ntander Coders 2024.1 | Front-End | #1178

stema de Tipos (type system) da linguagem

Isaias Soares
Id: 1178008

Sistema de Tipos (type system) da linguagem

Existem duas vertentes de linguagens que tratam a alocação de memória de formas diferentes:

- **Estática**: Obriga a declaração dos tipos ao criarmos uma variável. Mesmo que a variável não tenha um valor ainda. Assim a memória pode ser imediatamente alocada.
- Linguagens de tipagem estática não permitem que o programador coloque em uma variável já declarada um novo valor que tenha um tipo diferente, por exemplo, um texto em uma variável numérica.

```
// Aqui vai ser uma string
string nome;

// Aqui gera um erro de tipo
nome = 2
```

- **Dinâmica**: Utiliza o valor colocado na variável para inferir que tipo utilizar. A memória só é alocada quando a variável receber um valor.
- Linguagens dinâmicas, com algumas exceções, permitem que o tipo do valor mude. Ela irá realocar a memória sempre que necessário.

```
// Aqui vai ser uma string
let nome = "João"

// Agora é um número
nome = 2

// Agora é booleana
nome = false
```

Chamamos o sistema de tipos que não permite a troca de tipos de uma variável de **fortemente tipado**, em oposição ao **fracamente tipado**, que permite a troca.

Javascript é uma linguagem **fracamente tipado e dinâmica**, por isso, não precisamos declarar os tipos das variáveis, apenas o nome, e podemos atribuir à mesma variável valores de qualquer tipo.

O JavaScript, conforme especificação da linguagem ECMAScript, tem apenas 6 tipos definidos, sendo eles:

Primitivos

- **Undefined**

Suporta o apenas valor: `undefined`

```
let nome = undefined
```

- **Null**

Suporta o apenas valor: `null`

```
let nome = null
```

- **Boolean**

Suporta os valores: `true` e `false`

```
let temNome = true;
temNome = false;
```

- String

Suporta os valores de texto. Esses valores são definidos por aspas (`"`) ou apóstrofos (`'`)

```
let nome = 'Algum nome'
nome = "Algum nome 2"
```

- Number

Suporta os valores numéricos `inteiros`, `decimais`, `NaN (Not a number = Não é um número)` e `Infinity`

```
let inteiro = 1
let decimal = 0.5

// NaN é retornado quando existe uma tentativa de operação matemática de um não número com um número
let naoENumero = NaN
naoENumero = "infinity" * 5 // O retorno será NaN

let infinitoPositivo = Infinity
let infinitonegativo = -Infinity
```

Complexos

- Object

Suporta valores complexos e múltiplas propriedades.

```
let carro = {
  numeroRenavam: '89579066136'
}
```

Referências e Materiais Complementares

- ECMAScript
- Sintaxe e tipos
- JavaScript's type system
- Onux

Próximo Tópico