06

Para saber mais: Palavras reservadas

Já aprendemos a declarar variáveis, sejam elas `let` ou `const`, utilizando a palavra-chave e um nome que escolhemos para a variável. Chamamos este nome justamente de **identificador**, e o ideal é que sejam sempre o mais explicativos possível:

```
let cpfUsuario = "12312312312"
```

COPIAR CÓDIGO

Mas o que acontece se tentarmos identificar uma variável com um termo que faça parte da linguagem, como nos casos abaixo?

```
let var = 0;
let if = 0;
let const = "Alura";
```

COPIAR CÓDIGO

Faça o teste para ver que o JavaScript não consegue reconhecer estas palavras-chave como identificadores e nem interpretar o que deve ser executado nestas linhas. Isso acontece porque `var`, `if` e `const` são **palavras reservadas** do JavaScript. Ou seja, não podemos usá-las para dar nomes (identificar) variáveis, funções ou outros blocos de código que precisem de identificadores.

Por outro lado, os exemplos abaixo são aceitáveis:

```
let varInicial = 0;
let ifFalse = 0;
let constDeTexto = "Alura";
```

COPIAR CÓDIGO

No JavaScript, algumas palavras são totalmente reservadas (não podem ser utilizadas como identificador em nenhum caso), enquanto outras podem ser utilizadas dependendo do contexto, e ainda outras não podem ser consideradas totalmente reservadas por razões de compatibilidade com versões mais antigas da linguagem, como é o caso de `let` (lembrando que, até o ES6, só era possível declarar variáveis com `var`). A palavra `let` vem do verbo em inglês "permitir", então não é possível deixá-la reservada pois pode já estar sendo usada como identificador em algum código mais antigo.

A melhor prática, nesse caso, é não utilizar nenhum dos termos da lista abaixo como identificadores, seja de variáveis, funções, classes ou qualquer outro bloco que precise de um nome. **As únicas exceções são `from`, `set` e `target`, que são seguras e comumente utilizadas.**

```
arguments
as
async
await
break
case
catch
class
const
continue
debugger
default
delete
```

```
do
else
eval
export
extends
false
finally
for
from
function
get
if
import
in
instanceof
let
of
new
null
return
set
static
super
switch
target
this
throw
true
try
typeof
var
void
while
with
yield
```

COPIAR CÓDIGO

Como as linguagens estão em constante desenvolvimento, o JavaScript também restringe o uso de mais algumas palavras que podem ser utilizadas nas próximas versões:

```
enum
implements
interface
package
private
protected
public
```

COPIAR CÓDIGO

Dica de boas práticas: sempre procure nomear/identificar seu código da forma mais **semântica** possível, pensando em qual é o dado que está sendo salvo na variável e para que ele será utilizado. Além de evitar palavras reservadas, faz com que o código fique mais compreensível e de leitura mais fluida.