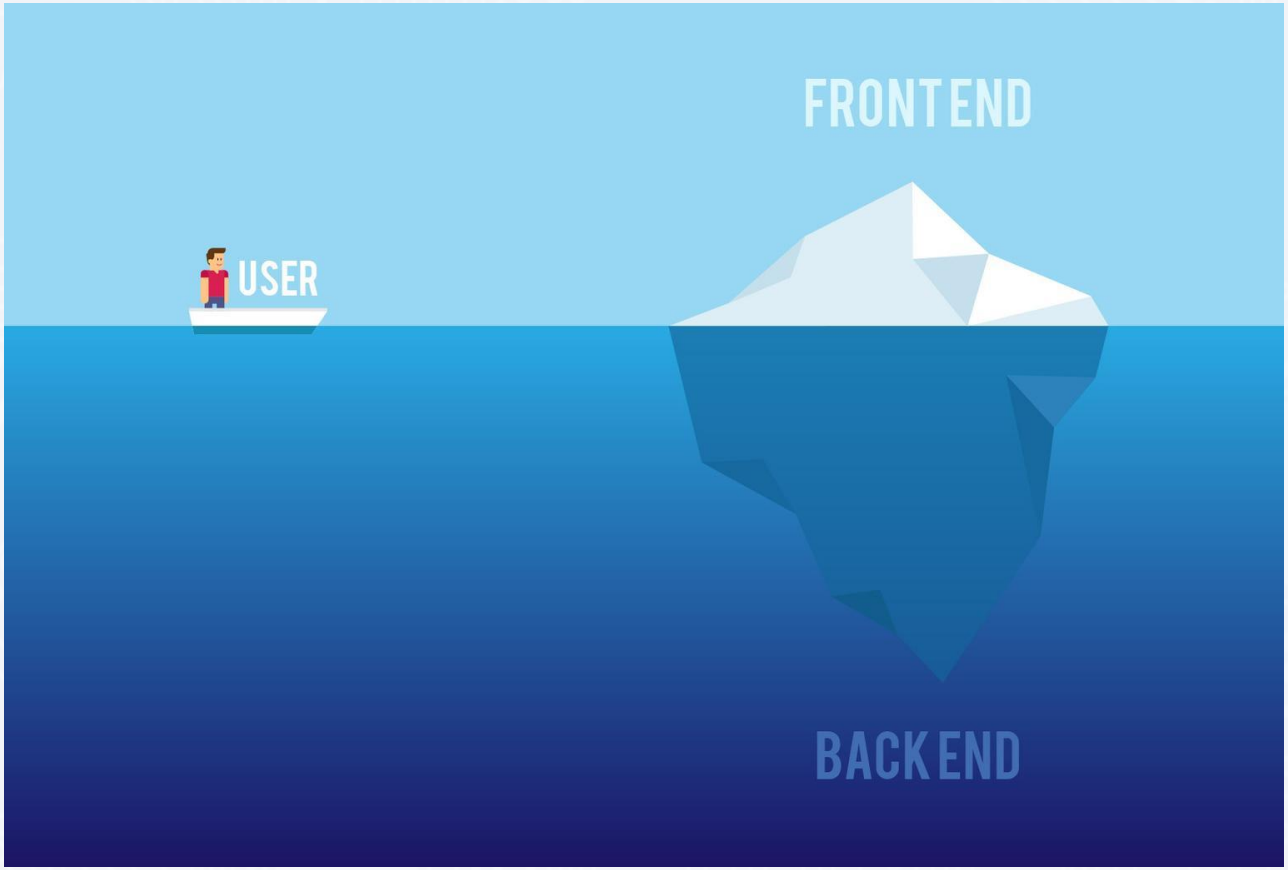
# PROGRAMAÇÃO EM MICROINFORMÁTICA

SANTO OLIANI JUNIOR

# INTRODUÇÃO

JavaScript é uma linguagem de programação interpretada estruturada, de script em alto nível com tipagem dinâmica fraca e multiparadigma. Juntamente com HTML e CSS, o JavaScript é uma das três principais tecnologias da World Wide Web.

# Front end / back end




https://joaopedrodias.com.br/blog/wp-content/uploads/2018/11/front-back-end.jpg

# Sintaxe

JavaScript é case-sensitive e usa o conjunto de caracteres Unicode.

No JavaScript, instruções são chamadas de declaração e são separadas por um ponto e vírgula (;)

# Sintaxe

Espaços, tabulação e uma nova linha são chamados de espaços em branco.

O código fonte dos scripts em JavaScript são lidos da esquerda para a direita e são convertidos em uma sequência de elementos de entrada como símbolos, caracteres de controle, terminadores de linha, comentários ou espaço em branco

# Comentários

```
// comentário de uma linha

/* isto é um comentário longo
   de múltiplas linhas.
 */

/* Você não pode, porém, /* aninhar
comentários */ SyntaxError */
```

# Variáveis

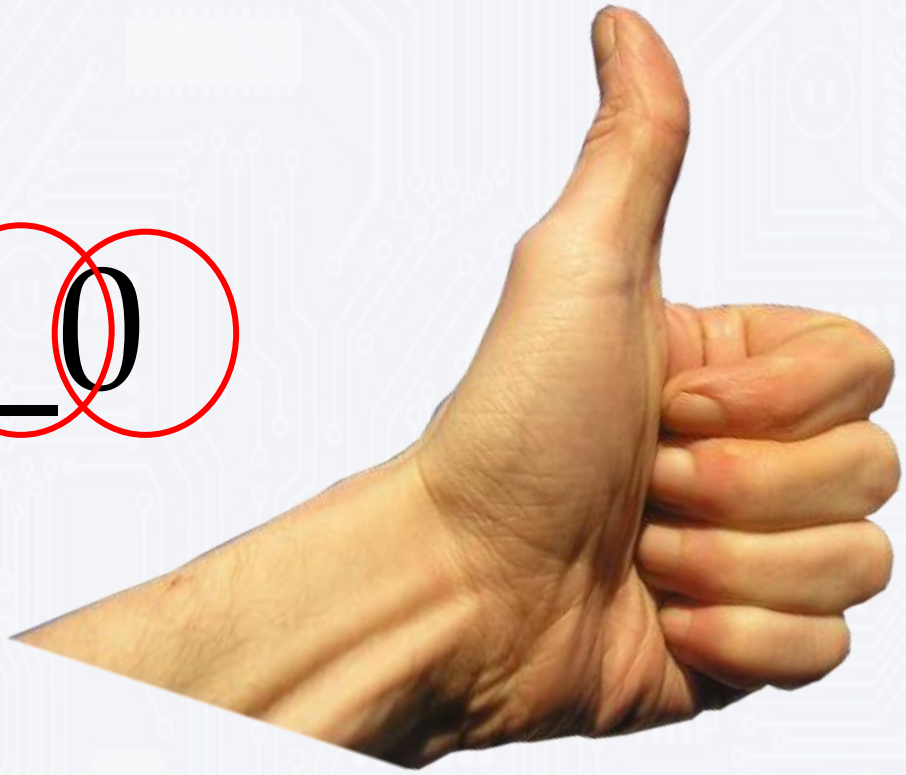
Você usa variáveis como nomes simbólicos para os valores em sua aplicação. O nome das variáveis, chamados de identificadores, obedecem determinadas regras.

# Variáveis

Um identificador JavaScript deve começar com uma letra, underline (_), ou cifrão ($); os caracteres subsequentes podem também ser números (0-9). Devido JavaScript ser case-sensitive, letras incluem caracteres de "A" a "Z" (maiúsculos) e caracteres de "a" a "z" (minúsculos).

# VARIÁVEIS

Variavel_01



# VARIÁVEIS

1variavel01

# Declaração de variáveis

Com a palavra chave var. Por exemplo, var x = 42. Esta sintaxe pode ser usada para declarar tanto variáveis locais como variáveis globais.

# Declaração de variáveis

Por simples adição de valor. Por exemplo, x = 42. Isso declara uma variável global.

Essa declaração gera um aviso de advertência no JavaScript. Você não deve usar essa variante

# Declaração de variáveis

Com a palavra chave let. Por exemplo, let y = 13.

Essa sintaxe pode ser usada para declarar uma variável local de escopo de bloco.

# Atribuição de valores

O sinal de = é atribuição de valores de variáveis.

Exemplo:

```
x = 10;
a = 5;
y = x + a;
z = x - a;
x = x + 10;
```

# OPERADORES ARITMÉTICOS

| Operador | Descrição | Exemplo |
|---|---|---|
| + | Adição | A + B = 30 |
| – | Subtração | A – B = -10 |
| * | Multiplicação | A * B = 200 |
| / | Divisão | B / A = 2 |
| % | Módulo (resto da divisão) | B % A = 0 |
| ++ | Incremento | A++ = 11 |
| — | Decremento | A– = 9 |

# OPERADORES DE ATRIBUIÇÃO

| Operador | Descrição | Exemplo |
|---|---|---|
| = | Atribuição | C = A + B atribui o valor de A + B em C |
| += | Atribuição de soma | C += A equivale a C = C + A |
| -= | Atribuição de subtração | C -= A equivale a C = C – A |
| *= | Atribuição de multiplicação | C *= A equivale a C = C * A |
| /= | Atribuição de divisão | C /= A equivale a C = C / A |
| %= | Atribuição de resto | C %= A equivale a C = C % A |

# == versus ===

== (Igual a ou Equal to) - é usado para comparação entre duas variáveis, independentemente do tipo de dados da variável.

Para x= 10 temos que :

x == 8 -> retorna false

x == 10 -> retorna true

x == "10" -> retorna true


# == versus ===

=== (Valor e Tipo igual) - é usado para a comparação entre duas variáveis, mas isso irá verificar o tipo estrito, o que significa que ele irá verificar o tipo de dados e comparar dois valores.

Para x= 10 temos que :

x === 8 -> retorna false

x === 10 -> retorna true

x === "10" -> retorna false

# Classificando variáveis

Uma variável declarada usando a declaração var ou let sem especificar o valor inicial tem o valor  undefined.

Uma tentativa de acessar uma variável não declarada resultará no lançamento de uma exceção ReferenceError:

# Classificando variáveis

```
var a;
console.log("O valor de a é " + a);
// saída "O valor de a é undefined"

console.log("O valor de b é " + b);
// executa uma exception de erro de referência (ReferenceError)
```

# Classificando variáveis

Você pode usar undefined para determinar se uma variável tem um valor. No código a seguir, não é atribuído um valor de entrada na variável e a declaração if será avaliada como verdadeira (true).

# Classificando variáveis

```
var input;
if(input === undefined){
  facaIsto();
} else {
  facaAquilo();
}
```

# Escopo de variável

Quando você declara uma váriavel fora de qualquer função, ela é chamada de variável global, porque está disponível para qualquer outro código no documento atual. Quando você declara uma variável dentro de uma função, é chamada de variável local, pois ela está disponível somente dentro dessa função.

# Escopo de variável

```
if (true) {
  var x = 5;
}
console.log(x);

// 5
```

```
if (true) {
  let y = 5;
}
console.log(y);

// ReferenceError: y não está definido
```

# Constantes

Você pode criar uma constante apenas de leitura por meio da palavra-chave const. A sintaxe de um identificador de uma constante é semelhante ao identificador de uma variável: deve começar com uma letra, sublinhado ou cifrão e pode conter caractere alfabético, numérico ou sublinhado.

# Constantes

```
const PI = 3.14;
```

Uma constante não pode alterar seu valor por meio de uma atribuição ou ser declarada novamente enquanto o script está em execução. Deve ser inicializada com um valor.

As regras de escopo para as constantes são as mesmas para as váriaveis let de escopo de bloco. Se a palavra-chave const for omitida, presume-se que o identificador represente uma variável.

# Constantes

Você não pode declarar uma constante com o mesmo nome de uma função ou variável que estão no mesmo escopo. Por exemplo:

```
// Isto irá causar um  erro
function f() {};
const f = 5;

// Isto também irá causar um erro.
function f() {
  const g = 5;
  var g;

  //declarações
}
```

# Tipos de dados

Seis tipos de dados são os chamados primitivos:

- Boolean. true e false.
- null. Uma palavra-chave que indica valor nulo.

  Devido JavaScript ser case-sensitive, null não é o mesmo que Null, NULL, ou ainda outra variação.
- undefined. Uma propriedade superior cujo valor é indefinido.
- Number. 42 ou 3.14159.
- String. "isso é um texto"
- Symbol (novo em ECMAScript 6). Um tipo de dado cuja as instâncias são únicas e imutáveis.
- e Object

# Conversão de tipos de dados

JavaScript é uma linguagem dinamicamente tipada. Isso significa que você não precisa especificar o tipo de dado de uma variável quando declará-la, e tipos de dados são convertidos automaticamente conforme a necessidade durante a execução do script.

# Conversão de tipos de dados

Então, por exemplo, você pode definir uma variável da seguinte forma:

```
var answer = 42;
```

E depois, você pode atribuir uma string para a mesma variável, por exemplo:

```
answer = "Obrigado pelos peixes...";
```

# Conversão de tipos de dados

Devido JavaScript ser dinamicamente tipado, essa declaração não gera uma mensagem de erro.

Em expressões envolvendo valores numérico e string com o operador +, JavaScript converte valores numérico para strings.

# Conversão de tipos de dados

Por exemplo, considere a seguinte declaração:

```
x = "A resposta é " + 42 // "A resposta é 42"
y = 42 + " é a resposta" // "42 é a resposta"
```

Nas declarações envolvendo outros operadores, JavaScript não converte valores numérico para strings. Por exemplo:

```
"37" - 7 // 30
"37" + 7 // "377“
```

# Descobrir o tipo de dado

typeof (variável);

# Convertendo strings para números

No caso de um valor que representa um número está armazenado na memória como uma string, existem métodos para a conversão.

parseInt()

parseFloat()

# Convertendo strings para números

Uma método alternativo de conversão de um número em forma de string é com o operador + (operador soma):

"1.1" + "1.1" = "1.11.1"

(+"1.1") + (+"1.1") = 2.2

// Nota: Os parênteses foram usados para deixar mais legível o código, ele não é requirido.

# Array

```
var estados = ["Rio de Janeiro", "São
  Paulo", "Bahia"];

Console.log estados [1];
Console.log estados;
```

# Array

```
var estados = ["Rio de Janeiro", "São
  Paulo", "Bahia"];

Estados [1] = “SP”;
console.log (estados);
```

# Variáveis / Array

```
var titulo = "Programador";
var quantidadeDeVagas = 5;
var vagaAtiva = true;

var vaga = [“Programador”, 5, true];
console.log (vaga.lenght); //imprime o
  tamanho do array
```

# Variáveis / Array

```
var telefones = [
    '(11) 98899 - 8787',
    '(22) 3455 - 8819',
    '(91) 95620 - 0000'
];
```

# EXERCÍCIO

1) Escreva um algoritmo em JS para calcular e escrever a área do retângulo.

2) Faça um algoritmo em JS que escreva a idade de uma pessoa expressa em dias. Obs: considerar ano com 365 dias e mês com 30 redondo.

# EXERCÍCIO

3) Escreva um algoritmo em JS para calcular e escrever o valor de um novo salário do funcionário.

A jornada de trabalho semanal de um funcionário é de 40 horas. O funcionário que trabalhar mais de 40 horas receberá hora extra, cujo cálculo é o valor da hora regular com um acréscimo de 50%. Escreva um algoritmo que leia o número de horas trabalhadas em um mês, o salário por hora e escreva o salário total do funcionário, que deverá ser acrescido das horas extras, caso tenham sido trabalhadas (considere que o mês possua 4 semanas exatas).

# FUNÇÕES

```
function square() {
   resultado = numero * numero;
}
```

# OBRIGADO

 santo.oliani@fatec.sp.gov.br