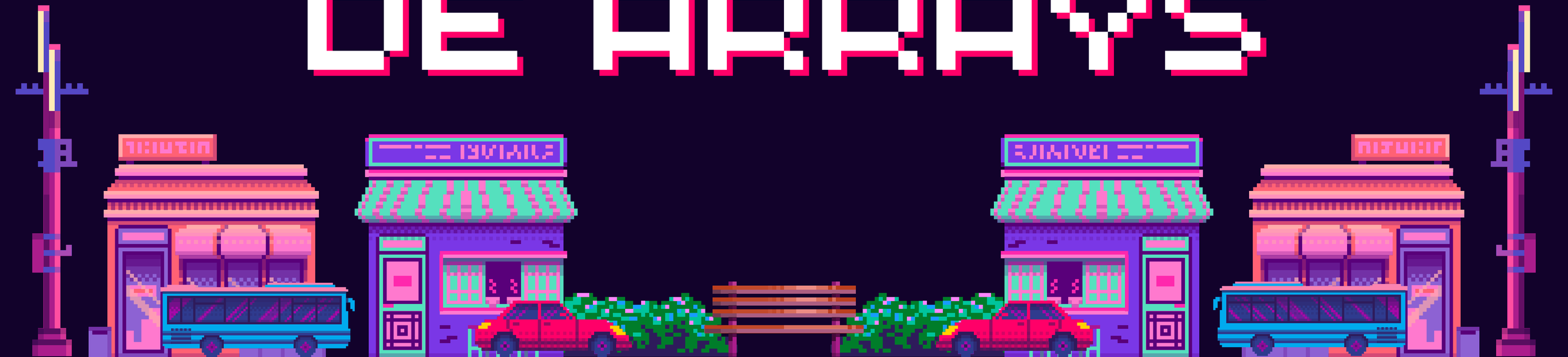
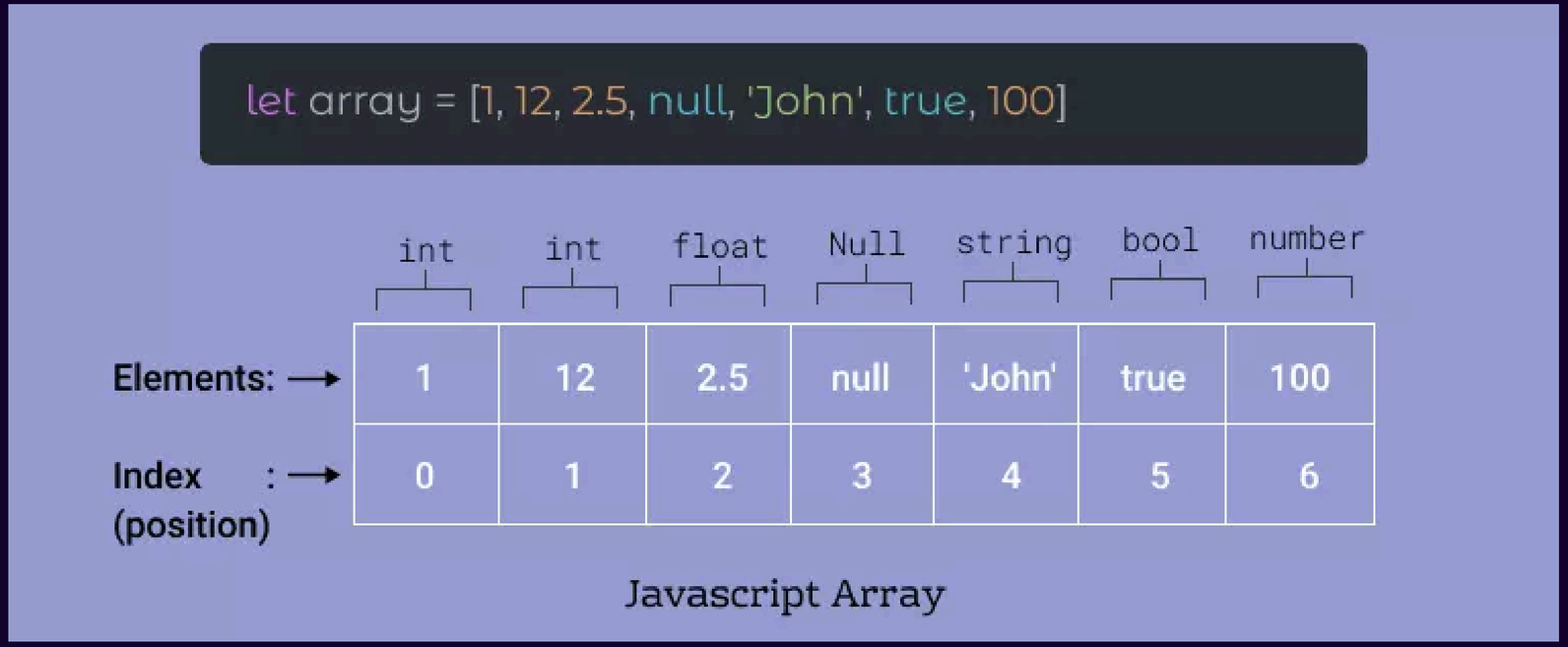

HELLO
WORLD

MANIPULAÇÃO DE ARRAYS



O QUE É?

O objeto Array do JavaScript é um objeto global usado na construção de 'arrays': objetos de alto nível semelhantes a listas.



CRIANDO UMA LISTA:

Usar um literal de array é a maneira mais fácil de criar um array JavaScript.
Sintaxe:

```
const array_name = [item1, item2, ...];
```

Espaços e quebras de linha não são importantes. Uma declaração pode abranger várias linhas:

```
const cars = [
  "Saab",
  "Volvo",
  "BMW"
];
```

Você também pode criar uma matriz e então fornecer os elementos:

```
const cars = [];
cars[0]= "Saab";
cars[1]= "Volvo";
cars[2]= "BMW";
```

USANDO A PALAVRA-CHAVE JAVASCRIPT NEW

O exemplo a seguir também cria um Array e atribui valores a ele:

```
const cars = new Array("Saab", "Volvo", "BMW");
```

Os dois exemplos acima fazem exatamente a mesma coisa.

Não há necessidade de usar `new Array()`.

Para simplicidade, legibilidade e velocidade de execução, use o método literal de matriz.

RELAÇÃO ENTRE LENGTH E PROPRIEDADES NUMÉRICAS

As propriedades length e numéricas de um array Javascript são conectadas. Varios dos métodos javascript pré-definidos (por exemplo, join, slice, indexOf etc.) levam em conta o valor da propriedade length de um array quando eles são chamados. Outros métodos (por exemplo, push, splice etc.) também resultam em uma atualização na propriedade length do array.

```
var frutas = [];

frutas.push("banana", "maça", "pêssego");


console.log(frutas.length); // 3
```

ACESSANDO ELEMENTOS DE MATRIZ

Você acessa um elemento de matriz referindo-se ao número do índice :

```
const cars = ["Saab", "Volvo", "BMW"];
let car = cars[0];
```

Observação: os índices de matriz começam com 0.

[0] é o primeiro elemento. [1] é o segundo elemento.

ALTERANDO UM ELEMENTO DE MATRIZ

Esta declaração altera o valor do primeiro elemento em cars:

```
const cars = ["Saab", "Volvo", "BMW"];
cars[0] = "Opel";
```

CONVERTENDO UMA MATRIZ EM UMA STRING

O método JavaScript toString()converte uma matriz em uma sequência de valores de matriz (separados por vírgula).

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
```

Resultado:

Banana,Orange,Apple,Mango

ELEMENTOS DE MATRIZ DE LOOP

Uma maneira de percorrer um array é usando um forloop:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fLen = fruits.length;

let text = "<ul>";
for (let i = 0; i < fLen; i++) {
  text += "<li>" + fruits[i] + "</li>";
}
text += "</ul>";
```

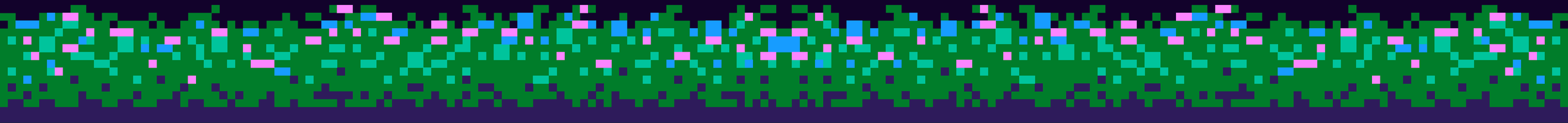
ADICIONANDO ELEMENTOS DE MATRIZ

A maneira mais fácil de adicionar um novo elemento a um array é usando o push()método:

```
const fruits = ["Banana", "Orange", "Apple"];
fruits.push("Lemon");  // Adds a new element (Lemon) to fruits
```

Novos elementos também podem ser adicionados a uma matriz usando a propriedade length:

```
const fruits = ["Banana", "Orange", "Apple"];
fruits[fruits.length] = "Lemon";  // Adds "Lemon" to fruits
```



MÉTODO JOIN()

O join()método também une todos os elementos do array em uma string.
Ele se comporta exatamente como toString(), mas além disso você pode especificar o separador:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.join(" * ");
```

Banana * Orange * Apple * Mango

MÉTODO POP()

O pop()método remove o último elemento de uma matriz:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.pop();
```

O método pop( ) retorna o valor que foi "retirado":

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fruit = fruits.pop();
```

MÉTODO PUSH()

O push()método adiciona um novo elemento a uma matriz (no final):

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.push("Kiwi");
```

O push()método retorna o novo comprimento da matriz:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let length = fruits.push("Kiwi");
```

MÉTODO SHIFT()

O shift()método remove o primeiro elemento da matriz e "desloca" todos os outros elementos para um índice mais baixo.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.shift();
```

O shift()método retorna o valor que foi "deslocado para fora":

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fruit = fruits.shift();
```

MÉTODO UNSHIFT()

O unshift() método adiciona um novo elemento a uma matriz (no início) e "desloca" elementos mais antigos:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.unshift("Lemon");
```

O unshift() método retorna o novo comprimento da matriz:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.unshift("Lemon");
```

MÉTODO SPLICE()

O splice()método pode ser usado para adicionar novos itens a uma matriz:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(2, 0, "Lemon", "Kiwi");
```

O primeiro parâmetro (2) define a posição onde novos elementos devem ser adicionados (emendados).
O segundo parâmetro (0) define quantos elementos devem ser removidos .
Os demais parâmetros ("Limão", "Kiwi") definem os novos elementos a serem adicionados .

MÉTODO SPLICE() PARA REMOVER ELEMENTOS

Com a configuração inteligente de parâmetros, você pode usar splice()para remover elementos sem deixar "buracos" na matriz:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(0, 1);
```

MÉTODO DELETE()

O unshift() método adiciona um novo elemento a uma matriz (no início) e "desloca" elementos mais antigos:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
delete fruits[0];
```

Aviso !

O uso `delete()` deixa `undefined` buracos na matriz.

Em vez disso, use pop() ou shift().

MÉTODO FLAT()

O flat() método cria uma nova matriz com elementos de submatriz concatenados a uma profundidade especificada

```
const myArr = [[1,2],[3,4],[5,6]];
const newArr = myArr.flat();
```

MÉTODO SLICE()

O slice() método divide um pedaço de uma matriz em uma nova matriz:

```
const fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];
const citrus = fruits.slice(1);
```

Observação

O `slice()` método cria uma nova matriz.

O `slice()` método não remove nenhum elemento do array de origem.

MÉTODO TOSTRING() AUTOMÁTICO

O JavaScript converte automaticamente uma matriz em uma string separada por vírgulas quando um valor primitivo é esperado.
Este é sempre o caso quando você tenta gerar uma matriz.
Esses dois exemplos produzirão o mesmo resultado:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
```

MÉTODO TOSTRING() AUTOMÁTICO

O JavaScript converte automaticamente uma matriz em uma string separada por vírgulas quando um valor primitivo é esperado.
Este é sempre o caso quando você tenta gerar uma matriz.
Esses dois exemplos produzirão o mesmo resultado:

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
```

Crie uma função que permita adicionar itens a uma lista de tarefas e exiba a lista atualizada toda vez que um novo item for adicionado.

GAME RULES

01

Escreva uma função chamada adicionarTarefa.

02

A função deve aceitar dois argumentos: um array que representa a lista de tarefas e uma string que representa a nova tarefa a ser adicionada.

03

Adicione a nova tarefa ao array.
Exiba a lista atualizada no no navegador

MEUS CONTATOS:

27 99500-7495

https://beacons.ai/prismatech

producaoprismatech@gmail.com

Avenida Jerônimo Monteiro 145, Vitória


THANK
YOU