

Curso: Front-End do Bem



AULA 24

100% Gratuita | Baixou... Estudou | Sem pedir e-mail

- Operadores de Comparação
- 16 Exemplos de Operadores de Comparação (Strings e Numbers)
- Sugestões de Prática para essa aula
 - Pergunta do Amigo Internauta:
Igor de São Paulo - SP
- O que vamos aprender na aula 25



Operadores de Comparação

Igor, Igor, Igor..... O que são os tais Operadores de Comparação?
Aqui teremos 2 respostas... a primeira resposta será da fonte MDN e a segunda resposta será o que eu entendo sobre Operadores de Comparação.

Fonte MDN Responde essa: Um operador de comparação... compara seus operandos e retorna um valor lógico(Boolean). (FONTE MDN)

Fonte Entendimento do Igor: Pessoal vocês se recordam quando estávamos estudando sobre atribuições a uma variável, nós vimos que igual dentro do JavaScript(JS) não é simbolizado por apenas um sinal de igual (=) e que nós veríamos como o igual seria simbolizado no decorrer do curso.... Eis que esse momento chegou... Bora se apresentarem ai:

// Pessoal esse é o duplo sinal de igual (==) que representa igual dentro do Javascript(JS), duplo sinal de igual (==) que representa igual dentro do Javascript(JS) esse aqui é o Pessoal.

Agora que nós já sabemos o que são os Operadores de Comparação, veremos alguns exemplos, tanto com Strings quanto com Numbers... pois assim conseguiremos fixar melhor esse conceito.

E bora ver o primeiro exemplo de Operadores de Comparação (Usando Strings).....

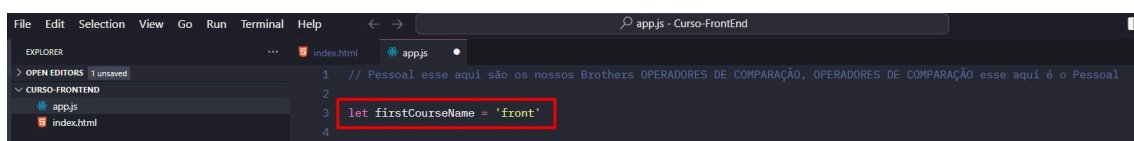


Operadores de Comparação

Ah, antes disso..... dentro do nosso nobre arquivo app.js, na linha 1 criaremos um comentário: `// Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal,`

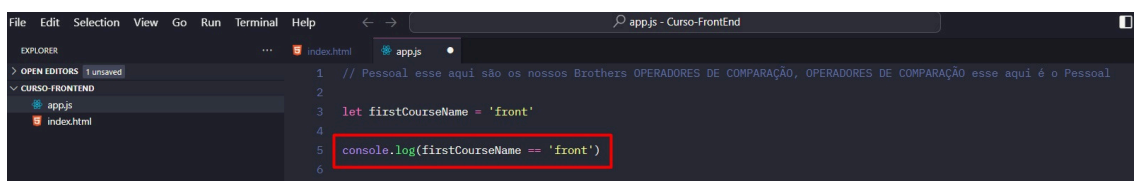
Então logo abaixo do comentário: `// Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal,` declararemos uma `let firstCourseName` que recebe `'front'`. Ficando assim:

```
let firstCourseName = 'front'
```



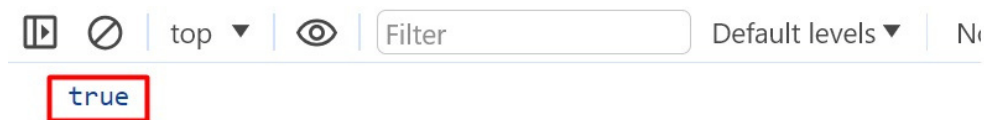
Adicionaremos um `console.log` passando a `firstCourseName` que é igual (`==`) a `'front'`. Ficando assim:

```
console.log(firstCourseName == 'front')
```



Operadores de Comparação

E ao salvarmos o arquivo app.js com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que true está sendo mostrado no console.



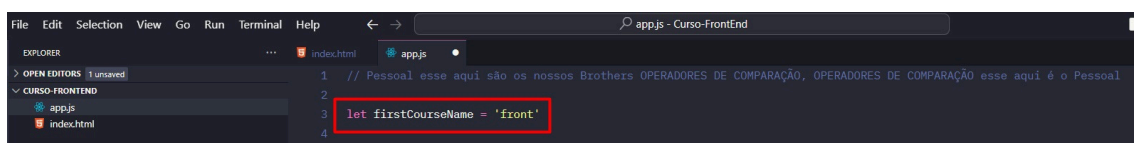
E porque True apareceu no exemplo acima?

Porque o duplo sinal de igual (==) fez uma verificação marota se o valor da `let firstCourseName` é igual `== 'front'`, como o valor da expressão `firstCourseName == 'front'` foi comparado que sim, então foi retornado true.

Operadores de Comparação

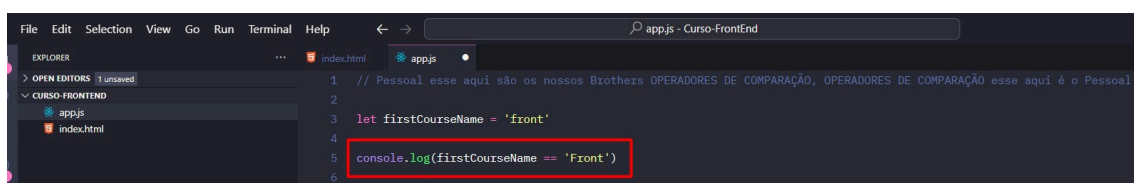
No nosso segundo exemplo de Operadores de Comparação (Strings), manteremos sem nenhuma alteração a nossa `let firstCourseName = 'front'`. Ficando assim:

```
let firstCourseName = 'front'
```



Adicionaremos um `console.log` passando a `firstCourseName` que é igual (`==`) a `'Front'` só que com o `'F'` em maiúsculo. Ficando assim:

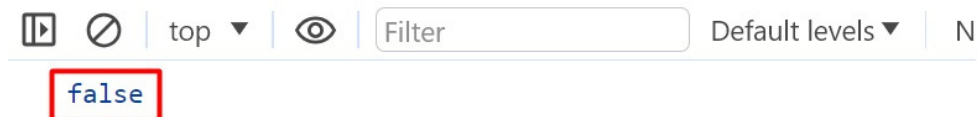
```
console.log(firstCourseName == 'Front')
```



E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `false` está sendo mostrado no console.



Operadores de Comparação



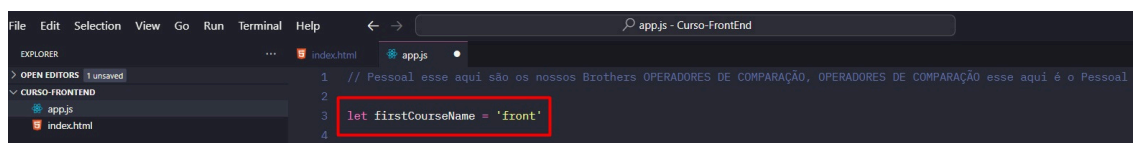
Igor, Igor, Igor, Igor, Igor, Igor.... Socorre a gente aqui, please.... Por que retornou false, sendo que as palavras são iguais?

Porque essa string que está em `let firstCourseName = 'front'`, é diferente da string 'Front' que está dentro do nosso `console.log`, e no nosso exemplo como a primeira letra de uma das palavras é maiúscula no caso a letra 'F' dentro do `console.log`, e essa alteração mínima é mais do que o suficiente para que o JavaScript(JS) faça com que essas strings sejam diferentes.

Operadores de Comparação

No nosso terceiro exemplo de Operadores de Comparação (Strings), manteremos sem nenhuma alteração a nossa `let firstCourseName = 'front'`. Ficando assim:

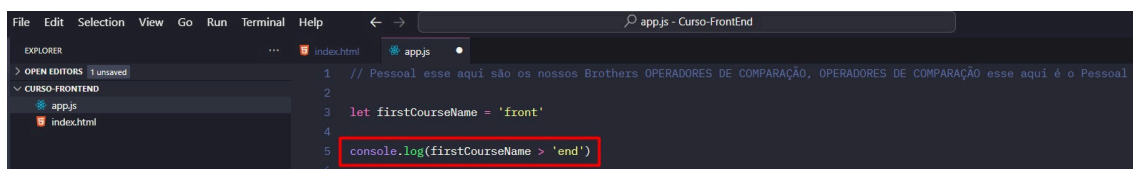
```
let firstCourseName = 'front'
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let firstCourseName = 'front'
4
```

Adicionaremos um `console.log` passando a `firstCourseName` que será maior > que 'end' com o 'e' em minúsculo. Ficando assim:

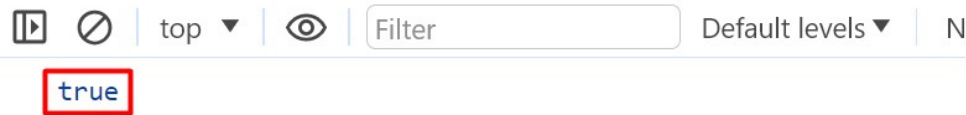
```
console.log(firstCourseName > 'end')
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let firstCourseName = 'front'
4
5 console.log(firstCourseName > 'end')
6
```

E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.

Operadores de Comparação



Igor, Igor, Igor, Igor, Igor..... eita que o negócio não entrou na nossa cabeça, como essa comparação é feita usando o sinal de maior > que. O que aconteceu aqui entre a string 'front' e a string 'end'. Pode esclarecer isso pra gente?

Claro, só ser for agora... peço a gentileza que isso aqui seja bem compreendido, pois aqui tem um detalhe que vai fazer a diferença.

O detalhe é o seguinte: Aqui no nosso exemplo ele está nos dizendo que 'front' é maior que 'end'... pois bem: no nosso alfabeto essa primeira letra(f) de 'front' da let firstCourseName = 'front' ela vem depois da letra 'e' de 'end'... então no nosso exemplo 'front' é maior que 'end'. Podemos entender com base nisso que as últimas letras do alfabeto são maiores que as primeiras letras do alfabeto.

O que esse detalhe acima quer dizer: Vamos pegar as 26 letras do alfabeto e colocaremos aqui embaixo (todas em minúsculas).

a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z.

Onde:



Operadores de Comparação

- a - Primeiro Valor
- b - Segundo Valor
- c - Terceiro Valor
- d - Quarto Valor
- e - Quinto Valor
- f - Sexto Valor
- g - Sétimo Valor
- h - Oitavo Valor
- i - Nono Valor
- j - Décimo Valor
- k - Décimo Primeiro Valor
- l - Décimo Segundo Valor
- m - Décimo Terceiro Valor
- n - Décimo Quarto Valor
- o - Décimo Quinto Valor
- p - Décimo Sexto Valor
- q - Décimo Sétimo Valor
- r - Décimo Oitavo Valor
- s - Décimo Nono Valor
- t - Vigésimo Valor
- u - Vigésimo Primeiro Valor
- v - Vigésimo Segundo Valor
- w - Vigésimo Terceiro Valor
- x - Vigésimo Quarto Valor
- y - Vigésimo Quinto Valor
- z - Vigésimo Sexto Valor



/igor-rebolla

Operadores de Comparação

Agora nós sabemos que as últimas letras do nosso alfabeto valem mais do que as primeiras. Chegamos a conclusão que:

a letra (string) f – é o nosso Sexto Valor

a letra (string) e – é o nosso Quinto Valor

f – é o nosso Sexto Valor > e – é o nosso Quinto Valor

Logo a string 'f' que faz parte da string 'front' é maior > que a string 'e' que faz parte da string 'end'.

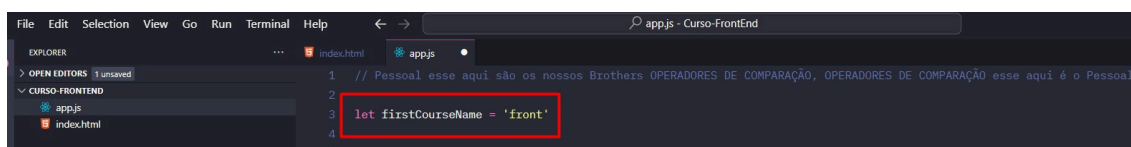


/igor-rebolla

Operadores de Comparação

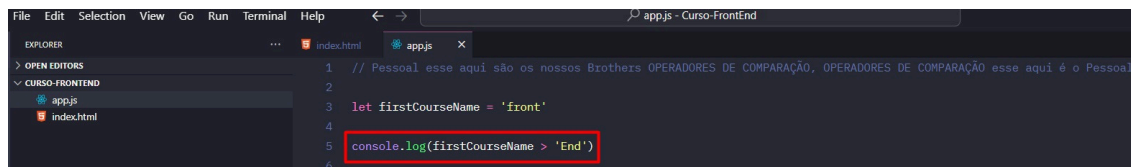
No nosso quarto exemplo de Operadores de Comparação (Strings), manteremos sem nenhuma alteração a nossa `let firstCourseName = 'front'`. Ficando assim:

```
let firstCourseName = 'front'
```

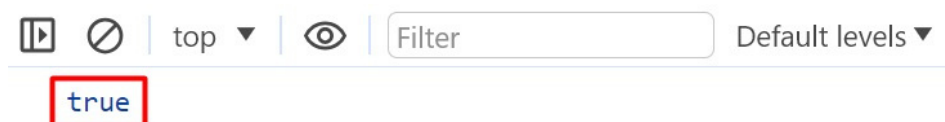


Adicionaremos um `console.log` passando a `firstCourseName` que será maior > que 'End' com o 'E' em maiúsculo. Ficando assim:

```
console.log(firstCourseName > 'End')
```



E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.



Operadores de Comparação

Igor, Igor, Igor, Igor, Igor..... eita que o negócio não entrou na nossa cabeça (parte 2), nós entendemos como funciona comparação entre 'strings' com minúsculas. O que aconteceu entre a string 'front' e a string 'End' agora com o 'E' em maiúsculo. Pode esclarecer isso pra gente?

Claro, só ser for agora... peço a gentileza que isso aqui também seja bem compreendido, pois aqui tem um novo detalhe que vai fazer a diferença.

E por que o TRUE acima foi exibido? Por que nesse exemplo não somente a letra 'f' minúscula da string 'front' na let firstCourseName é maior que a letra 'E' maiúscula, ela a letra 'f' minúscula é maior do que qualquer letra maiúscula, no caso aqui o (E) de 'End'.

O que esse detalhe acima quer dizer: Pegaremos agora 2 alfabetos com as 26 letras do alfabeto (cada) e colocaremos aqui embaixo (o primeiro alfabeto com as letras em minúsculas e o segundo alfabeto com as letras em maiúsculas).

Primeiro alfabeto (Letras minúsculas)

a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z.

Segundo alfabeto (Letras maiúsculas)

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z.

Onde



Operadores de Comparação

a - Primeiro Valor
b - Segundo Valor
c - Terceiro Valor
d - Quarto Valor
e - Quinto Valor
f - Sexto Valor
g - Sétimo Valor
h - Oitavo Valor
i - Nono Valor
j - Décimo Valor
k - Décimo Primeiro Valor
l - Décimo Segundo Valor
m - Décimo Terceiro Valor
n - Décimo Quarto Valor
o - Décimo Quinto Valor
p - Décimo Sexto Valor
q - Décimo Sétimo Valor
r - Décimo Oitavo Valor
s - Décimo Nono Valor
t - Vigésimo Valor
u - Vigésimo Primeiro Valor
v - Vigésimo Segundo Valor
w - Vigésimo Terceiro Valor
x - Vigésimo Quarto Valor
y - Vigésimo Quinto Valor
z - Vigésimo Sexto Valor

A - Primeiro Valor
B - Segundo Valor
C - Terceiro Valor
D - Quarto Valor
E - Quinto Valor
F - Sexto Valor
G - Sétimo Valor
H - Oitavo Valor
I - Nono Valor
J - Décimo Valor
K - Décimo Primeiro Valor
L - Décimo Segundo Valor
M - Décimo Terceiro Valor
N - Décimo Quarto Valor
O - Décimo Quinto Valor
P - Décimo Sexto Valor
Q - Décimo Sétimo Valor
R - Décimo Oitavo Valor
S - Décimo Nono Valor
T - Vigésimo Valor
U - Vigésimo Primeiro Valor
V - Vigésimo Segundo Valor
W - Vigésimo Terceiro Valor
X - Vigésimo Quarto Valor
Y - Vigésimo Quinto Valor
Z - Vigésimo Sexto Valor



/igor-rebolla

Operadores de Comparação

Notem que os valores tanto do alfabeto das letras minúsculas quanto do alfabeto das letras maiúsculas, são idênticos... só que para o JavaScript(JS)... qualquer letra minúscula independente de qual seja, será sempre maior do que qualquer letra maiúscula.

Chegamos a conclusão que:

a letra (string) f – por ser minúscula vai ser maior > que
a letra (string) E – por ser maiúscula independente do valor que ela tenha

f – por ser minúscula vai ser maior > que E – por ser maiúscula

Birutão isso né? Só um 'cadinho'... agora no nosso próximo exemplo misturaremos tudo pra aprendermos juntos e assim fixarmos melhor esse assunto.

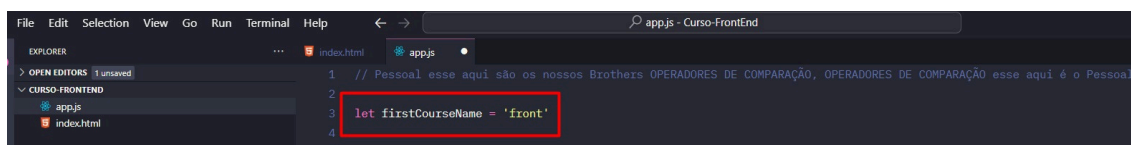
Bora ver isso na próxima página.....



Operadores de Comparação

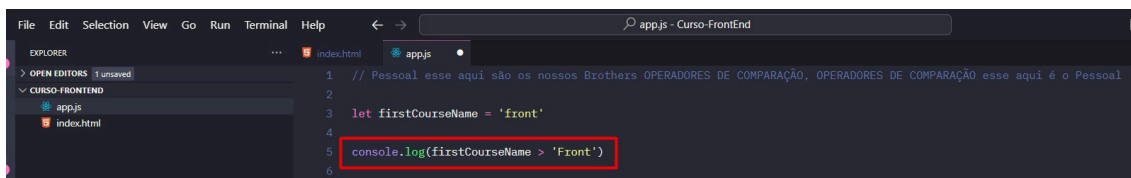
No nosso quinto exemplo de Operadores de Comparação (Strings), manteremos sem nenhuma alteração a nossa `let firstCourseName = 'front'`. Ficando assim:

```
let firstCourseName = 'front'
```

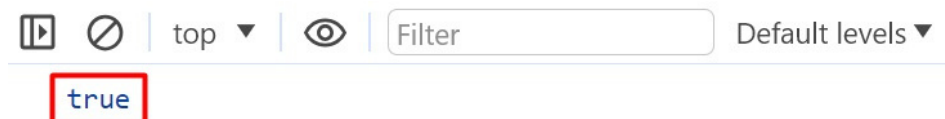


Adicionaremos um `console.log` passando a `firstCourseName` que será maior > que 'Front' com o 'F' em maiúsculo. Ficando assim:

```
console.log(firstCourseName > 'Front')
```



E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.



Operadores de Comparação

E porque true apareceu no exemplo acima?

Porque como vimos que qualquer letra minúscula será sempre maior que qualquer letra maiúscula.

Pois bem, aqui temos uma letra minúscula 'f' será sempre maior > que uma letra maiúscula 'F'. Por isso true foi retornado.

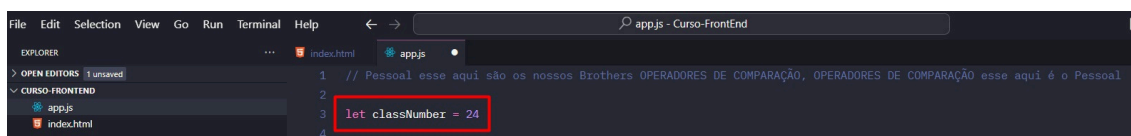


Operadores de Comparação

Agora que nós vimos as comparações entre STRINGS, chegou o grande momento de vermos comparações com NÚMEROS.

Então bora ver o primeiro exemplo sobre Números.... Yuppppppi!!!!

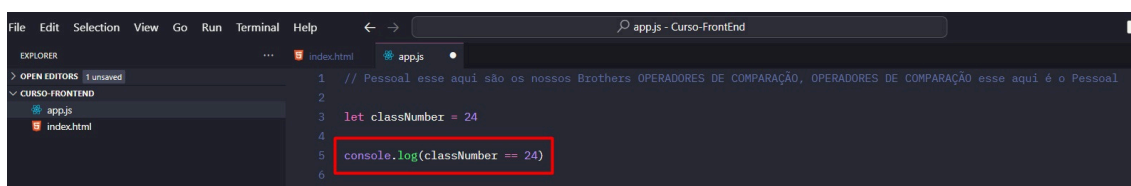
Como nós já definimos na linha 1 do arquivo app.js o nosso comentário // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal, daremos um enter maroto e na linha 3, criaremos uma let classNumber que receberá 24. Ficando assim:



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um console.log passando a classNumber que é igual (==) utilizando dois sinais de igual a 24. Ficando assim:

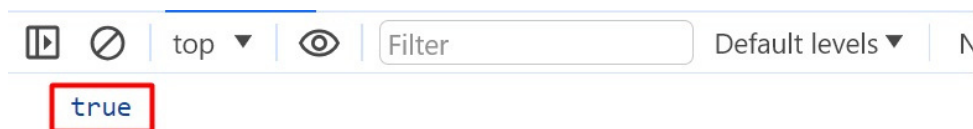
console.log(classNumber == 24)



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber == 24)
6
```

Operadores de Comparação

E ao salvarmos o arquivo app.js com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que true está sendo mostrado no console.



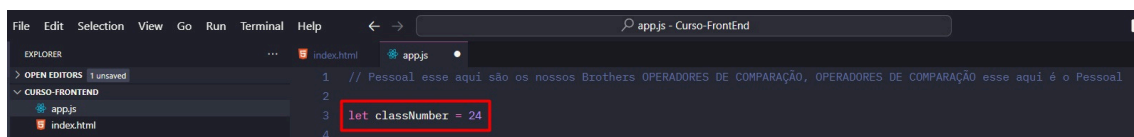
E porque true apareceu no exemplo acima?

Porque assim como vimos com o primeiro exemplo dessa aula em strings, o duplo sinal de igual (==) fez uma verificação marota se o valor da let classNumber é igual == a 24, se for igual..... essa expressão classNumber == 24 retornará true.

Operadores de Comparação

No nosso segundo exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

```
let classNumber = 24
```

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project named 'CURSO-FRONTEND' with files 'app.js' and 'index.html'. The main editor window is open to 'app.js' and contains four lines of code. The third line, `let classNumber = 24`, is highlighted with a red rectangular box. The code above it is a multi-line comment in Portuguese.

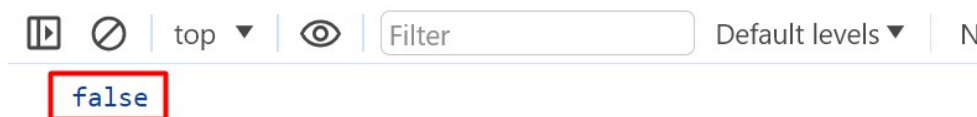
Adicionaremos um `console.log` passando a `classNumber` que é igual `==` a 19. Ficando assim:

```
console.log(classNumber == 19)
```

A screenshot of the Visual Studio Code editor interface, similar to the previous one. The Explorer sidebar shows the same project structure. The main editor window shows six lines of code. The fifth line, `console.log(classNumber == 19)`, is highlighted with a red rectangular box. The code above it includes the `let classNumber = 24` declaration.

Operadores de Comparação

E ao salvarmos o arquivo app.js com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que false está sendo mostrado no console.



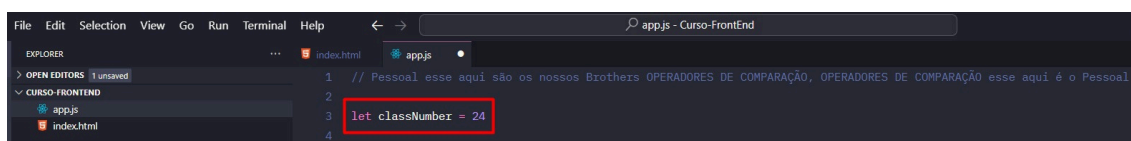
E porque false apareceu no exemplo acima?

O nosso amigão duplo sinal de igual (==) fez uma verificação marota se o valor da let classNumber é igual a 24, se for igual como o valor da expressão classNumber == 19..... e aqui foi comparado que não, então foi retornado false.

Operadores de Comparação

No nosso terceiro exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24`. Ficando assim:

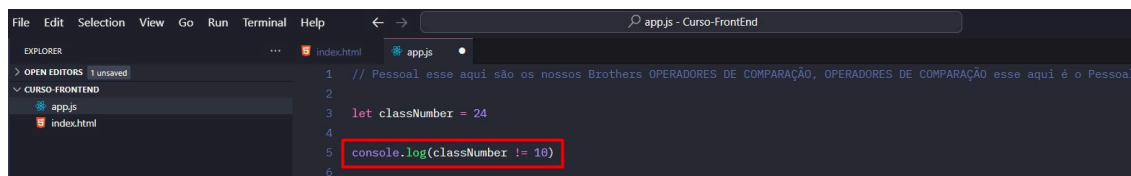
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que não será igual, representada pelo comparador `!=` seguido do número 10. Ficando assim:

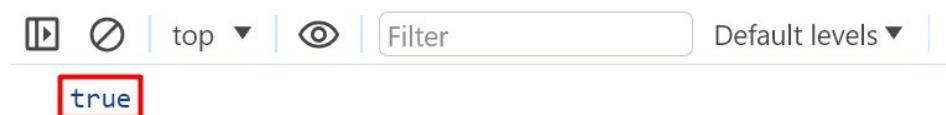
```
console.log(classNumber != 10)
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber != 10)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.

Operadores de Comparação



E porque true apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber não é igual a 10.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade(true)?

E a resposta recebida, é de que sim, isso é verdade(true).

É feita outra pergunta:

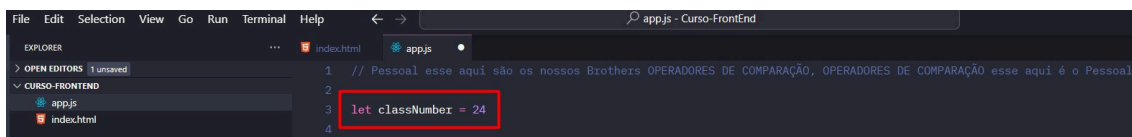
E porque isso é verdade(true)?

Porque 24 é não é igual a 10 ou seja: 24 != 10

Operadores de Comparação

No nosso quarto exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

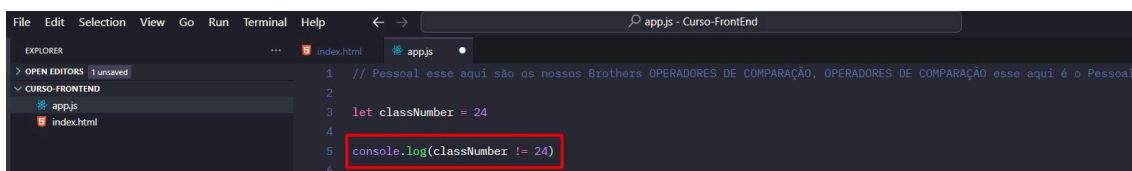
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que não será igual, representada pelo comparador `!=` seguido do número 24. Ficando assim:

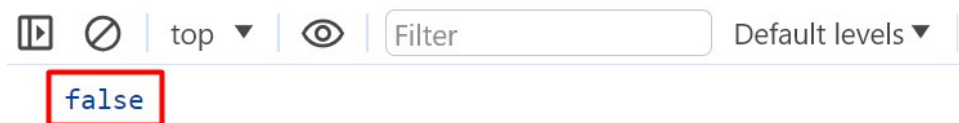
```
console.log(classNumber != 24)
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber != 24)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `false` está sendo mostrado no console.

Operadores de Comparação



E porque false apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber não é igual a 24.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade (true)?

E a resposta recebida, é de que não, isso é falso(false).

É feita outra pergunta:

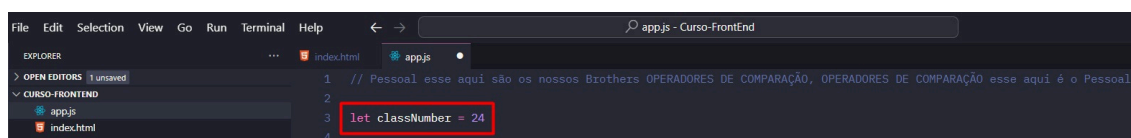
E porque isso é falso(false)?

Porque 24 é igual a 24 ou seja: 24 == 24

Operadores de Comparação

No nosso quinto exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

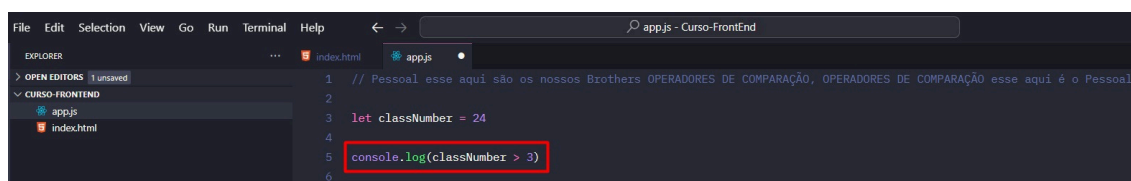
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que é maior `>` que 3, representada pelo comparador `>` seguido do número 3. Ficando assim:

```
console.log(classNumber > 3)
```

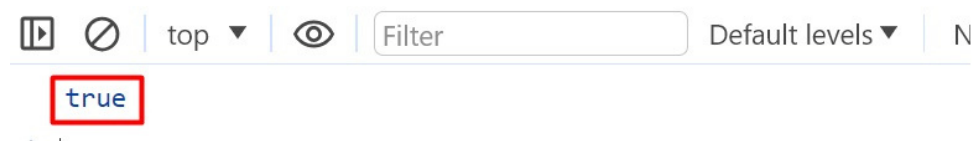


```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber > 3)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.



Operadores de Comparação



E porque true apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: `classNumber` é maior `>` que `3`.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade (`true`)?

E a resposta recebida, é de sim, isso é verdade(`true`).

É feita outra pergunta:

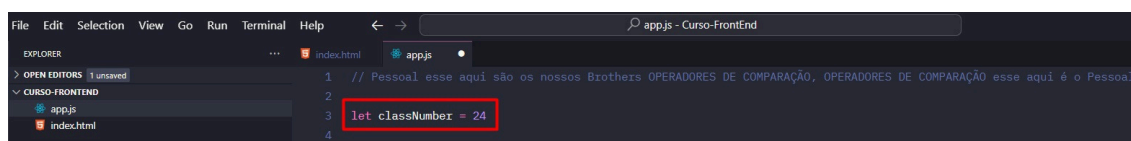
E porque isso é verdade(`true`)?

Porque `24` é maior que `3` ou seja: `24 > 3`

Operadores de Comparação

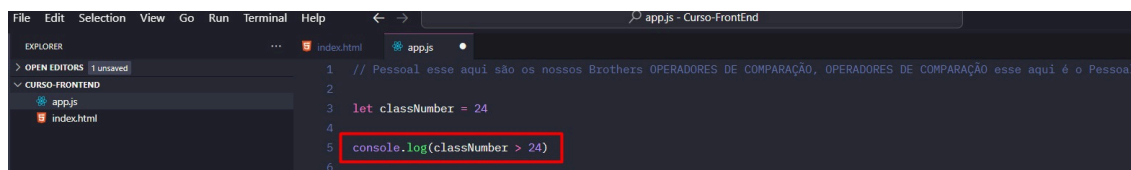
No nosso sexto exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

```
let classNumber = 24
```



Adicionaremos um `console.log` passando a `classNumber` que é maior `>` que 24, representada pelo comparador `>` seguido do número 24. Ficando assim:

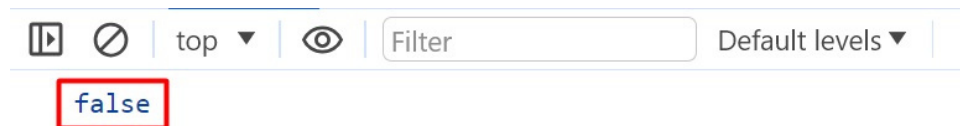
```
console.log(classNumber > 24)
```



E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que `false` está sendo mostrado no console.



Operadores de Comparação



E porque false apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber é maior > que 24.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade (true)?

E a resposta recebida, é de que não, isso é falso(false).

É feita outra pergunta:

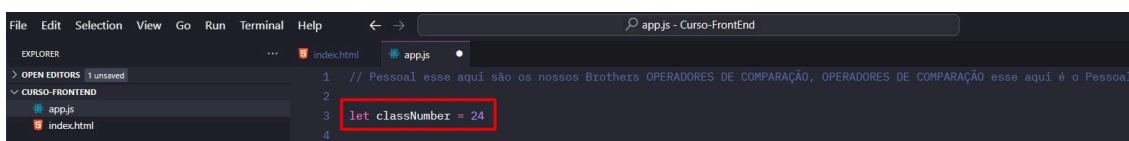
E porque isso é falso(false)?

Porque 24 não é maior que 24, e sim igual a 24 ou seja: 24 == 24

Operadores de Comparação

No nosso sétimo exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

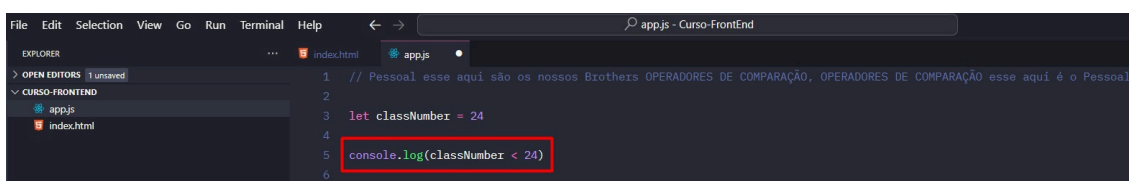
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que é menor `<` que 24, representada pelo comparador `<` seguido do número 24. Ficando assim:

```
console.log(classNumber < 24)
```

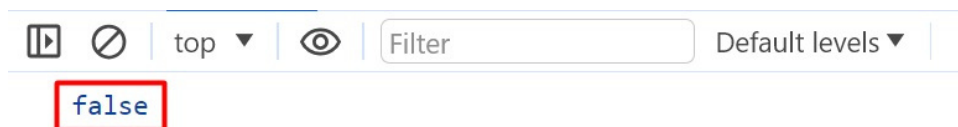


```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber < 24)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `false` está sendo mostrado no console.



Operadores de Comparação



E porque false apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber é menor < que 24.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade (true)?

E a resposta recebida, é de que não, isso é falso(false).

É feita outra pergunta:

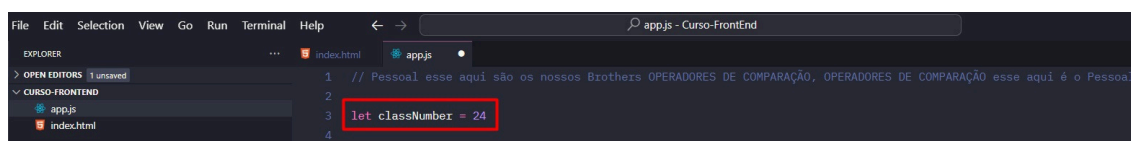
E porque isso é falso(false)?

Porque 24 não é menor que 24, e sim igual a 24 ou seja: 24 == 24

Operadores de Comparação

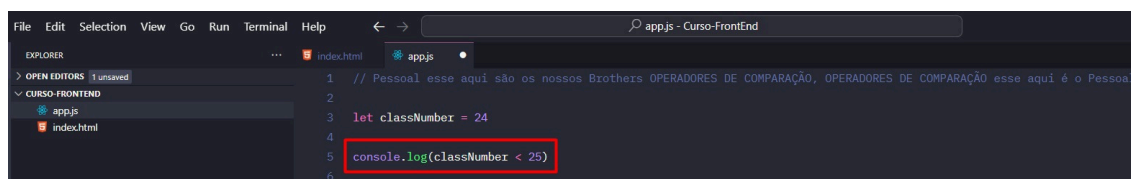
No nosso oitavo exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

```
let classNumber = 24
```



Adicionaremos um `console.log` passando a `classNumber` que é menor `<` que 25, representada pelo comparador `<` seguido do número 25. Ficando assim:

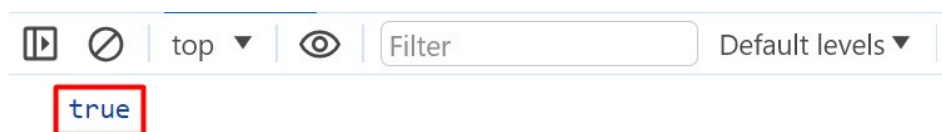
```
console.log(classNumber < 25)
```



E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que `false` está sendo mostrado no console.



Operadores de Comparação



E porque true apareceu no exemplo acima?

Porque o jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber é menor < que 25.

Depois que fizemos essa leitura, é perguntado:

Se isso é verdade (true)?

E a resposta recebida, é de que sim, isso é verdade(true).

É feita outra pergunta:

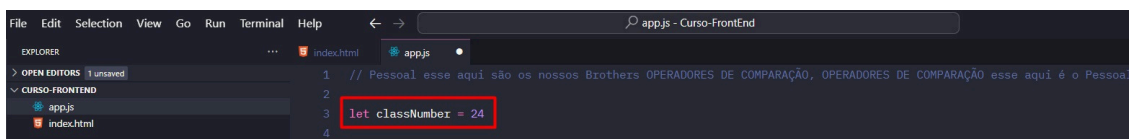
E porque isso é verdade(true)?

Porque 24 é menor que 25 ou seja: $24 < 25$

Operadores de Comparação

No nosso nono exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

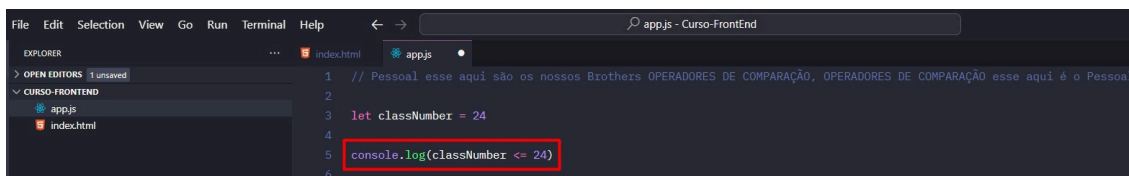
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que é menor ou = (`<=`) que 24, representada pelos comparadores `<=` seguido do número 24. Ficando assim:

```
console.log(classNumber <= 24)
```

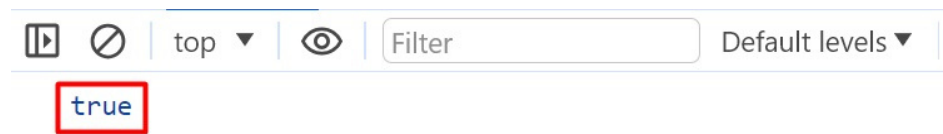


```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber <= 24)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.



Operadores de Comparação



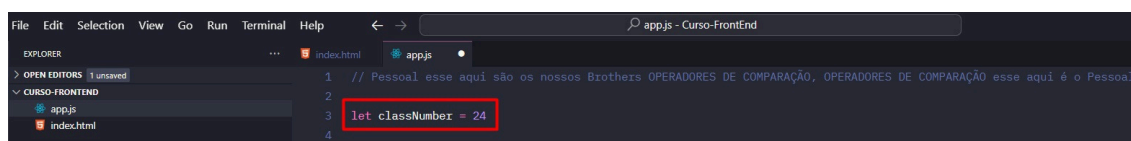
E porque true apareceu no exemplo acima?

Porque ao inserirmos que a classNumber é menor ou igual (\leq) a 24. Então ao adicionarmos um sinal de = logo após esse sinal de menor que, esse sinal vira um menor ou igual que. E esse OU marotinho aqui faz uma diferença enorme, porque 24 não é menor que 24, mas como esse sinal \leq verifica se o 24 é menor ou igual a 24, essa expressão `classNumber \leq 24` é verdadeira(true).

Operadores de Comparação

No nosso décimo exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

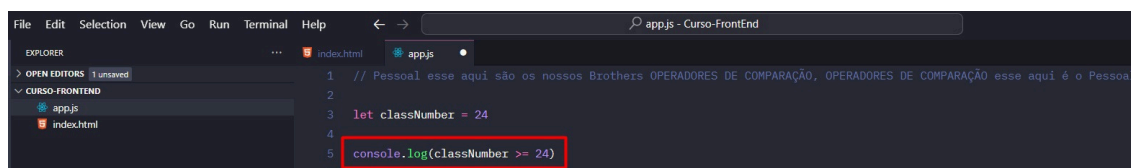
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que é maior ou igual (`>=`) que 24, representada pelos comparadores `>=` seguido do número 24. Ficando assim:

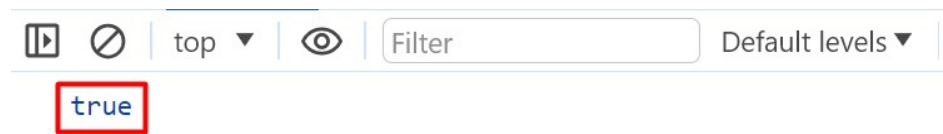
```
console.log(classNumber >= 24)
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber >= 24)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.

Operadores de Comparação



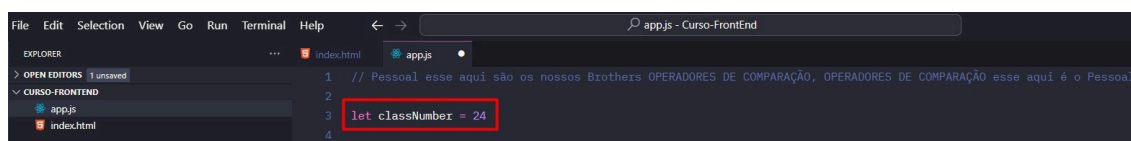
E porque true apareceu no exemplo acima?

Porque ao inserirmos que a classNumber é maior ou igual (\geq) a 24. Então ao adicionarmos um sinal de = logo após esse sinal de maior que, esse sinal vira um maior ou igual que. E esse OU marotinho faz uma diferença enorme, porque 24 não é maior que 24, mas como esse sinal \geq verifica se o 24 é maior ou igual a 24, essa expressão `classNumber \geq 24` é verdadeira.

Operadores de Comparação

No nosso décimo primeiro exemplo de Operadores de Comparação (Numbers), manteremos sem nenhuma alteração a nossa `let classNumber = 24` . Ficando assim:

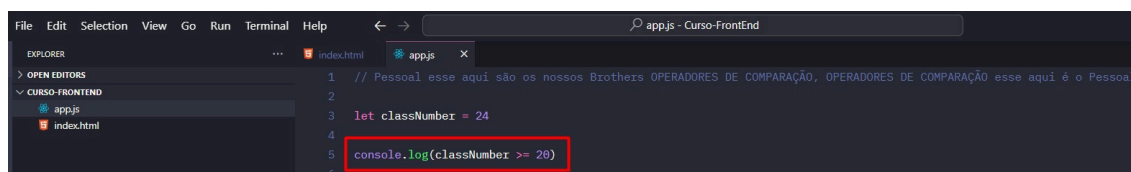
```
let classNumber = 24
```



```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
```

Adicionaremos um `console.log` passando a `classNumber` que é maior ou igual (`>=`) que 20, representada pelos comparadores `>=` seguido do número 20. Ficando assim:

```
console.log(classNumber >= 20)
```

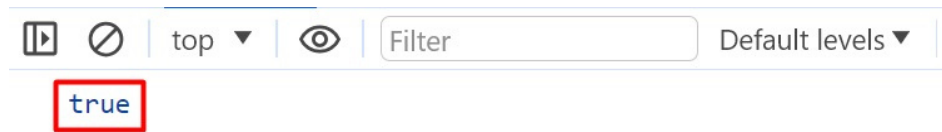


```
1 // Pessoal esse aqui são os nossos Brothers OPERADORES DE COMPARAÇÃO, OPERADORES DE COMPARAÇÃO esse aqui é o Pessoal
2
3 let classNumber = 24
4
5 console.log(classNumber >= 20)
6
```

E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que `true` está sendo mostrado no console.



Operadores de Comparação



E porque true apareceu no exemplo acima?

Essa resposta eu vou deixar para vocês pesquisarem.

Dica: tomem como base os últimos 2 exemplos (Lembrem do OU marotinho)

Sugestões de prática para essa aula:

1. Refazer e Entender os exemplos dessa aula;
2. Usar valores definidos por vocês para praticar mais sobre Operadores de Comparação com Strings;
3. Usar valores definidos por vocês para praticar mais sobre Operadores de Comparação com Numbers;
4. Notem que nós exemplos vistos nessa aula, utilizamos o `let` e se tivéssemos utilizado o `const` os resultados teriam sido(s) os mesmos? Fica mais essa sugestão para vocês realizarem!!!

Colocar isso em prática no VSCode e ver o resultado no console do Navegador.

Programação é prática, curiosidade e repetição!!!



/igor-rebolla

Pergunta do Amigo Internauta: Igor de São Paulo - SP

Na páginas 9 e 13 dessa aula nós vimos exemplos com o alfabeto começando com o caractere 'A' como primeiro valor, o caractere 'B' como segundo valor..... o caractere 'Z' como vigésimo sexto valor (tanto para o alfabeto na página 9 em minúsculo quanto para os 2 alfabetos - página 13 em minúsculo e maiúsculo).

Lembram da nossa aula 19 quando vimos que o JavaScript(JS) é zero-based(baseado em zero)..... Faria sentido o primeiro caractere tanto do alfabeto da página 9, quanto os alfabetos da pagina 13 começarem com o valor 0 ao invés de 1 (primeiro valor)?



/igor-rebolla

O que vamos aprender na aula 25:

Na vigésima quinta aula do curso (Décima Primeira aula de JavaScript) o que veremos: É surpresa – Parte 7 (Vem +++++++ coisa boa por ai!!!)

Feedbacks dessa vigésima quarta aula são bem-vindos.

Nos vemos na vigésima quarta aula que será disponibilizada no dia 09/05/2024 às 8h30 (Horário de Brasília) – Dentro do meu perfil no LinkedIn.

