04

Para saber mais: Tipos de erro

Enquanto trabalhamos com programação, passamos uma boa parte do tempo lidando com os erros que aparecem em nosso código. Isso é totalmente normal, pois eles nos ajudam a resolver problemas. Já imaginou como seria muito mais difícil programar sem um recurso para dizer onde estamos errando e como corrigir?

Como vimos, as pessoas que desenvolvem os programas e linguagens são as responsáveis pelo chamado "tratamento de erros", ou seja, permitir a comunicação de quais foram os problemas, e nem sempre isso acontece da melhor forma - embora já tenha melhorado muito nas últimas décadas.

Cada linguagem de programação tem sua própria forma de lidar com erros. O JavaScript começa dividindo cada tipo de erro possível em algumas categorias:

- `RangeError`: Quando o código recebe um dado do tipo certo, porém não dentro do formato aceitável. Por exemplo, um processamento que só pode ser feito com números inteiros maiores ou iguais a zero, mas recebe `-1`.
- `ReferenceError`: Normalmente ocorre quando o código tenta acessar algo que não existe, como uma variável que não foi definida; muitas vezes é causado por erros de digitação ou confusão nos nomes utilizados, mas também pode indicar um erro no programa.
- `SyntaxError`: Na maior parte dos casos ocorre quando há erros no programa e o JavaScript não consegue executá-lo. Os erros podem ser métodos ou propriedades escritos ou utilizados de forma incorreta, por exemplo, operadores ou sinais gráficos com elementos a menos, como esquecer de fechar chaves ou colchetes.
- `TypeError`: Indica que o código esperava receber um dado de um determinado tipo, tal qual uma string de texto, mas recebeu outro, como

um número, booleano ou null.

O NodeJS trabalha com outros tipos específicos de erro que não vamos abordar neste momento, mas que você pode sempre consultar na documentação oficial (https://nodejs.org/api/errors.html#errors_errors).

> Além do tipo de erro, o terminal também vai dar outras informações, como o nome do arquivo e linha onde foi detectado o erro. Muitas vezes isso já basta para identificar e corrigir, mas existem também casos onde o erro não é detectado pelo JavaScript na linha onde o código é declarado, por exemplo, mas onde ele é executado. Por isso é importante praticar sempre a leitura dos erros e da *stacktrace* e nunca pular esta etapa.

No futuro, ao trabalhar em suas aplicações, você também deverá criar seus próprios avisos de erro para ajudar os usuários.