07

Para saber mais: Deletando propriedades

Já vimos o principal, que é acessar, alterar e adicionar valores em um objeto. Mas ainda faltou falarmos sobre um caso: E quando queremos deletar um conjunto de chave/valor?

Vamos considerar o objeto abaixo:

```
const objPersonagem = {
 nome: "Gandalf",
 classe: "mago",
 nivel: "20",
 aliado: {
   nome: "Saruman",
   classe: "mago"
 },
 status: "desaparecido"
}
```

COPIAR CÓDIGO

Se quisermos, por exemplo, remover a propriedade `aliado`, podemos utilizar o operador `delete`:

```
delete objPersonagem.aliado

console.log(objPersonagem.aliado) //undefined
```

COPIAR CÓDIGO

Também é possível utilizar a notação de colchetes:

```
delete objPersonagem.aliado
delete objPersonagem["status"]

console.log(objPersonagem.aliado) //undefined
console.log(objPersonagem.status) //undefined
```

COPIAR CÓDIGO

Importante! Veja que o `delete` remove do objeto **o valor da propriedade, assim como a chave**.

Após remover as duas propriedades acima, o objeto agora está desta forma:

```
{
 nome: "Gandalf",
 classe: "mago",
 nivel: "20",
}
```

COPIAR CÓDIGO

O valor de retorno do operador `delete` é um booleano, ou seja, retorna sempre `true` ou `false` para cada operação. Porém, é importante notar que ele **não** retorna `false` se tentarmos remover, por exemplo, uma propriedade que não existe no objeto:

```
const delProp = delete objPersonagem.aliado
const delPropInexistente = delete objPersonagem["endereco
```

```
console.log(delProp) //true
console.log(delPropInexistente) //true
```

COPIAR CÓDIGO

O operador `delete` não remove propriedades herdadas de outro objeto, e aí sim, neste caso, retornará `false` se tentarmos fazer isso. Vamos entender melhor o tema “heranças” mais adiante no curso.