

Javascript

Programa de atração de formação de talentos
Bradesco / Visionaire

Aula 01 – Primeiros passos

Mark Joselli
mark.joselli@pucpr.br



Javascript

- Padrão ECMA. Sintaxe baseada no Java.
- Lançado em 1995 no Netscape Navigator.
- Melhorias significativas a partir de 2005 (JS 6).
- Interpretado
- Tipagem dinâmica
- Funções de primeira classe
- Orientação a objeto baseada em protótipos

A large yellow square containing the letters 'JS' in a bold, black, sans-serif font, representing the JavaScript logo.

Declarando variáveis e constantes

Variáveis e constantes são declaradas através dos comandos **let** ou **const**

```
let numero = 27.5;
```

```
const nome = "Vinícius";
```

```
let pessoaJuridica = false;
```


Declarando variáveis e constantes

Variáveis e constantes são declaradas através dos comandos **let** ou **const**

```
let numero = 27.5;
```

```
const nome = "Vinícius";
```

```
let pessoaJuridica = false;
```



O ponto-e-vírgula ao final da linha é opcional



IMPORTANTE

Declarando variáveis e constantes

O javascript também possui a palavra **var**, mas ela deve ser **evitada** pois:

- Seu escopo é a função onde está, e não o comando
- Variáveis criadas com var serão consideradas declaradas desde o início da função e não no ponto onde foram criadas (*hoisting*)
- Não a utilize em código novo!

Números

Em Javascript toda variável numérica aceita números negativos e casas decimais

```
let a = 15;  
let b = -2.5;  
  
const soma = a + b * 4;
```


$$f(x) = g(x)$$

Secant
Lines

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

$$f(x) = \lim_{h \rightarrow 0} (x+h)$$

$$= \lim_{h \rightarrow 0} x^2 + 2xh + h^2$$

$$= \lim_{h \rightarrow 0} 2xh + h^2$$

$$f(x+h) - g(x)$$

$$= \lim_{h \rightarrow 0} h(2x + h)$$

Aritmética

- Os operadores +, - * e / podem ser usados para fazer operações
- A precedência dos operadores é respeitada
- Há também o operador %, para a operação de resto
- Você pode combina-los ao sinal de = para realizar a operação sobre a própria variável
- Parênteses também podem ser usados

Aritmética

```
let somaUm = 10;
```

```
somaUm++;
```

```
let negativo = -somaUm;
```

```
let expressao = (20 - idade) * somaUm;
```


Números especiais

Há valores especiais que são considerados números:

+Infinity e -Infinity: Representando o infinito

NaN: Representa uma indeterminação numérica

```
const indeterminado = 0 / 0;  
const infinito = 1 / 0;
```

Texto

Textos podem ser criados utilizando aspas simples, duplas ou inversa:

```
let tipo = 'Pontificia';  
let funcao = `Universidade Católica`;  
let esfera = "Paraná";
```

As aspas inversas permitem o uso de expressões através da sintaxe `${variável}` e múltiplas linhas:

```
const nome =  
  `${tipo} ${funcao}`  
  do `${esfera}`;
```

Texto

Texto também pode ser concatenado através dos operadores + e +=

A \ pode ser usada para inserir caracteres especiais (como \n) ou \" para as aspas

```
const nome = tipo + " " + funcao +  
              "\n do " + esfera;
```

Quando possível, dê preferência a forma com crase. Hoje a concatenação com operador + é mais usada em código legado.

Texto

Há uma série de funções interessantes para lidar com texto:

- `parseInt` e `parseFloat`: Converte o texto em inteiro ou float. Caso a conversão não seja possível, retorna NaN
- `.toFixed(decimais)`: Converte um número em um texto, com a quantidade especificada de casas decimais. Faz arredondamento se necessário.
- `.toLowerCase()`, `.toUpperCase()`: Converte o texto em letras maiúsculas e minúsculas
- `.length`: Retorna o tamanho do texto

Booleanos e comparações

Representam valores **verdadeiro** (**true**) e **falso** (**false**)

- São o resultado de comparações e expressões lógicas

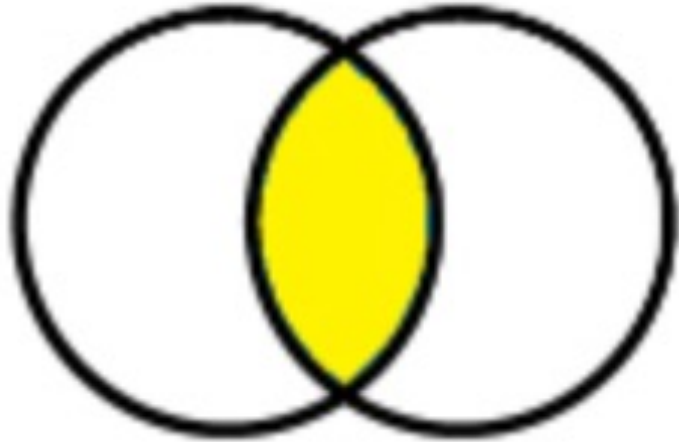
```
let quinzeAnos = idade == 15;  
let maior = idade >= 18;
```

- Outros operadores são >, <, <=, == e !=

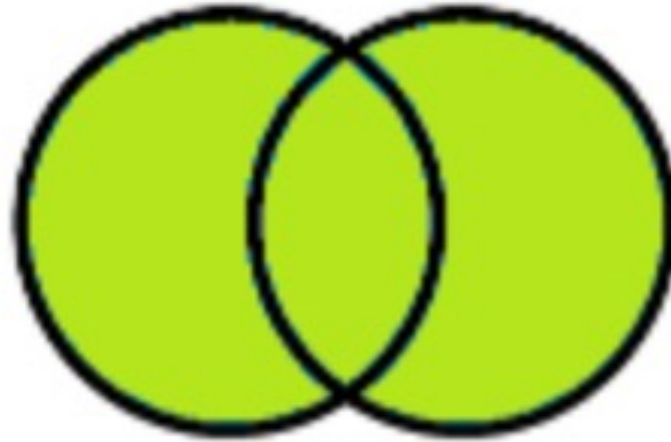
Booleans e comparações

- Texto também pode ser comparado
- O JavaScript utilizará o valor Unicode de cada caracter
- Isso aproxima da ordem alfabética porém:
 - Letras maiúsculas sempre são “menores” do que minúsculas
 - Qualquer símbolo será comparado
- O valor NaN é o único que nunca será igual a ele mesmo. Para testar se um valor é NaN, use a função **isNaN**

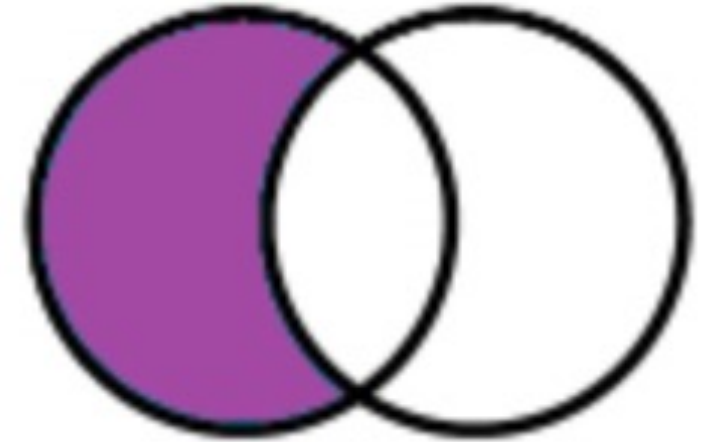
AND



OR



NOT



Operadores lógicos

- Permitem associar expressões booleanas. São eles:

E: **&&** Ou: **||** Não: **!**

- Eles tem precedência sobre qualquer operação. Em seguida estão as comparações e depois o resto

Operador ternário

- Permite a escolha de um valor com base em uma expressão

```
const adulto = idade >= 18 ? "maior" : "menor";
```

The diagram illustrates the components of the ternary operator expression `idade >= 18 ? "maior" : "menor"`. It uses three colored brackets to group the parts: an orange bracket under `idade >= 18`, a green bracket under `"maior"`, and a red bracket under `"menor"`. Below each bracket is a corresponding colored rounded rectangle with a label: "Condição" (orange), "Valor se verdadeiro" (green), and "Valor se falso" (red).

Condição	Valor se verdadeiro	Valor se falso
<code>idade >= 18</code>	<code>"maior"</code>	<code>"menor"</code>

Nulidade

- O Javascript possui dois valores que representam o nada, são os valores **undefined** e **null**
- Uma variável declarada, mas não inicializada, terá o valor **undefined**.
- O **null** representa que a variável foi inicializada, mas não contém valor.

Coersão

O Javascript tentará converter tipos de dados automaticamente quase sempre.

```
console.log(8 * null) // → 0
console.log("5" - 1) // → 4
console.log("5" + 1) // → 51
console.log("five" * 2) // → NaN
console.log(false == 0) // → true
```



CUIDADO

Dica: As comparações com === e !== incluem o tipo de dado nos testes. Dê preferência a elas no lugar de == e !=



Vamos
praticar

Teste o valor da expressão

```
console.log(("b" + 'a' + + 'z' + `a`).toLowerCase())
```

Você consegue explicar o resultado?

Falsy/truthy values

- Uma série de valores podem ser considerados falsos / verdadeiros:
 - **Falso**: false, null, undefined, 0, NaN, ""
 - **Verdadeiro**: true, {}, [], "foo", new Date(), números != 0, incluindo Infinity

Curtos circuitos

- O operador `||` converte o lado esquerdo da expressão em um booleano e retorna o valor do lado direito se, e somente se, este for considerado falso
- O operador `&&` faz o mesmo com valores verdadeiros
- Isso permite escrever expressões como:

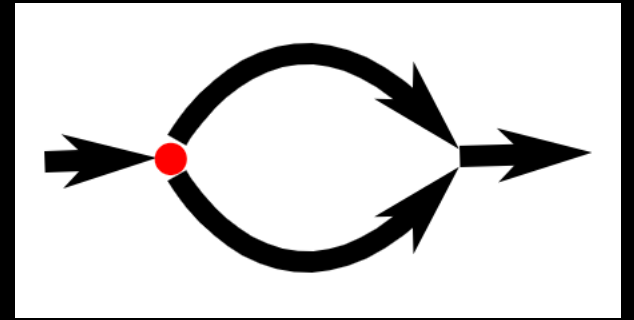
```
console.log(null || "usuario") // → usuario  
console.log("Agnes" || "usuario") // → Agnes
```

Lendo e imprimindo valores

- Utilize o comando `prompt` para solicitar um valor. O valor retornado será sempre um texto
- Para imprimir valores, utilize o comando `console.log`

```
const nome = prompt("Nome?");  
console.log("Seu nome é ", nome);
```

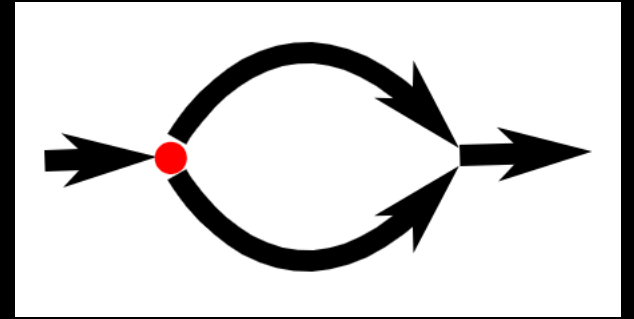
Condicionais: if



- O comando if é utilizado para condicionais
- Ele pode conter uma instrução else:

```
if (idade < 13) {  
    console.log("Criança");  
} else if (idade < 18) {  
    console.log("Adolescente");  
} else {  
    console.log("Adulto")  
}
```

Condicionais: switch



- O comando switch permite testar uma expressão contra um conjunto de valores
- Cada bloco deve ser terminado com **break**
- Ele pode conter uma instrução **default**:

```
const dia = 0;
switch (dia) {
  case 0:
    console.log("Domingo");
    break;
  case 1:
    console.log("Sabado");
    break;
  default:
    console.log("Dia útil");
    break;
}
```

Repetição: *while* e *do...while*



Permitem repetir comandos enquanto a condição for verdadeira

```
while (condição) {  
    //Comandos  
}
```

```
do {  
    //Comandos  
} while (condição);
```

Repetição: for



O comando for é útil para a repetição com contadores

```
for (inicialização; condição; operação) {  
    //Comandos  
}
```

Exemplo:

```
console.log("1 indiozinho");  
for (let x = 2; x < 10; x++) {  
    console.log(`${x} indiozinhos`);  
}
```


break e continue



O comando **break** interrompe uma repetição imediatamente

O comando **continue** faz com que o fluxo seja imediatamente desviado para o início do condicional

Vamos
praticar



Atividades

1. Crie as variáveis peso e altura leia e as inicialize com seu peso e sua altura. Calcule o valor do IMC ($\text{peso} / \text{altura}^2$). Associe o valor verdadeiro a variável obeso caso o valor do IMC seja maior ou igual a 30. Imprima o texto, substituindo os valores em *itálico* pelas respectivas variáveis: O valor do IMC para o peso de *peso* quilos e altura *metros* é de *imc*.

2. Leia um número e aplique sobre ele a conjectura de Collatz. Ela diz que uma sequência pode ser feita com base na seguinte regra:

- Se o número n for par, o próximo é $n / 2$
- Se for ímpar é $3n+1$
- A sequência termina em 1

3. Chico tem 1,50 metro e cresce 2 centímetros por ano, enquanto Zé tem 1,10 metro e cresce 3 centímetros por ano. Construa um algoritmo que calcule e imprima as alturas de Chico e Zé até que Zé seja maior que Chico