12

Para saber mais: Tipos null e undefined

Nesta aula falamos sobre três tipos primitivos: `number`, `string` e `boolean`. Mas existem ainda mais dois tipos que não abordamos com profundidade: `null` e `undefined`.

O `null` é um tipo especial, pois pode ser traduzido como "ausência de valor" e pode ser atribuído como valor de uma variável:

```
let input = null;

if (input === null) {
 console.log('não há informação');
} else {
 console.log(input);
}
```

COPIAR CÓDIGO

Nesse caso, qual seria a diferença entre os dois casos abaixo?

```
let input = null;
let input2;

console.log(input); // null
console.log(input2); // undefined
```

COPIAR CÓDIGO

É aqui que entra o tipo `undefined`. Este tipo também representa "ausência de valor", porém de uma outra forma: usualmente, enquanto `null` é um valor

atribuído a uma variável que existe e foi iniciada, `undefined` se refere ao valor de uma variável que não foi inicializada (ou seja, não foi atribuído nenhum valor a ela).

> `undefined` também é o valor retornado por uma função que não tem cláusula `return`. Veremos mais sobre funções e `return` mais adiante no curso.

É importante notar que, embora os dois tipos sejam utilizados para sinalizar ausência de valor, os operadores de comparação do JavaScript podem ou não diferenciá-los:

```
console.log(null == undefined); // true
console.log(null === undefined); // false
```

COPIAR CÓDIGO

No cotidiano é comum considerar `undefined` como uma ausência de valor "inesperada" (causada por um *bug* ou erro no código) e `null` como um tipo de dado que também significa ausência de valor, mas não de maneira inesperada. Por exemplo, um campo em uma tabela de um banco de dados que esteja sem dados ou uma informação solicitada que não seja obrigatória e não tenha sido preenchida pelo usuário pode ter valor `null`.