assim, ao fazer essa atribuição ao índice 5, foram criados esses "buracos" com valor `undefined`, certo? O ideal, para inserir elementos em um array no final dele, é usar o método `post`, mas veremos isso na próxima videoaula.

Por enquanto, deixe como comentário no código: `notas[5]=10`, para continuarmos os testes. Eu gostaria, para terminar esse exemplo, de apresentar uma outra maneira de iterar sobre o array. Nós utilizamos o laço `for` padrão, que tem uma inicialização da variável `i` e tem a condição de enquanto `i` for menor que `notas.length` e incrementa o `i` no final. Mas, temos uma outra maneira de fazer o laço `for`, que é essa que aparece na Figura 9 entre as linhas 24 e 26. Esse `for`, ele faz o seguinte: para toda `nota` do array `notas`, o `for` vai executar esse comando da linha 25.

Figura 9 - Usando `for/of`



Dessa maneira, para toda `nota`, e seguindo a ordem do array `notas`, ele vai executar esse comando, expondo o texto `"A nota foi"`, e aparecerá o valor da nota. Tá certo? Veja que nesse exemplo perdemos o índice, teríamos como trabalhar isso, mas não o faremos nesse momento. Então, o que faremos é simplesmente exibir qual foi a nota.

Se você voltar ao exemplo e colocar, novamente, as três notas: 7, 8 e 9, verá que será exibido "A nota foi 7", "A nota foi 8" e "A nota foi 9". Assim, conseguimos iterar sobre todos os elementos do array.

Terminamos por aqui esta videoaula. Nas próximas videoaulas, você conhecerá diversos métodos que JavaScript disponibiliza para trabalhar com arrays. Até lá!!!