# Aula 6 – tratamento de dados

Breve revisão aula anterior:

O que é uma variável?

- É um espaço onde podemos adicionar números, palavras, strings etc.

Como declarar uma variável numérica em JS?

- (eu fiz no node e consegui)

- ou usando os comandos ao lado.

Tipos primitivos do JS?

- Essa eu não lembro bem, revisar!

O que acontece se você coloca ‘null’ dentro de uma variável?

- Ele apaga a informação da variável.

• TRatamento de dados

Na aula passada, aprendemos a usar caixas de diálogo onde o usuário consegue digitar algo, mas agora vamos ver como gravar essas informações. Para usar caixa de diálogo usamos:

window.prompt('Qual é seu nome?')

Com esse comando até exibimos a mensagem para o usuário, porém, a informação fornecida pelo o usuário se perde pois não “pedimos” ao JS para salvar essa informação, mas podemos fazer isso criando uma variável:

var nome = window.prompt('Qual é seu nome?')

Depois podemos usar o window.alert para exibir essa variável e dar uma saudação para o usuário, por exemplo:

window.alert('É um prazer te conhecer, ' + nome)

Observação: dentro de JS o sinal de + nessa linha tem a função de **concatenação.** Isso significa que o sinal de + nessa situação vai apenas juntas as strings e não somar!!

Mas e quando precisamos somar o número contido em duas variáveis? Para isso precisamos, além de fazer a variável pegar a o número, converter a variável para ser um número, pois o comando acima recebe informação apenas como string, se botarmos um número mesmo assim pro JS vai continuar sendo uma string e vai apenas concatenar, por isso convertemos a variável com os seguintes comandos:

• Number.parseInt(number): para converter como número inteiro;

•Number.parseFloat(number): para converter como número real (número com vírgula)

•Number(number): Essa sintaxe reconhece sozinha se o número é inteiro ou real.

Exemplo:

var n1 = Number.parseInt (window.prompt('Digite um número:'))

Escrevendo o código assim, window.prompt que reconhece tudo como strings, converte para número, assim podemos manipular números pois sem isso esse comando nativamente reconhece apenas como strings qualquer informação.

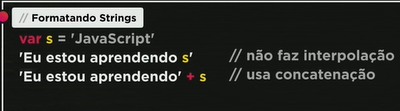
Também podemos converter um número para strings usando o seguinte comando:

•String(number)

ou

•number.toString()

• FOrmatando Strings



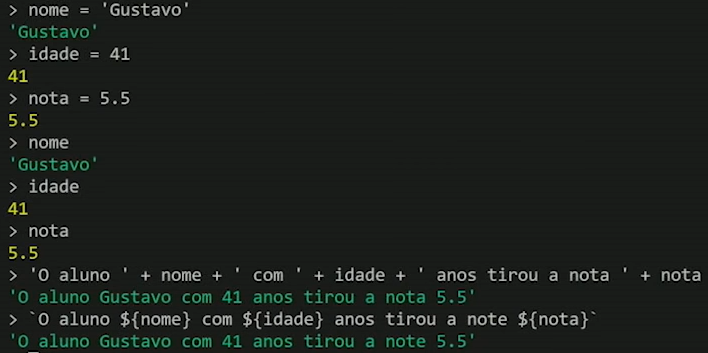
Acima a diferença entre não interpolação e concatenação.

Também podemos formatar strings usando **template strings:**



Neste caso em vez de aspas estamos usando crase, pois a crase é a delimitadora das templates strings.

Aquele cifrão seguido de chaves chamamos de place holder. Abaixo confira como fica mais fácil escrever inumeras variáveis na mesma string com template strings.



Note que usando as templates strings o código fica mais bonito e limpo em comparação a quando concatenamos tudo, e fica mais fácil adicionar todas as variáveis em apenas uma string.

**Manipulando Caracteres:**

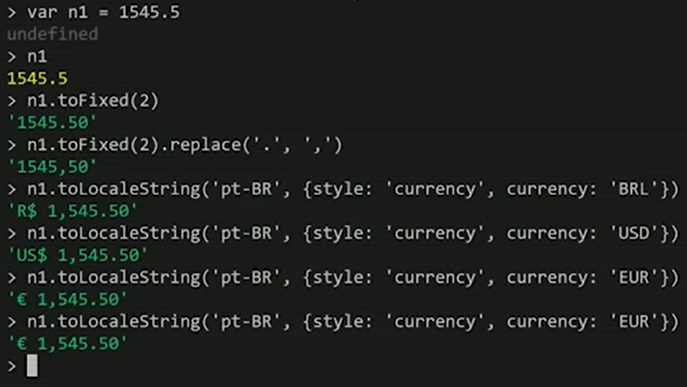
• string.length: diz quantos caracteres a string tem;

• strings.toUpperCase(): põe tudo para MAIUSCULAS;

•string.toLowerCase(): põe tudo para minúsculas;

•formatando moedas

Abaixo algumas opções de como formatar moedas:



•n1.toFixed(2) - faz com que a variavel tenha duas casas depois da virgula, mesmo inserindo a variavel com um numero depois da virgula, ela vai ter duas, e o que determina isso é o numero dentro do parênteses.

•n1.toFixed(2).replace(‘.’ , ’,’) – o atributo .replace vai substituir o ponto pela vírgula, dentro do parênteses especificamos primeiro o que vai ser substituído, e depois pelo o que vamos substituir.

•n1.toLocalString(‘pt-br’, {style ‘currency’, currency: ‘BRL’}) – essa string reconhece que vamos ’localizar’ o Brasil, depois através dos atributos seguintes vai inserir o tipo de moeda que foi localizada e exibir o valor da variável como dinheiro do brasil, e se alterarmos as abreviações dentro de ‘currency’ para ‘USD’ ou ‘EUR’ por exemplo, vai exibir o valor em dólar ou euro, respectivamente.