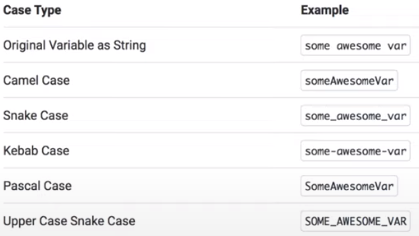
VARIÁVEIS E TIPOS

AULA 01 - ATRIBUINDO VALORES:

**1.VARIÁVEIS:**

\*atribuição de valores:



-camel case é o mais usado geralmente.

-upper case snake case: usado para constantes.

\*variáveis: var e let.

-var: usado para escopo global.

-let: usada para escopo local se declarado dentro de bloco, e de escopo global se declarado fora de bloco.

\*declaração: tanto para var quanto para let. Se for feita para let dentro de bloco com um valor e fora do bloco com outro valor, o valor de dentro do bloco só vale para o bloco, fora dele ela terá o outro valor.

\*atribuição: tento para var quanto para let. é a atribuição de valor para a variável usando o =.

\*redeclaração: só ocorre para o var. quando se declara novos valores para variáveis.

\*reatribuição: se for feita para var dentro do bloco, ela valerá também fora do bloco.

\*hoisting: só ocorre para o var. é a atribuição de valor para uma variável antes de se declarar ela. Se ela for declarada posteriormente como let, o js não vai entender, pois o hoisting só funciona para escopo global (var).

**2.CONSTANTES:**

-são declaradas em SNAKE\_UPPER\_CASE

-tem escopo de bloco

-não faz hoisting. se usado, vai entender que é var. se tentar atribuir acima, vai sinalizar que não é possível reatribuir.

-não é possível reatribuir valor para constante.

-não é possível redeclarar uma constante.

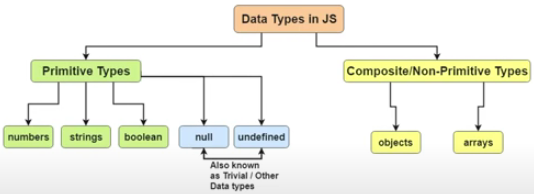
**3.LET, VAR E CONST:**



AULA 02 - TIPOS:

**1.ESTRUTURA DE DADOS:**

-o js tem tipagem dinâmica e fraca, ou seja, a tipagem pode mudar ao longo da codificação e antes de declarar valor não é necessário especificar o tipo dele (inteiro, string, etc).



**2.STRINGS:**

-comumente utilizada para textos.

-valores declarados entre aspas simples ou duplas ou crases.

-interpolação (template strings): com a utilização de crases para delimitação, é possível criar uma string interpolando texto e variáveis identificadas com ${}.



-escape: para pular linha: “\n;” para printar “: “/””;

\*nomedavar.xxx: funcionalidades das strings (apenas para retorno, não modifica. pode nova declaração)

-split: separa os argumentos da string item por item.

-split(“ ”): separa as palavras da string.

-includes(“ xxx ”): para saber se xxx está contido na string.

-concat: para concatenar strings

-replace

-o ponto de exclamação significa negação junto de uma string, como a de validação, por exemplo.

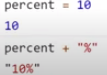
**3:NUMBERS:**

-inteiros ou decimais.

-declarados sem aspas.

-propriedade math.: math.floor(arredonda para baixo). para usar é necessário definir let para chamar math.floor(let).

-porcentagem só é printada na tela por concatenação.



**4.BOOLEANS:**

-ou é true ou é false.

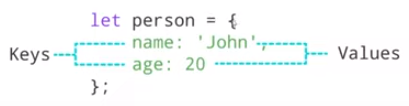
-declaração de variável com comparação (===) sempre retornará valor booleano.

**5.ARRAYS:** OU VETORES

-são listas iteráveis de elementos.

-podemos declarar por let um array [] ; dar .push para inserir no final; dar .pop para retirar do final; .shift para retirar do início; dar unshift para adicionar no início;

**6.OBJETOS:**



-estrutura do tipo chave e valor.

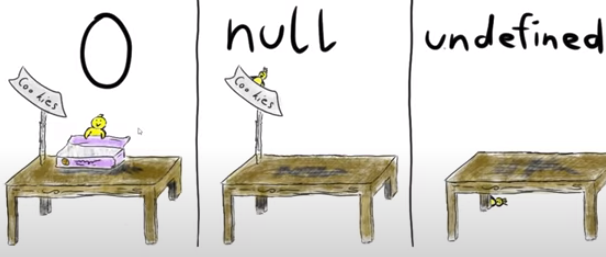
-declara a let e atribui valor.

-chama a let coloca ponto, define valor e atribui.

- chama object.value(let) para ver os valores; .keys para ver as chaves;

-para mandar uma key e value para let já declarada, usa-se chaves.

**7.EMPTY, UNDEFINED E NULL:**



-empty: zero

-null: propositalmente não é nada. Definido como null.

-undefined: não existe.

AULA 03 - ATIVIDADE PRÁTICA:

comentários no código no github.