

Curso: Front-End do Bem



AULA 18

- Uma Visão Geral sobre Tipo de Dados
 - Comentários
 - Exemplos de Comentários
 - Numbers
 - Exemplos de Numbers
- Sugestões de Prática para essa aula
 - O que vamos aprender na aula 19



Uma visão geral sobre tipos de dados

Na aula 17 nós vimos como os valores são armazenados em uma variável, só os valores que nós vimos foram apenas formados por Numbers(Números), E os Numbers(Números) que nós utilizamos nos exemplos estavam relacionados com: cep, id, month e streetNumber.

E o tipo Number(Número) é só o primeiro dos tipos de dados em JavaScript(JS). Em JavaScript nós podemos trabalhar com até 8 tipos de dados. Listaremos cada um deles separadamente, mostrando suas particularidades.

- E só reforçando o primeiro tipo de dado é o que nós já vimos que é o Number(Número)... 9,10,11, 8.90 4567 ah e os números decimais também fazem parte do tipo Number(Número)

- O segundo tipo de dados são as boas e velhas Strings. Strings nada mais são que uma cadeia de caracteres carinhosamente chamados de textos mas para que possamos representar textos dentro do JavaScript(JS), temos que representa-los dentro de aspas duplas ou simples...

Aspas duplas ("Strings") - Exemplo: "aula18@aula18.com"

Aspas simples ('Strings') - Exemplo: 'Salve, galera'

- O terceiro tipo de dado é o nobre Boolean. Booleans são lógicas que representam valores verdadeiro(s) ou falso(s) (true / false) e usaremos esse tipo de dado para validar se determinadas condições são verdadeiras ou falsas (true / false).



Uma visão geral sobre tipos de dados

- O quarto tipo de dado é o Null. O Null é um meio para falarmos explicitamente que uma variável não vale nada ou seja não tem valor, podemos criar uma variável e atribuir um Null a ela, para apontar que essa variável ainda não tem nenhum valor

- O quinto tipo de dado é o Undefined. O Undefined parece com o Null só que é diferente (parece brincadeira, mas é isso mesmo). A diferença é que o Undefined é colocado automaticamente pelo JavaScript(JS)

OBS IMPORTANTE: Tanto o Null como o Undefined representam valores vazios, só que o NULL é utilizado pela pessoa que tá fazendo o desenvolvimento e o Undefined é colocado de forma automática pelo JavaScript(JS).

OBS IMPORTANTE 2 - A missão: Peço a gentileza para quem estiver estudando por esse material, para não se preocupar com Undefined, Null, Condição entre outros termos nesse momento, pois conforme vamos evoluindo com as aulas, esses tipos de dados farão cada vez mais sentido para vocês. Confia no tio!!!

- O sexto tipo de dado é o OBJECT. OBJECT são estruturas de dados um pouco mais robustas e dentro dessas estruturas podemos ter (funções e várias propriedades).

Existem diversos tipos de Objetos diferentes dentro da linguagem JavaScript(JS) os quais podemos utilizar, eles são: (Arrays, Objetos Literais ,Funções e Datas).



Uma visão geral sobre tipos de dados

E não menos importante temos os 2 últimos tipos de dados (BIGINT e o SYMBOL), o qual falaremos brevemente aqui abaixo para que vocês tenham conhecimento sobre os mesmos.

- O sétimo tipo de dado é o BIGINT, o BigInt é utilizado para simbolizar números inteiros “grandão”.
- O oitavo tipo de dado é o SYMBOL, o Symbol é uma adição mais recente dentro do JavaScript(JS) que está relacionado aos objetos

Então nas próximas aulas estudaremos (Number, String, Boolean, Null, Undefined e Arrays com mais detalhes).

Uma variável nós já vimos o que é, uma variável pode armazenar diferentes tipos de dados (Uma string, um número e etc...). Eu não gostaria que de forma alguma você se sinta carregado com muita informação ou venha a achar que terá que guardar todos esses tipos de dados, não foca nisso, por favor.

Isso aqui é só um resumo bem rápido para que você possa saber que existem tipos de dados na linguagem e isso é tudo o que você necessita conhecer até esse momento . E com o passar das aulas e do tempo quanto mais você conseguir praticar no seu dia a dia, essas informações surgirão de forma automática pra ti.

Por isso eu sempre falo:

Programação é prática, curiosidade e repetição!!!!



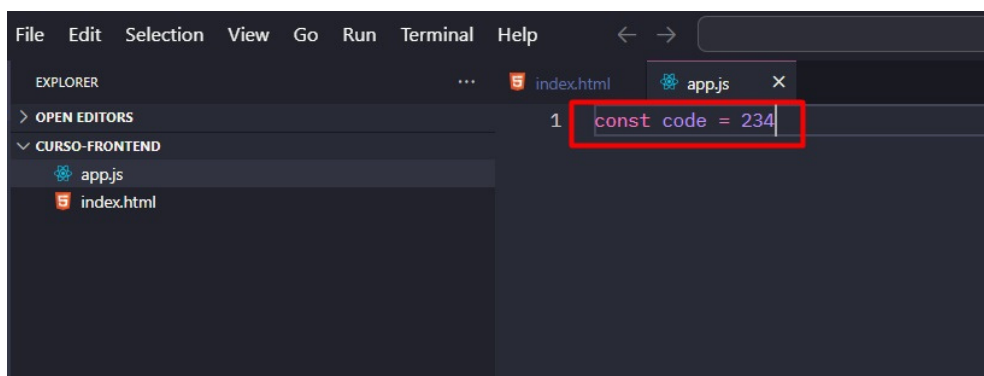
Comentários

Vamos abrir o nosso VSCode para melhor entendimento, colocar o Live Server no Ar e usaremos o nosso arquivo app.js

Digamos que você queira fazer algum tipo de comentário nesse código para que outra pessoa faça a leitura desse código, você pode fazer isso, criando um comentário de uma única linha: usando duas barras e em seguida o seu comentário, ficando assim: `//` Esse é o meu primeiro comentário de uma única linha

Digitaremos uma const code que recebe 234, ficando assim:

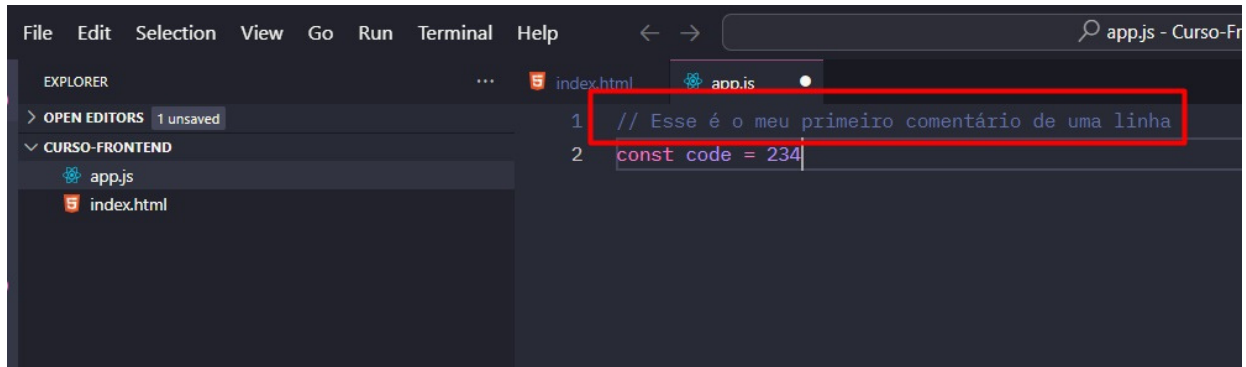
```
const code = 234
```



Vamos colocar o cursor do mouse entre o número 1 e o letra “c” da palavra const, pressionaremos o enter a assim vai criar um espaço onde digitaremos o nosso comentário, ficando assim:

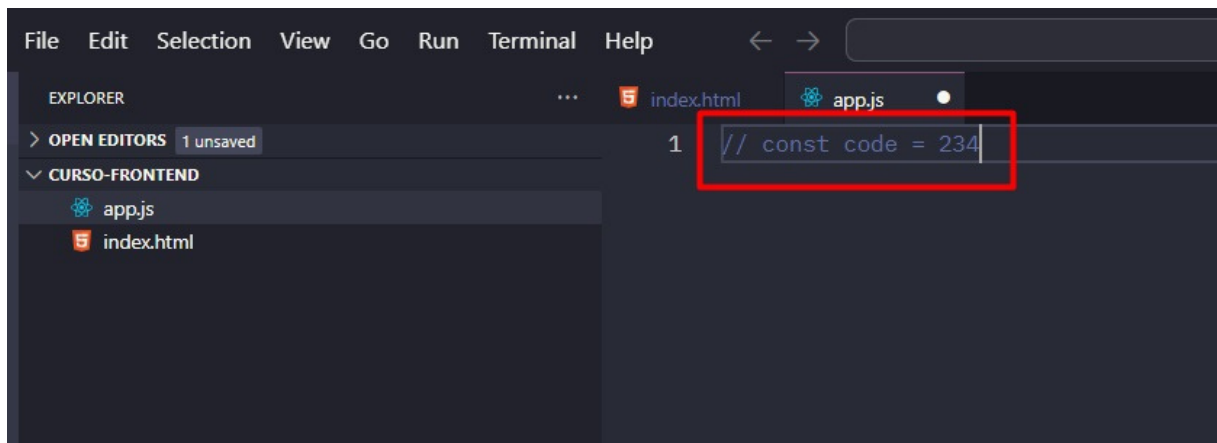
```
// Esse é o meu primeiro comentário de uma linha  
const code = 234
```

Comentários



É possível também comentarmos o código inserindo `//` antes do código, para isso apagaremos o comentário de uma linha do exemplo acima `// Esse é o meu primeiro comentário de uma linha` e colocaremos: `//` antes do `const code = 234`, ficando assim:

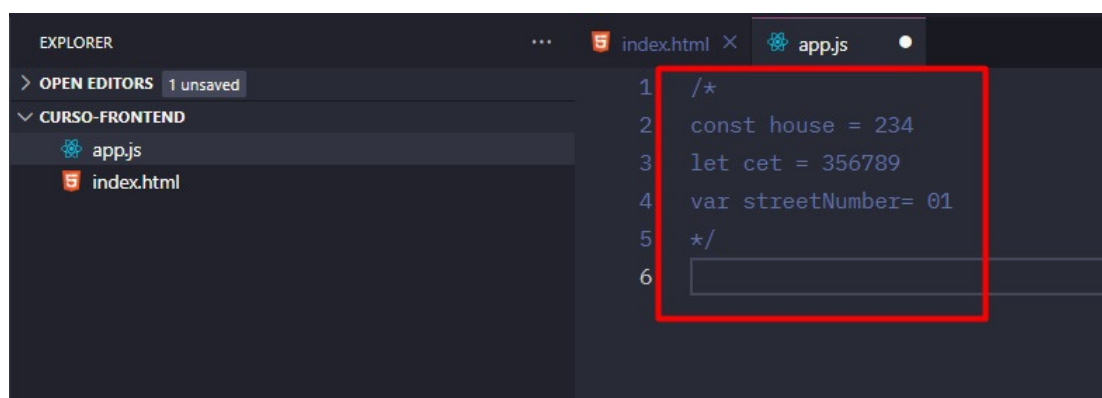
```
// const code = 234
```



Comentários

Conseguimos também escrever comentários de várias linhas, onde podemos quebrar uma linha e escrevermos um bloco de texto dentro dele, para isso, basta abrimos o comentário com uma barra e um asterisco `/*` Colocar o comentário aqui!!! E fechar o comentário com um asterisco barra `*/` como fizemos quando aprendemos o CSS. Ficando assim:

```
/*  
const house = 234  
let cet = 356789  
var streetNumber= 01  
*/
```



Também é possível comentarmos **bloco de código** com esse tipo de comentário e assim esse código não será executado pelo JavaScript(JS)

Igor, mas essa explicação sobre comentários foi muito rápida, é isso mesmo?

Vou responder essa pergunta com um comentário de uma linha até para ir praticando

```
// Sim
```



Primeiro tipo de dado: Numbers

Na aula 17 quando aprendemos sobre variáveis, vimos um pouco sobre Numbers e agora falaremos mais detalhadamente referente aos Numbers

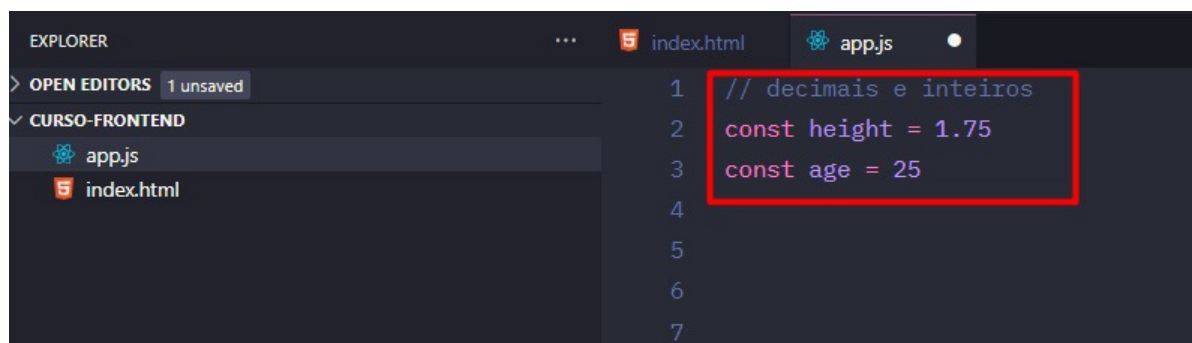
Existem inúmeras situações que os Numbers podem ser usados no JavaScript e agora veremos alguns desses casos

Como eu já estou com o meu VsCode aberto, o Live Server marotinho no ar e o arquivo app.js no jeito, bora ver esses Numbers Danadinhos de forma mais detalhada.

Abaixo do comentário: `// decimais e inteiros` criaremos 2 consts(constantes), a primeira const se chamará height(altura) que recebe 1.75 e a segunda se chamará age(idade) que recebe 25, ficando assim:

```
const height = 1.75  
const age = 25
```

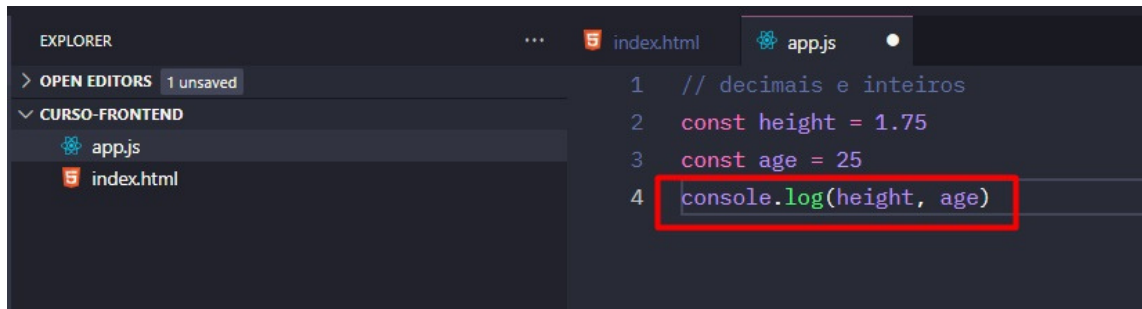
IMPORTANTE SABER: Para representarmos números decimais em JavaScript(JS) nos utilizamos o ponto (.) ao invés de virgula (,)



```
1 // decimais e inteiros  
2 const height = 1.75  
3 const age = 25  
4  
5  
6  
7
```

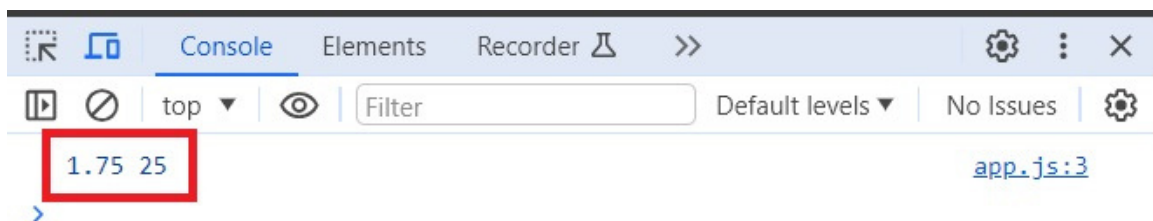

Primeiro tipo de dado: Numbers

E agora abaixo dessas 2 consts declararemos um `console.log(height, age)`.



```
1 // decimais e inteiros
2 const height = 1.75
3 const age = 25
4 console.log(height, age)
```

E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que os 2 valores são exibidos lado a lado. Onde o primeiro valor é um número decimal e o segundo valor é um número inteiro.

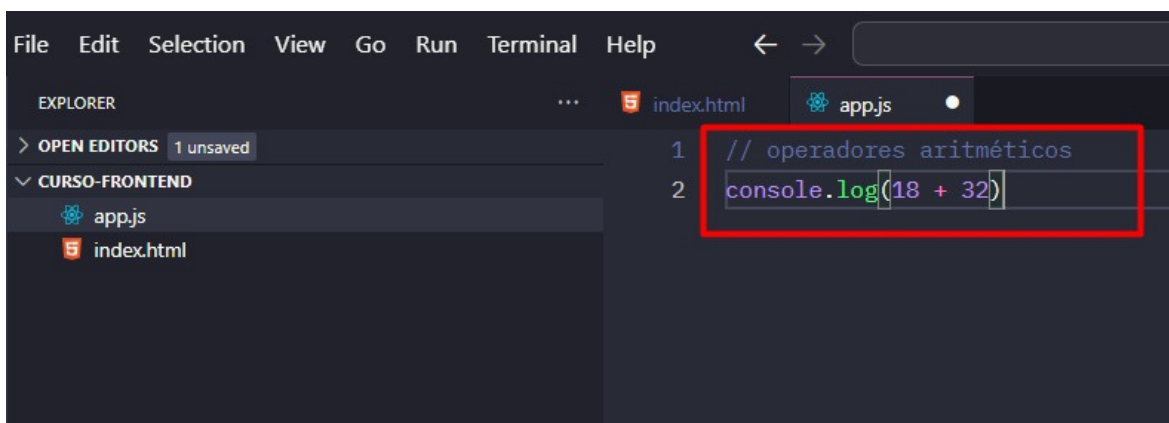


Primeiro tipo de dado: Numbers

Agora que vimos tanto os números decimais quanto os números inteiros, veremos o que é possível fazer “magicamente” com os números no JavaScript(JS). De forma resumida: todas as operações matemáticas, tais como: (adição, subtração, divisão, multiplicação, potencia e etc...)

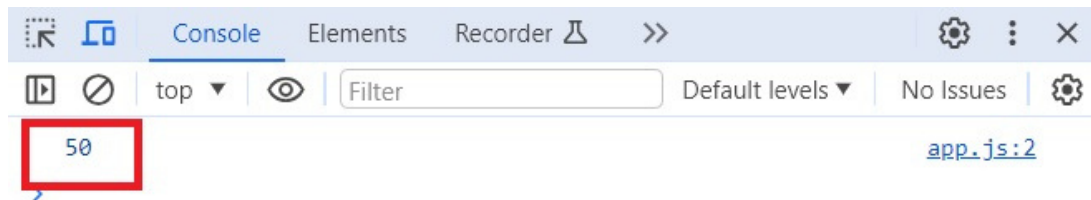
Abaixo do comentário `// operadores aritméticos` vamos inserir um `console.log` e passar `18 + 32`, ficando assim:

```
console.log(18 + 32)
```



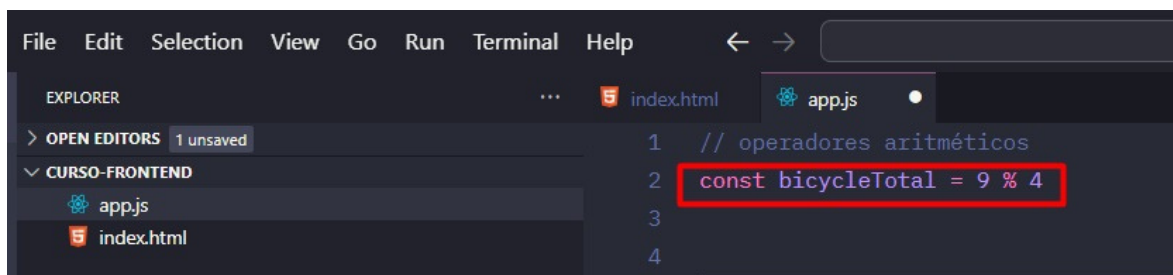
E ao salvarmos o arquivo `app.js` com aquele `CTRL + S` maroto podemos ver no console do Navegador(Chrome) que o número 50 será exibido.

Primeiro tipo de dado: Numbers



E ainda Abaixo do comentário `// operadores aritméticos` vamos declarar uma const `bicycleTotal` que recebe 9 módulo 4, ficando assim:

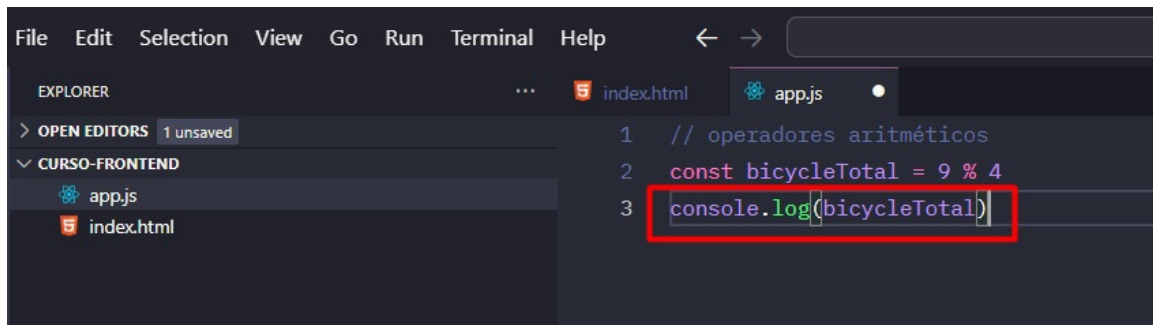
```
const bicycleTotal = 9 % 4
```



Digitaremos um `console.log` passando a `bicycleTotal`, ficando assim:

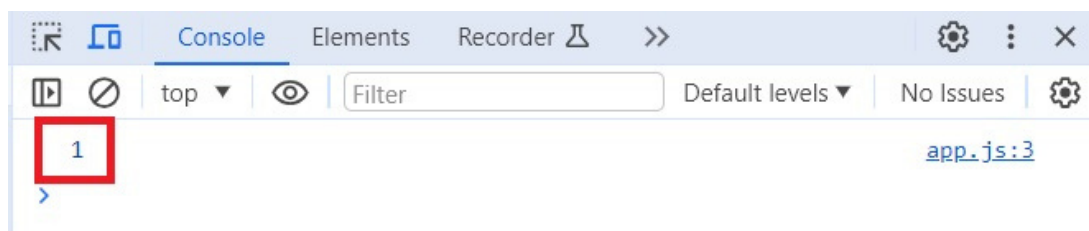
```
console.log(bicycleTotal)
```

Primeiro tipo de dado: Numbers



```
1 // operadores aritméticos
2 const bicycleTotal = 9 % 4
3 console.log(bicycleTotal)
```

E ao salvarmos o arquivo app.js com aquele CTRL + S maroto poderemos ver no console do Navegador(Chrome) que o valor 1 é exibido.



Igor, ficou confuso muito confuso mesmo... Módulo, Símbolo de Porcentagem que recebe determinado valor, poderia explicar melhor? Claro, o operador modulo aqui dentro do JavaScript(JS) é representado pelo símbolo de Porcentagem... ele recebe o resto da divisão entre 2 números.

Vou dar um exemplo, para ficar ainda mais claro.... bora ver na próxima página.

Primeiro tipo de dado: Numbers

O nome da variável `bicycleTotal` foi pensado cuidadosamente para melhor entendimento....Se você tem 9 bicicletas e divide essas bicicletas entre 4 amiguinhos. Com quantas bicicletas cada amiguinho fica?

Você responderia e eu confesso que também responderia: Ah.... com 2 bicicletas e meia, só que não podemos cortar uma bicicleta ao meio ou seja nesse exemplo vai sobrar uma bicicleta inteira, então o resto dessa operação será 1.

E logo abaixo do comentário `//` ordem das operações matemáticas vamos falar um pouco sobre ordem de precedência entre operações que é a mesma forma que a ordem da precedência básica da matemática, resumindo: quando uma expressão tem mais de uma operação você pode obter respostas diferentes dependendo da ordem que a expressão é resolvida, por isso que existe uma ordem de precedência determinada para execução das operações que é a seguinte:

1º ordem de precedência determinada: Parênteses

2º ordem de precedência determinada: Raízes ou Potências

3º ordem de precedência determinada: Multiplicação e Divisão

4º ordem de precedência determinada: Adição e Subtração



/igor-rebolla

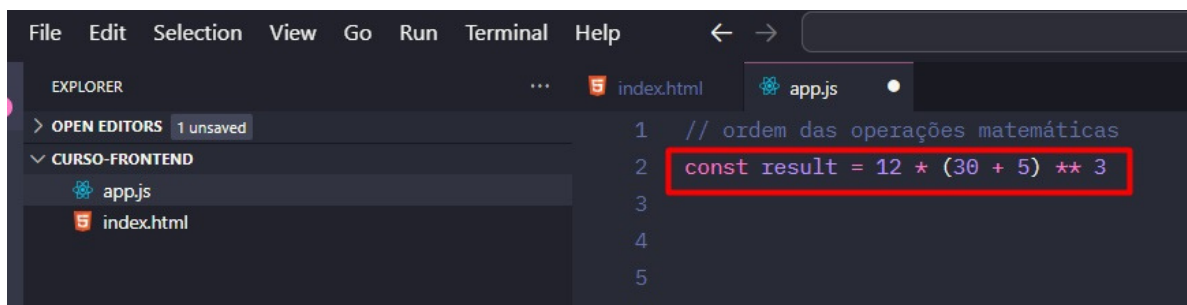
Primeiro tipo de dado: Numbers

Resumindo: Antes de qualquer coisa o JavaScript(JS) verificará se uma expressão tem parênteses, caso não tenha parênteses ela vai procurar uma raiz ou uma potência, caso também não tenha nenhuma raiz e nenhuma potência ela procurará por multiplicação e divisão, caso também não tenha nenhuma multiplicação e divisão... Calma que tá acabando... UFA.... e por último e não menos importante a boa e velha adição e subtração caso ela encontre uma adição ou subtração ela vai resolver isso.

Ainda abaixo do comentário `// ordem das operações matemáticas`, declararemos uma `const result` que recebe doze vezes, abre e fecha parênteses e dentro desses parênteses vamos inserir `30 + 5` e fora desses parênteses vamos eleva-lo ao cubo, ficando assim:

```
// ordem das operações matemáticas
const result = 12 * (30 + 5) ** 3
```

Obs.: A operação digitada acima mostra que essa ordem clássica de precedência prevalecerá na expressão.

A screenshot of a code editor interface. The top menu bar includes 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', and 'Help'. The left sidebar shows the 'EXPLORER' view with a file tree containing 'index.html' and 'app.js'. The main editor area shows the 'app.js' file with the following code:

```
1 // ordem das operações matemáticas
2 const result = 12 * (30 + 5) ** 3
3
4
5
6
```

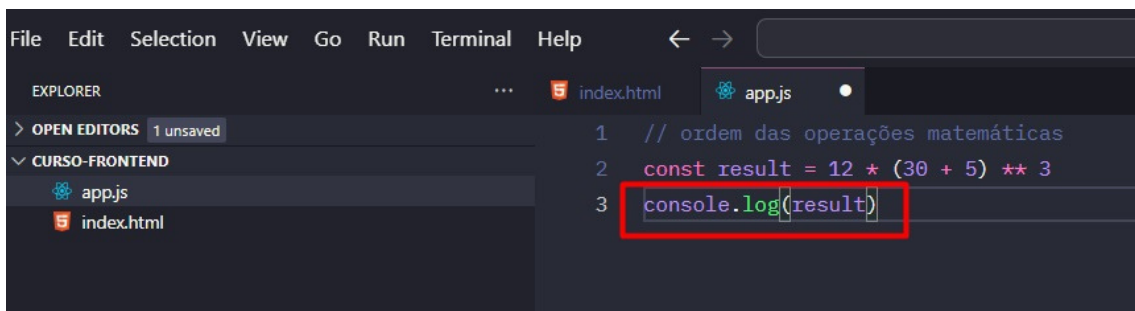
The line `const result = 12 * (30 + 5) ** 3` is highlighted with a red rectangular box.

Primeiro tipo de dado: Numbers

Antes de qualquer coisa a expressão entre parênteses será resolvida, onde $30 + 5$ é igual a 35 e logo depois dos parênteses a expressão que envolve as potências é resolvida onde 35 elevado ao cubo é igual a 42875 e depois das potências a expressão que envolve multiplicação é resolvida, onde 12 vezes 42875 é igual a 514.500.

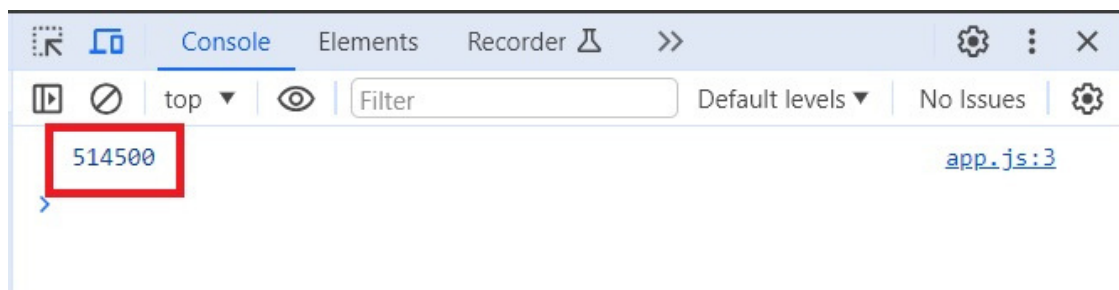
Logo abaixo vamos digitar `console.log` e passar a `result`, ficando assim:

`console.log(result)`



```
1 // ordem das operações matemáticas
2 const result = 12 * (30 + 5) ** 3
3 console.log(result)
```

E ao salvarmos o arquivo `app.js` com aquele CTRL + S maroto podemos ver no console do Navegador(Chrome) que o valor 514.500 será exibido no console



Sugestões de prática para essa aula:

1. Refazer e Entender os exemplos dessa aula;
2. Na página 7 dessa aula eu deixei **blocos de códigos** destacado tentar fazer o comentário do bloco de códigos dentro do VSCode
3. Usar valores definidos por vocês para praticar mais sobre: Módulos e ordem das operações matemáticas
Colocar isso em prática no VSCode e ver o resultado.

Programação é prática, curiosidade e repetição!!!!

O que vamos aprender na aula 19:

Na décima nona aula do curso(Quinta aula de JavaScript) vamos: Concluir o tipo de dados Number e Entender sobre o segundo tipo de dados (Strings).

Feedbacks dessa décima ótima aula são bem-vindos.

Nos vemos na décima nona aula que será disponibilizada no dia 14/03/2024 às 8h30 (Horário de Brasília) - Dentro do meu perfil no LinkedIn.



/igor-rebolla