

# Curso: Front-End do Bem



## AULA 25

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

- Operadores de Comparação  
(+4 Exemplos)
- Operadores de Comparação Estrita
  - Exemplos com Operadores de Comparação Estrita
- Sugestões de Prática para essa aula
  - O que vamos aprender na aula 26



# Operadores de Comparação (+4 Exemplos)

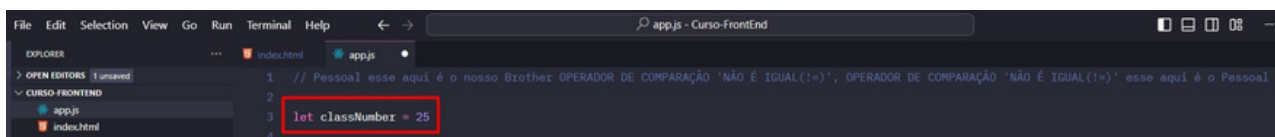
Antes de partirmos para o aprendizado sobre Operadores de Comparações Estrita, veremos +4 exemplos referentes aos Operadores de Comparação (Revisão da Aula 24). E faremos isso, por 2 motivos:

- 1º Para relembrarmos o conceito e praticarmos mais um pouquinho;
- 2º Para compararmos quando usarmos: Comparações Estrita (tema dessa aula 25)

Bora abrir o nosso arquivo app.js e na primeira linha dentro desse arquivo criaremos um comentário: `// Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL (!=)', OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL (!=)' esse aqui é o Pessoal`, para podermos comparar com o Uso de Comparações com Igualdade Estrita.

E logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

```
let classNumber = 25
```

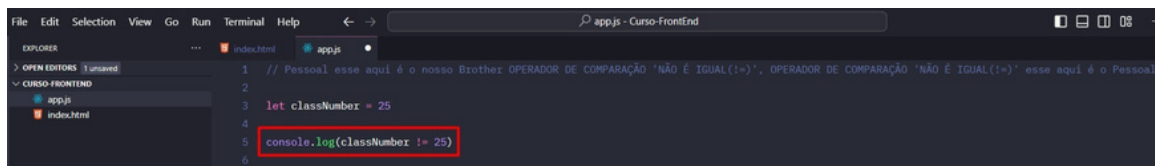


```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL (!=)', OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL (!=)' esse aqui é o Pessoal
2
3 let classNumber = 25
4
```

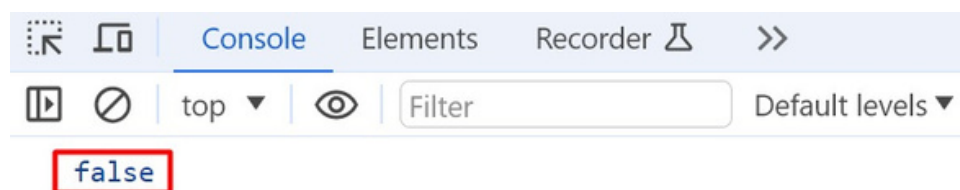
# Operadores de Comparação (+4 Exemplos)

Adicionaremos um `console.log` passando a `classNumber` que não será igual a 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.



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 25.

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 25 é igual a 25 ou seja: `25 == 25`

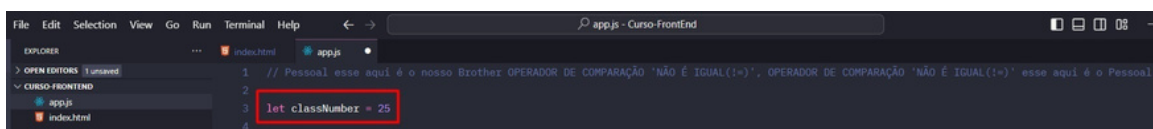


# Operadores de Comparação (+4 Exemplos)

Agora vamos comparar um número com uma string.

E logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

```
let classNumber = 25
```

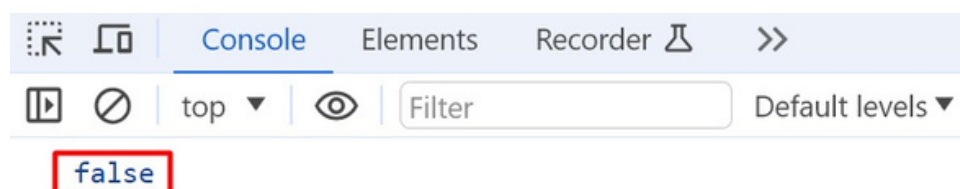


Adicionaremos um `console.log` passando a `classNumber` que não será igual a '25', representada pelo comparador `!=` seguido da string '25'. Ficando assim:

```
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 (+4 Exemplos)

E porque false apareceu no exemplo da página anterior?

Nós estamos falando que number 25 é diferente da string '25' e isso deveria ocasionar um TRUE, pois o tipo de um número (classNumber) é diferente do tipo do outro ('25') que é uma string.

[illegible]

Só que conforme a conversão de tipos que esse operador (!=) faz, nas profundezas nunca antes navegadas do JavaScript(JS), essa expressão `classNumber != '25'` resultará em FALSE.

Aqui que vem o "jump of the cat (pulo do gato)"... Por que como essa string '25' é convertida para number 25 nas profundezas nunca antes navegadas do javascript, essa expressão `className != '25'` é igualzinha essa aqui `className != 25`

Notem que estamos usando os operadores de comparação != (o que eu falei aqui ficará mais claro quando entrarmos em Operadores de comparação estrita) por isso eu pedi para vocês prestarem muita atenção aqui.

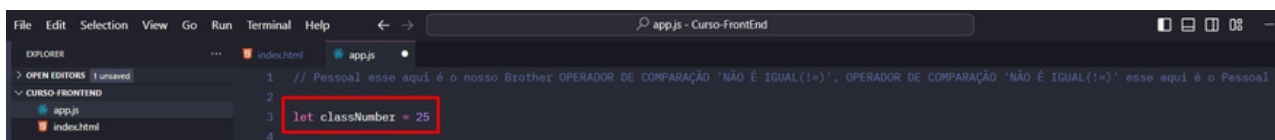


**/igor-rebolla**

# Operadores de Comparação (+4 Exemplos)

Bora abrir o nosso arquivo app.js e na primeira linha dentro desse arquivo criaremos um novo comentário: // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'IGUAL A (==)', OPERADOR DE COMPARAÇÃO 'IGUAL A (==)' esse aqui é o Pessoal, e logo abaixo do comentário declararemos uma let classNumber que recebe 25. Ficando assim:

```
let classNumber = 25
```



```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL(!=)', OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL(!=)' esse aqui é o Pessoal
2
3 let classNumber = 25
4
```

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

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

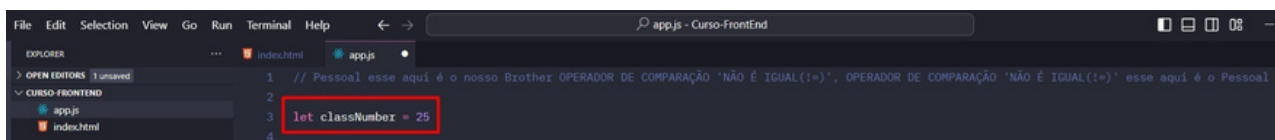


```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'IGUAL A (==)', OPERADOR DE COMPARAÇÃO 'IGUAL A (==)' esse aqui é o Pessoal
2
3 let classNumber = 25
4
5 console.log(classNumber == 25)
6
```

# Operadores de Comparação (+4 Exemplos)

Bora abrir o nosso arquivo app.js e na primeira linha dentro desse arquivo criaremos um novo comentário: // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'IGUAL A (==)', OPERADOR DE COMPARAÇÃO 'IGUAL A (==)' esse aqui é o Pessoal, e logo abaixo do comentário declararemos uma let classNumber que recebe 25. Ficando assim:

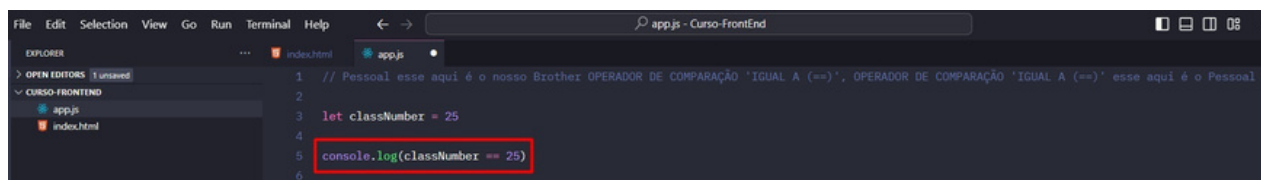
```
let classNumber = 25
```



```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL(!=)', OPERADOR DE COMPARAÇÃO 'NÃO É IGUAL(!=)' esse aqui é o Pessoal
2
3 let classNumber = 25
4
```

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

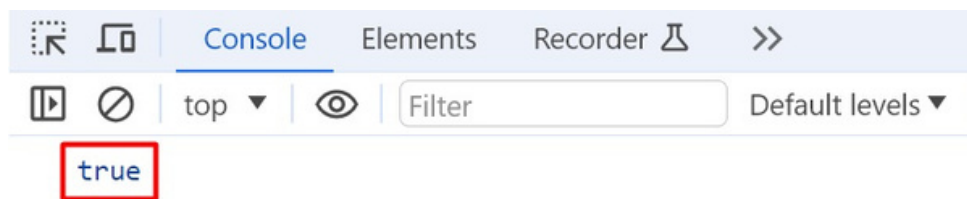
```
console.log(classNumber == 25)
```



```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO 'IGUAL A (==)', OPERADOR DE COMPARAÇÃO 'IGUAL A (==)' esse aqui é o Pessoal
2
3 let classNumber = 25
4
5 console.log(classNumber == 25)
6
```

# Operadores de Comparação (+4 Exemplos)

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 jeito certinho para efetuarmos a leitura da expressão que está dentro do nosso console.log é: classNumber é igual a 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 25 é igual a 25 ou seja: `25 == 25`

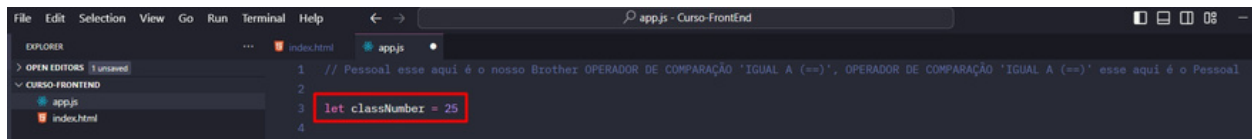


# Operadores de Comparação (+4 Exemplos)

Agora vamos comparar um número com uma string.

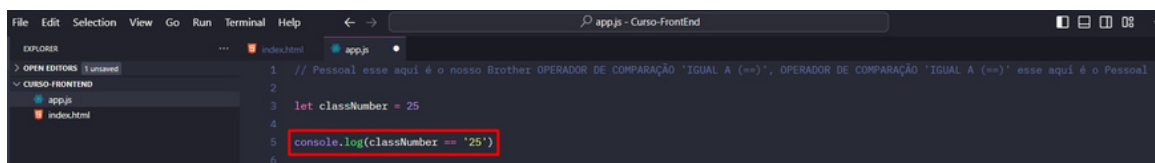
E logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

```
let classNumber = 25
```



Adicionaremos um `console.log` passando a `classNumber` que é igual `==` a 25, representada pelo comparador `==` seguido da string `'25'`. Ficando assim:

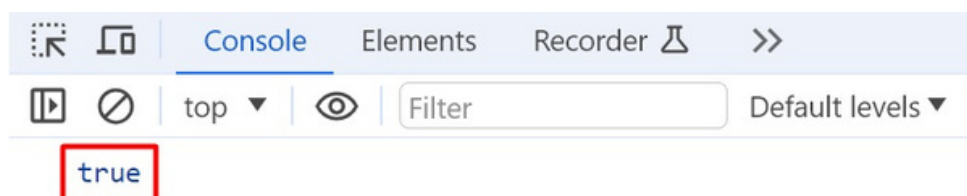
```
console.log(classNumber == '25')
```



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



# Operadores de Comparação (+4 Exemplos)



Igor, Igor, Igor, Igor.. Eita porque true apareceu no exemplo acima?

O JavaScript tá informando que o number 25 é igual a string '25'.

Sempre que o duplo sinal de igual (==) ou o sinal de exclamação seguido do sinal de igual (!=) são utilizados.... tanto um quanto o outro realizarão uma marotinha conversão de tipos.

Ou seja no nosso exemplo: Temos a nossa querida expressão (classNumber == '25') só que antes que a comparação entre esses valores fosse realizada, essa string '25' foi convertida para number 25 ou seja, quando o duplo sinal de igual ou o sinal de exclamação seguido do sinal de igual são usados, uma expressão que contem tipos diferentes pode ocasionar em TRUE, devido essa conversão que ocorre 'nas profundezas nunca antes navegadas do JavaScript(JS)'.

Vocês viram como as coisas começam a se encaixar.... e ficará mais claro quando começarmos com os exemplos: Operadores de Comparação Estrita.

E bora ver na próxima página esses Operadores de Comparação Estrita...



# Operadores de Comparação Estrita (Strict Comparision)

Como nós acabamos de ver, realizar comparações com os operadores não é igual (!=) e igual a (==), não são o jeito mais confiável de compararmos valores.

Igor, Igor, Igor, Igor. Uma dúvida aqui, na verdade são 2.... Se eles não são o jeito mais confiável de executar comparações... porque você mostrou isso pra gente? E a segunda dúvida: O que são esses operadores de comparação estrita (strict comparision)

Dúvidas excelentes... Bora lá responde-las:

- 1 - Quando vocês encontrarem em códigos de outras pessoas os sinal de exclamação e o sinal de igual (!=) ou o duplo sinal de igual (==) em códigos de terceiros e conhecerem o que está acontecendo ali;
- 2- Eles são do mesmo tipo e do mesmo valor. Tanto o sinal de (===) quanto o sinal de (!==) são chamados de Comparação Estrita(Strict Comparion). (Vai ficar mais claro essa definição com os exemplos que veremos a seguir).

Então bora de exemplos de comparação estrita....

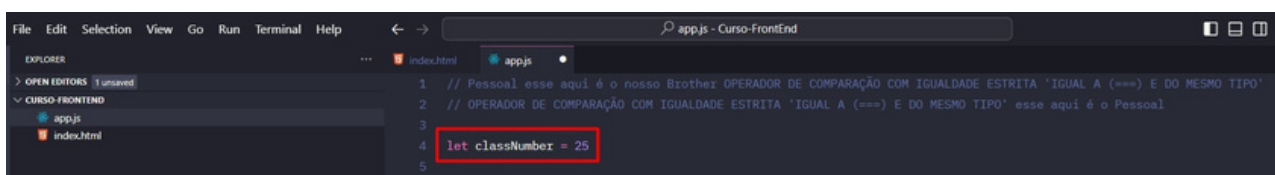
Só que antes vamos criar mais um comentário maroto, abriremos o nosso arquivo app.js e na primeira linha dentro desse arquivo criaremos um comentário: // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===)', OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===)' esse aqui é o Pessoal.



# Operadores de Comparação Estrita (Strict Comparision)

E logo abaixo do comentário declararemos uma let classNumber que recebe 25. Ficando assim:

```
let classNumber = 25
```

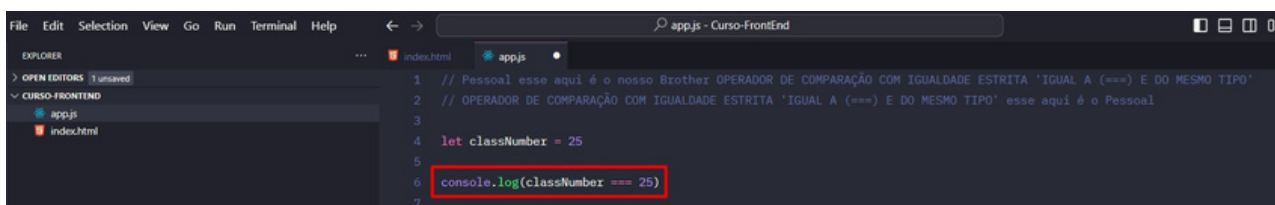
A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project named 'app.js - Curso-FrontEnd' with files 'app.js' and 'index.html'. The main editor window displays the 'app.js' file with the following code: 

```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===) E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===) E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
```

 The line `let classNumber = 25` is highlighted with a red rectangular box.

Adicionaremos um console.log passando a classNumber que é igual === a 25, representada pelo comparador de igualdade estrita === seguido do número 25. Ficando assim:

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

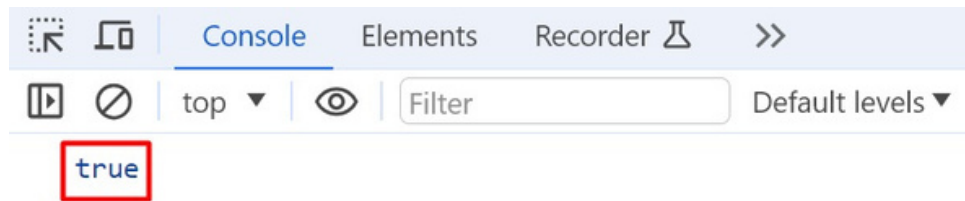
A screenshot of the Visual Studio Code editor interface, similar to the previous one. The main editor window displays the 'app.js' file with the following code: 

```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===) E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'IGUAL A (===) E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
6 console.log(classNumber === 25)
7
```

 The line `console.log(classNumber === 25)` is highlighted with a red rectangular box.

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 Estrita (Strict Comparision)



E porque true apareceu no exemplo acima?

Porque ele fez a comparação estrita tanto do valor quanto do tipo ou seja:

- o número 25 que está armazenado na let classNumber é igual (tem o mesmo valor) que o 25 que está dentro do console.log?

Sim, é igual(tem o mesmo valor).

- os 2 são do tipo number?

Sim, os dois são do tipo number.

Resumindo: Estamos informando que por causa desse sinal (===), que a classNumber tem o mesmo valor e também o mesmo tipo que 25.

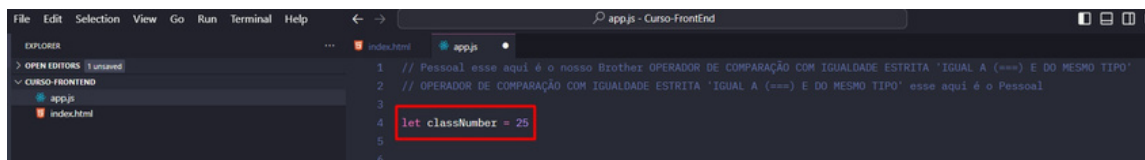
Por isso true apareceu no console.

# Operadores de Comparação Estrita (Strict Comparision)

Agora vamos comparar (usando o operador de comparação estrita) um número com uma string.

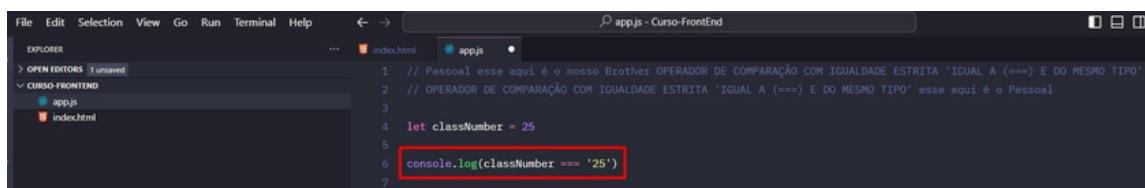
E logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

```
let classNumber = 25
```



Adicionaremos um `console.log` passando a `classNumber` que é igual `===` a `'25'`, representada pelo comparador de igualdade estrita `===` seguido da string `'25'`. Ficando assim:

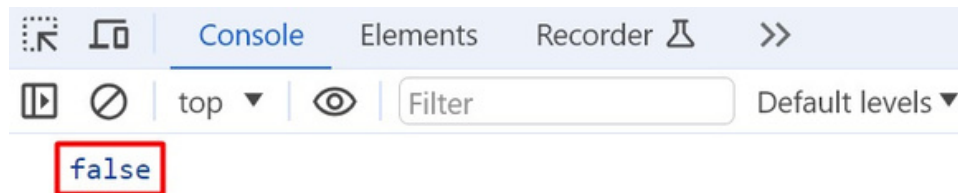
```
console.log(classNumber === '25')
```



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 Estrita (Strict Comparision)



E porque false apareceu no exemplo acima?

Porque estamos verificando tanto o valor quanto o tipo.

E classNumber que é um number 25 não é igual a uma string '25'.

E lembra que eu pedi pra vocês prestarem muitaaaaaaaaaaaa atenção em uma explicação na página 5 dessa aula, pois bem... chegou o motivo que essa explicação fará sentido:

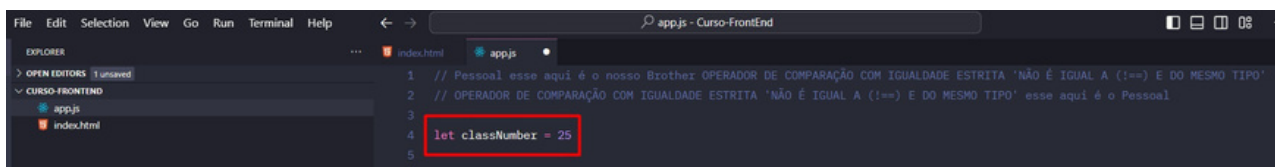
Aqui não tem conversão de tipos sendo realizada nas profundezas nunca antes navegadas do JavaScript(JS).

Ou seja quando usamos o Operador de Comparação estrita (===) ele não faz essa conversão... diferentemente do seu irmão (==) que realiza essa conversão.

# Operadores de Comparação Estrita (Strict Comparision)

Bora abrir o nosso arquivo app.js e na primeira linha dentro desse arquivo criaremos um novo comentário: `// Pessoal esse aqui é o nosso Brother` OPERADOR DE COMPARAÇÃO ESTRITA 'NÃO É IGUAL (!=)', OPERADOR DE COMPARAÇÃO ESTRITA 'NÃO É IGUAL (!=)' esse aqui é o Pessoal, e logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

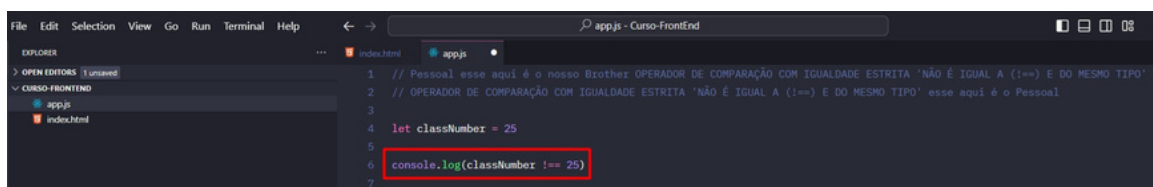
```
let classNumber = 25
```



```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!=)' E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!=)' E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
```

Adicionaremos um `console.log` passando a `classNumber` que não é igual `!==` a 25, representada pelo comparador (aqui chamarei carinhosamente) de não igualdade estrita `!==` seguido do número 25. Ficando assim:

```
console.log(classNumber !== 25)
```

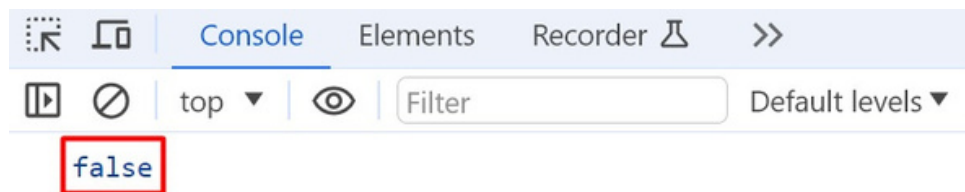


```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!=)' E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!=)' E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
6 console.log(classNumber !== 25)
7
```



# Operadores de Comparação Estrita (Strict Comparision)

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 número 25 que está armazenado na classNumber não é igual (!==) ao número 25?

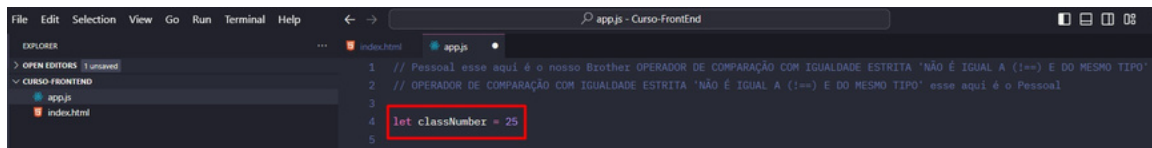
Não(false), pois 25 é igual a 25.

# Operadores de Comparação Estrita (Strict Comparision)

Agora vamos comparar (usando o operador de comparação estrita) um número com uma string.

E logo abaixo do comentário declararemos uma `let classNumber` que recebe 25. Ficando assim:

```
let classNumber = 25
```



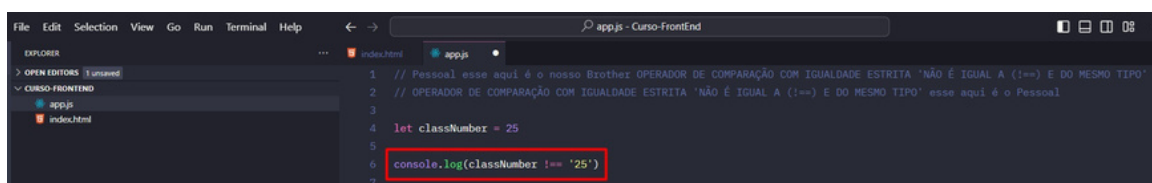
A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project named 'app.js - Curso-FrontEnd' with files 'index.html' and 'app.js'. The main editor window displays the 'app.js' file. The code in the editor is as follows:

```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!==) E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!==) E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
```

The line `let classNumber = 25` on line 4 is highlighted with a red rectangular box.

Adicionaremos um `console.log` passando a `classNumber` que não é igual `!==` a `'25'`, representada pelo comparador (aqui chamarei carinhosamente) de não igualdade estrita `!==` seguido da string `'25'`. Ficando assim:

```
console.log(classNumber !== '25')
```



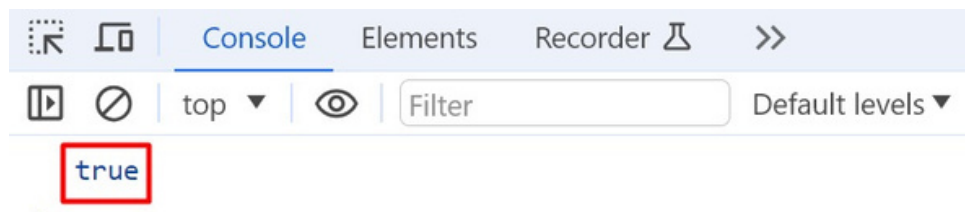
A screenshot of the Visual Studio Code editor interface, similar to the previous one. The main editor window displays the 'app.js' file with the following code:

```
1 // Pessoal esse aqui é o nosso Brother OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!==) E DO MESMO TIPO'
2 // OPERADOR DE COMPARAÇÃO COM IGUALDADE ESTRITA 'NÃO É IGUAL A (!==) E DO MESMO TIPO' esse aqui é o Pessoal
3
4 let classNumber = 25
5
6 console.log(classNumber !== '25')
7
```

The line `console.log(classNumber !== '25')` on line 6 is highlighted with a red rectangular box.

# Operadores de Comparação Estrita (Strict Comparision)

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 agora nós estamos utilizando a comparação estrita(!==).

E o número 25 que está armazenado na classNumber não será igual a string '25'

OBS.: O operador de comparação estrita !==, pode ser chamado (ambos estarão corretos no dia a dia):

- não é igual a
- diferente de

\* **não igualdade estrita** eu utilizei como forma didática para ilustrar os exemplos.

# 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 Estrita `===` com Strings e Numbers;
3. Usar valores definidos por vocês para praticar mais sobre Operadores de Comparação Estrita `!==` com Strings e 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!!!



# 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 26:

Na vigésima sexta aula do curso (Décima Segunda aula de JavaScript) o que veremos: É surpresa – Parte 8 (Vem mais coisa boa por ai!!!)

**Feedbacks dessa vigésima quinta aula são bem-vindos.**

Nos vemos na vigésima sexta aula que será disponibilizada no dia 23/05/2024 às 8h30 (Horário de Brasília) – Dentro do meu perfil no LinkedIn.



**/igor-rebolla**