Tópicos especiais em JS

Prof. Guilherme Alves

Tipagens

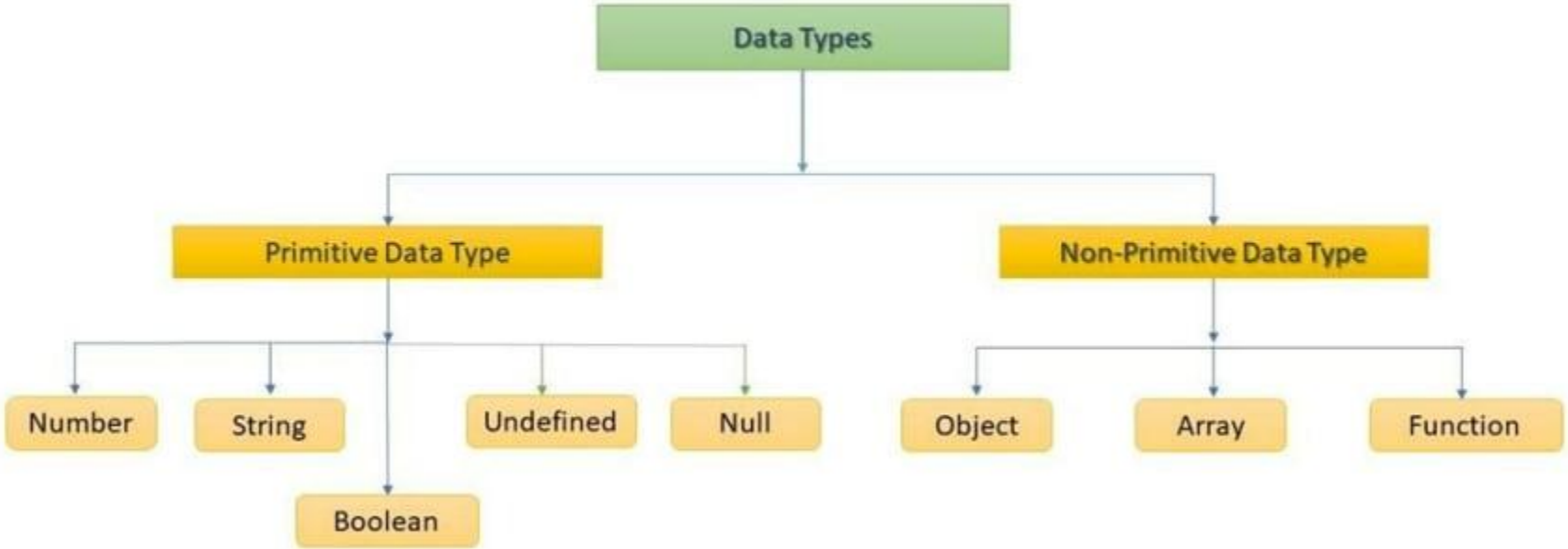
O que são linguagens fortemente tipadas, fracamente tipadas e não-tipadas?

Linguagens fortemente tipadas, são as linguagens que precisam ser declarados os tipos das variáveis.

Linguagem fracamente tipada, podemos dizer que objetos que tem todos os atributos de uma classe pode ser usado como se fosse uma instância dela.

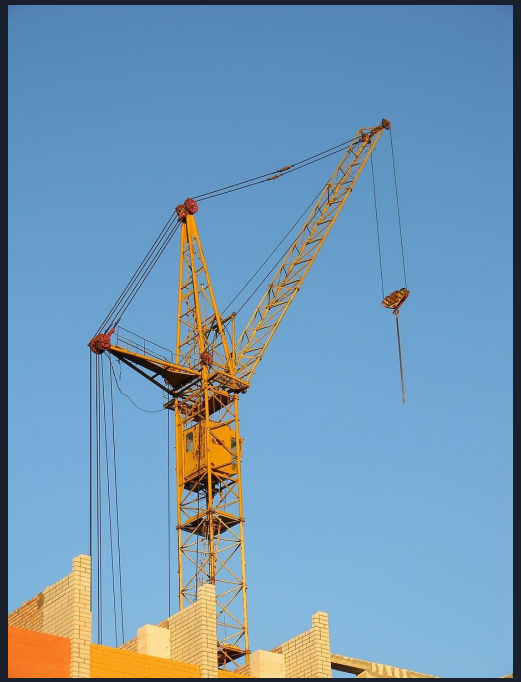
Por fim, nas linguagens não tipadas, não podemos inferir tipos específicos a uma variável.

E o que são linguagens dinamicamente tipadas?


Data Types
Primitive Data Type
Non-Primitive Data Type
Number
String
Boolean
Undefined
Null
Object
Array
Function

Hoisting

O hoisting ou içamento, permite que você use funções e variáveis antes de serem declaradas. Isso ocorre porque o interpretador divide a declaração e atribuição de funções e variáveis, levantando as declarações para o topo de seu escopo antes da execução. Funções também são içadas e se forem funções nomeadas podem ser invocadas antes da declaração sem utilizar a regra de declaração de variáveis.

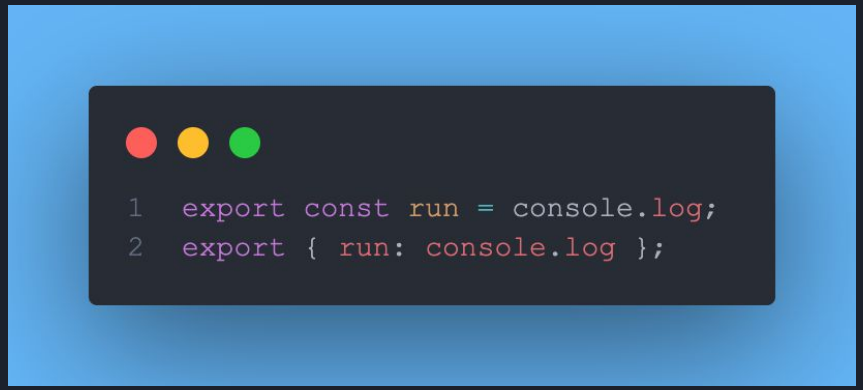


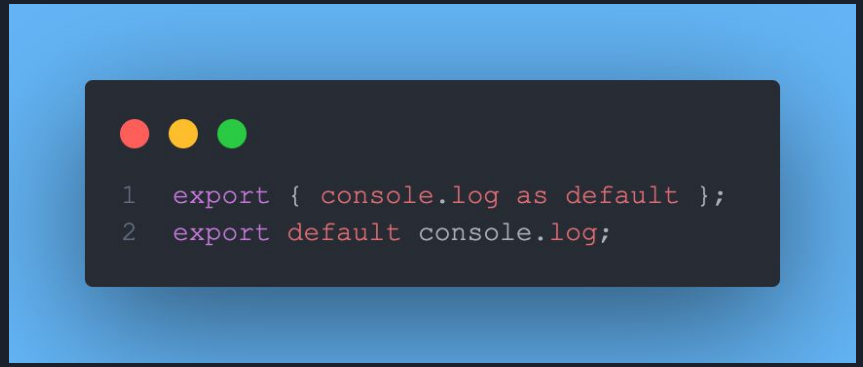
Scopo

O escopo define o contexto atual de execução. O escopo de uma variável é a parte de um programa onde ela está disponível para uso. As variáveis têm escopo lexical, o que significa que podemos determinar o escopo de uma variável de onde ela é declarada no fonte. Tipos *var* não possuem escopo lexical devido ao conceito de içamento. Podemos acessar variáveis do escopo em que elas são declaradas e de cada escopo interno. Em um navegador, o escopo global está no nível superior de um script, assim, criar uma variável no escopo global a torna acessível em todos os escopos. Logo antes da execução do código ocorre um processo chamado *lexing*, ou tokenização, onde uma sequência de caracteres é transformada em uma sequência de *tokens*. Nesse momento o escopo léxical é definido.

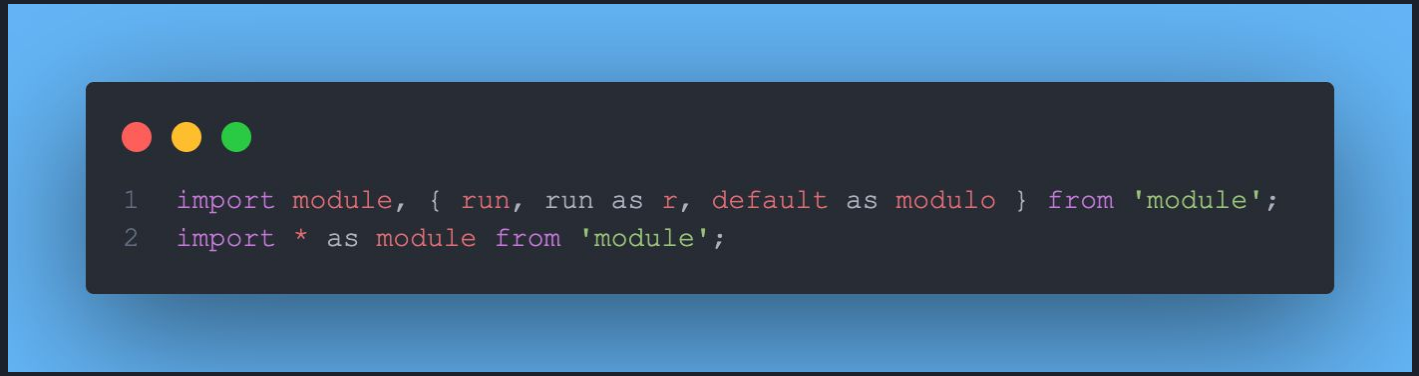
```
/*
Sempre que iniciamos nosso código, é criado um escopo global,
quando definimos funções e variáveis não aninhadas, de fato
elas são declaradas no escopo global. Existe apenas um escopo
global em um documento.
*/
{
    /*
    Sempre que definimos uma instrução de condição, loop ou dentro
    de dois conjuntos de colchetes, criamos um escopo de bloco.
    */

    function () {
      /*
      Sempre que definimos uma função com dois conjuntos de colchetes,
      criamos um escopo de função. Haverá um escopo de função diferente
      para cada chamada dessa função.
      */
    }
}
```

Módulos ES6


export const run = console.log;
export { run: console.log };


export { console.log as default };
export default console.log;

Módulos ES6


1 import module, { run, run as r, default as modulo } from 'module';
2 import * as module from 'module';