

# Executar os processos de codificação, manutenção e documentação de aplicativos computacionais para desktop

---

SENAC PE

Thiago Nogueira

Instrutor de Educação Profissional

# Introdução ao JavaScript

# Introdução ao JavaScript

## Console

---

- O console é um painel que exibe mensagens importantes, como erros, para desenvolvedores. Grande parte do trabalho que o computador realiza com nosso código é invisível para nós por padrão.
- Uma ação, ou método, que está incorporada no objeto console é o método `.log()`. Quando escrevemos `console.log()`, o que colocamos dentro dos parênteses será impresso, ou registrado, no console.

```
// imprime o valor 5
```

```
console.log(5)
```

# Introdução ao JavaScript

## Comentários

---

- À medida que escrevemos JavaScript, podemos escrever comentários em nosso código que o computador irá ignorar enquanto nosso programa for executado. Esses comentários existem apenas para leitores humanos.
- Existem dois tipos de comentários de código em JavaScript:
  - Um *comentário de uma única linha* comentará uma única linha e será indicado por duas barras *//* antes dela.

```
// Esse trecho será ignorado  
console.log('imprimindo qualquer coisa...');
```

- Um *comentário de várias linhas* comentará várias linhas e será indicado com */\** para iniciar e *\*/* finalizar o comentário.

```
/*  
Tudo isso está sendo ignorado  
mesmo que tenha código escrito  
console.log(123);  
*/
```

# Tipos de dados

# Tipos de dados

## Resumo

---

- *Número* : Qualquer número, incluindo números com decimais: 4, 8, 1516, 23.42
- *BigInt* : Qualquer número maior que  $2^{53}-1$  ou menor que  $-(2^{53}-1)$ , com n anexado ao número: 1234567890123456n.
- *String* : Qualquer agrupamento de caracteres no seu teclado (letras, números, espaços, símbolos, etc.) entre aspas simples: ' ... ' ou aspas duplas " ... ".
- *Boolean* : Este tipo de dados possui apenas dois valores possíveis - true ou false.
- *Nulo* : Este tipo de dado representa a ausência intencional de um valor, e é representado pela palavra-chave null(sem aspas).
- *Indefinido* : este tipo de dados é indicado pela palavra-chave undefined(sem aspas). Também representa a ausência de um valor, embora tenha um uso diferente de null. undefinedsignifica que um determinado valor não existe.
- *Symbol* : um recurso mais recente da linguagem, os símbolos são identificadores exclusivos, úteis em codificações mais complexas. Não há necessidade de se preocupar com isso por enquanto.
- *Object*: Coleções de dados relacionados.

# Tipos de dados

## Operadores

---

- Um operador é um personagem que executa uma tarefa em nosso código. JavaScript possui vários *operadores aritméticos* integrados , que nos permitem realizar cálculos matemáticos em números. Incluem:
  - Soma +
  - Subtração -
  - Produto \*
  - Divisão /
  - Resto %

```
console.log(3 + 4); // Imprime 7  
console.log(5 - 1); // Imprime 4  
console.log(4 * 2); // Imprime 8  
console.log(9 / 3); // Imprime 3  
console.log(11 % 3); // Imprime 2  
console.log(12 % 3); // Imprime 0
```

# Tipos de dados

## Concetenção de Strings

---

- Operadores não servem apenas para números! Quando um **+** operador é usado em duas strings, ele anexa a string direita à string esquerda:

```
console.log('bubba' + 'loo'); // Imprime 'bubbaloo'  
console.log('wo' + 'ah'); // Imprime 'woah'
```



# Tipos de dados

## Propriedades

---

- Todos os tipos de dados têm acesso a propriedades específicas que são transmitidas a cada instância. Por exemplo, cada instância de string possui uma propriedade chamada length que armazena o número de caracteres dessa string. Você pode recuperar informações de propriedade anexando à string um ponto final e o nome da propriedade:

```
console.log('Hello'.length); // Imprime 5
```

- O `.` é outro operador! Chamamos isso de *operador ponto* `.`

# Tipos de dados

## Métodos

---

- Métodos são ações que podemos realizar. Os tipos de dados têm acesso a métodos específicos que nos permitem lidar com instâncias desses tipos de dados. JavaScript fornece vários métodos de string.
- Chamamos ou *usamos* esses métodos anexando uma instância com:
  - um ponto (o operador ponto)
  - o nome do método
  - abrindo e fechando parênteses
- Por exemplo `'string de exemplo'.methodName()`.

```
console.log('hello'.toUpperCase()); // Retorna a string maiúscula - 'HELLO'  
console.log('Hey'.startsWith('H')); // Verifica com qual inicial inicia a string - True
```

# Tipos de dados

## Objetos Integrados

---

- Além do **console**, existem outros objetos incorporados no JavaScript. Esses objetos "embutidos" estão cheios de funcionalidades úteis.
- Por exemplo, se você quiser realizar operações matemáticas mais complexas do que aritméticas, o JavaScript possui o objeto integrado **Math**.
- A grande vantagem dos objetos é que eles possuem métodos! Vamos chamar o método `.random()` do objeto embutido **Math**:

```
console.log(Math.random()); // Imprime um valor aleatório entre 0 e 1
```

- Para gerar um número aleatório entre 0 e 50, poderíamos multiplicar esse resultado por 50.

# Variáveis

# Variáveis

## Hoisting

---

- Em JavaScript, toda variável é “**elevada/içada**” (**hoisting**) até o topo do seu contexto de execução. Esse mecanismo move as variáveis para o topo do seu escopo antes da execução do código.

```
var exibeMensagem = function() {  
    mensagem = 'Teste';  
    console.log(mensagem);  
    var mensagem;  
}  
exibeMensagem(); // Imprime 'Teste'
```

# Variáveis

## var

---

- Vamos considerar o exemplo:

```
var myName = 'Maria';  
console.log(myName); // Output: Maria'
```

- var, abreviação de variável, é uma palavra-chave do JavaScript que cria, ou declara, uma nova variável.
- myName é o nome da variável. Escrever dessa forma é uma convenção padrão em JavaScript chamada *camelcase*. Nele, você agrupa palavras em uma só, a primeira palavra é em minúsculas, e cada palavra subsequente terá a primeira letra em maiúsculas. (por exemplo, *camelCaseTudo*).
- = é o operador de atribuição. Ele atribui o valor ('Maria') à variável (myName).
- Após a variável ser declarada, o valor da string 'Maria' é impresso no console referenciando o nome da variável: `console.log(myName)`.

# Variáveis

## let

- Foi pensando em trazer o escopo de bloco (tão conhecido em outras linguagens) que o ECMAScript 6 destinou-se a disponibilizar essa mesma flexibilidade (e uniformidade) para a linguagem.

```
var exibeMensagem = function() {  
    if(true) {  
        var escopoFuncao = 'Debi';  
        let escopoBloco = 'Loide';  
        console.log(escopoBloco); // Loide  
    }  
    console.log(escopoFuncao); // Debi  
    console.log(escopoBloco);  
}  
exibeMensagem(); // Imprime 'Loide', 'Debi' e dá um erro
```

- Veja que quando tentamos acessar uma variável que foi declarada através da palavra-chave **let** fora do seu escopo, o erro *Uncaught ReferenceError: escopoBloco is not defined* foi apresentado.

# Variáveis

## const

---

- A **const** palavra-chave também foi introduzida no ES6 e é uma abreviação da palavra constante. Assim como com **var** e **let** você pode armazenar qualquer valor em uma **const** variável.

```
const myName = 'Gilberto';  
console.log(myName); // Output: Gilberto
```

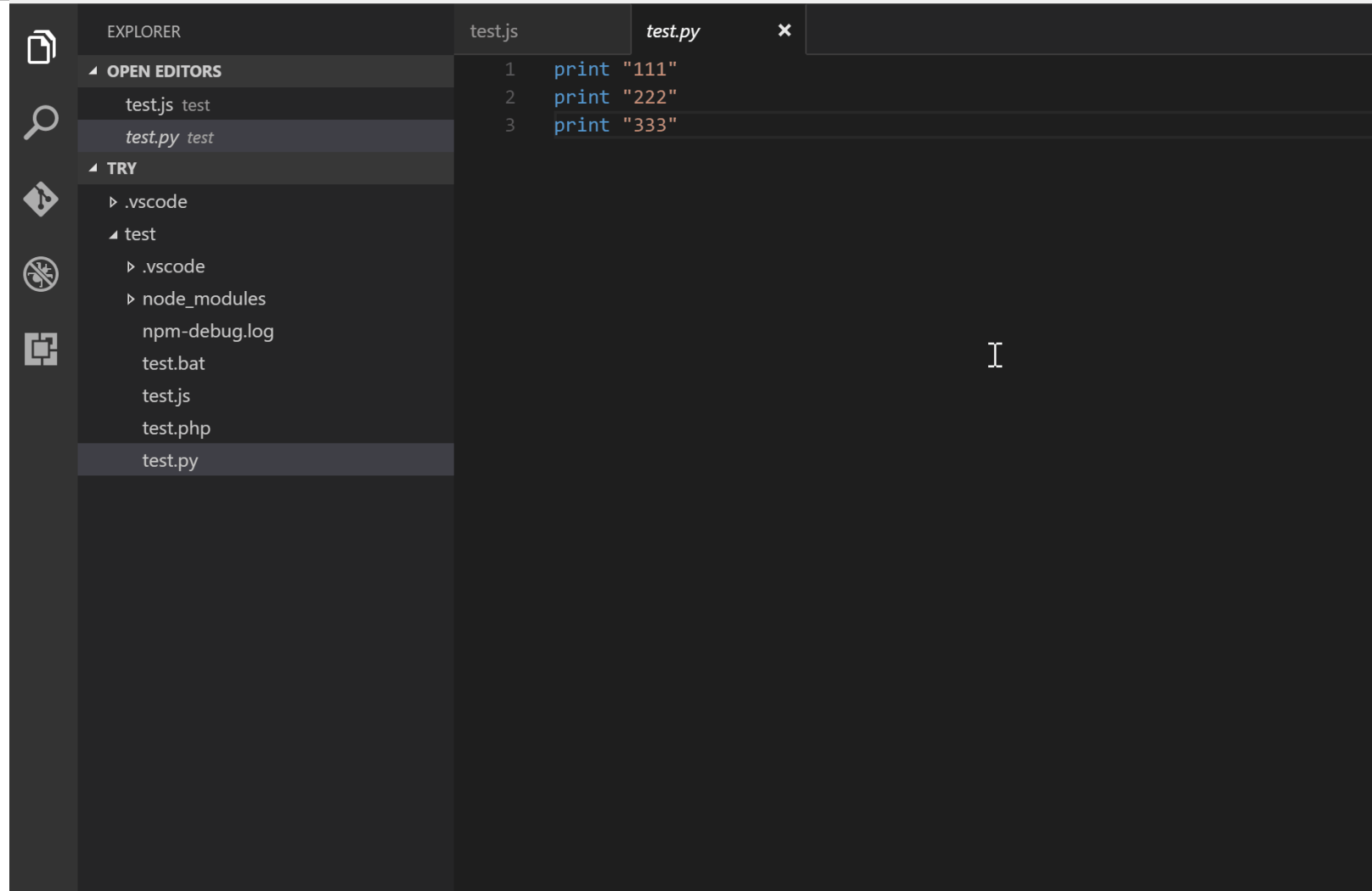
- No entanto, uma **const** variável não pode ser reatribuída porque é *constante*. Se você tentar reatribuir uma **const** variável, obterá um arquivo **TypeError**.
- Variáveis constantes *devem* receber um valor quando declaradas. Se você tentar declarar uma **const** variável sem valor, obterá um arquivo **SyntaxError**.



# Executando Projetos JavaScript

# Executando Instruções

- Instale o node.js
- Instale o VSCode
- Instale a extensão Code Runner
- Use o atalho Ctrl+Alt+N
- Pressione F1 e selecione Run Code



# Exercícios

# Exercícios

## Instruções

---


- Resolver os exercícios da aula 06
- Para cada exercício crie um arquivo .js diferente
- Enviar as respostas para qualquer repositório online
- Anexar o link do repositório com os exercícios resolvidos (e comentados)


*“Ensinar é impregnar  
de sentido o que  
fazemos a cada  
instante”*


*Paulo Freire*

**Obrigado!**

Thiago Nogueira

 [linkedin.com/tdn](https://www.linkedin.com/tdn)

 [thiago.nogueira@pe.senac.br](mailto:thiago.nogueira@pe.senac.br)

 (81) 9 9627-0419